Avec les compliments de IBM

2ème Edition Limitée IBM

DevOps

POUR LES NULS®

Vous apprendrez :

- La valeur métier de DevOps
- Les fonctionnalités et chemins d'adoption de DevOps
- Comment le Cloud accélère DevOps
- Les dix mythes sur DevOps

Sanjeev Sharma
Bernie Coyne

DevOps

POUR

LES NULS®

2ème Edition Limitée IBM

de Sanjeev Sharma et Bernie Coyne

WILEY

DevOps pour les Nuls®, 2ème Edition Limitée IBM
Publié par
John Wiley & Sons, Inc.
111 River St.
Hoboken, NJ 07030-5774
www.wiley.com

ISBN: 978-1-119-17743-2 (pbk); ISBN: 978-1-119-17744-9 (ebk)

Fabriqué aux Etats-Unis

10 9 8 7 6 5 4 3 2 1

Remerciements de l'éditeur

Voici une liste non exhaustive des personnes qui ont contribué à la publication de cet ouvrage :

Rédacteur du projet : Carrie A. Johnson
Responsable des acquistions : Katie Mohr
Responsable éditorial : Rev Mengle
Délégué du développement : Sue Blessing
Coordinateur de production : Melissa Cossell

Table des matières

Introduction

DevOps (abréviation de développement et opérations), à l'instar de nombreuses nouvelles approches, est souvent un mot à la mode pour beaucoup de personnes. Tout le monde en parle, mais tout le monde ne sait pas ce que c'est. De manière générale, DevOps est une approche qui repose sur les principes Lean et Agile dans lesquels les responsables métiers avec les services de développement, des opérations et d'assurance qualité collaborent pour délivrer le logiciel en continu dans l'objectif de permettre à l'entreprise de saisir plus rapidement les opportunités du marché et d'accélérer la prise en compte des retours clients. En effet, les applications d'entreprise sont si diverses et composées de tant de technologies, bases de données, d'équipements utilisateurs, etc., que seule une approche DevOps permet de gérer avec succès toute cette complexité. Cependant, les opinions sur son utilisation divergent.

Certains affirment que DevOps s'adresse aux professionnels uniquement. D'autres avancent qu'il tourne autour du Cloud. IBM propose une vue large et holistique et considère DevOps comme une approche de livraison du logiciel orientée métier : une approche qui porte une fonctionnalité nouvelle ou une amélioration de l'idée jusqu'à la mise en production, en fournissant de la valeur métier aux clients finaux de manière efficace et en capturant les retours des utilisateurs de cette nouvelle fonctionnalité. Pour ce faire, la participation de toutes les parties prenantes, bien au-delà des équipes de développement ou des opérations, est indispensable. Une véritable approche DevOps englobe les lignes métiers, les utilisateurs, les responsables, les partenaires, les fournisseurs, etc.

A propos de cet ouvrage

Ce manuel propose une approche métier de DevOps. Face à des évolutions de plus en plus rapides, DevOps devient essentiel pour les entreprises qui doivent être suffisamment agiles et efficaces pour répondre rapidement aux changements imposés par les demandes des clients, les conditions du marché, les pressions de la concurrence ou les réglementations.

Si vous lisez ce manuel, nous supposons que vous avez entendu parler de DevOps, mais que vous voulez connaître ce qu'il signifie et les avantages qu'il peut apporter à votre entreprise. Ce manuel

s'adresse aux dirigeants, aux décideurs et aux utilisateurs qui ne connaissent pas DevOps, qui veulent en savoir plus sur l'approche et qui ne désirent ne pas se limiter aux seuls commentaires sur le concept pour en comprendre tous les avantages qu'il peut apporter.

Les icônes utilisées dans ce livre

Des icônes figurent dans les marges du manuel. Leur signification.

L'icône Truc souligne des informations particulièrement utiles sur différents aspects de DevOps.

Les points signalés par une icône N'oubliez pas ! signalent des points à bien garder à l'esprit.

L'icône Attention ! indique des informations qui peuvent être critiques.

Les informations techniques permettent d'aller au-delà des fondamentaux de DevOps. Cependant elles ne sont pas essentielles.

Après ce manuel

Vous pouvez en savoir plus sur DevOps, l'approche et les services IBM disponibles en visitant les pages Web suivantes :

- **IBM DevOps Solution:** `ibm.com/devops`
- **DevOps: the IBM approach (livre blanc) :** `ibm.biz/BdEnBz`
- **The Software Edge (enquête) :** `ibm.co/156KdoO`
- **Adopting the IBM DevOps Approach (article):** `ibm.biz/adoptingdevops`
- **DevOps Services for Bluemix (service):** `bluemix.net`

Chapitre 1

Qu'est-ce que DevOps ?

Dans ce chapitre

- Examen d'un besoin métier de DevOps
- Recherche de valeur métier dans DevOps
- Description des principes DevOps

Changer les habitudes de travail est toujours compliqué et demande un certain investissement. En conséquence, lorsqu'une organisation adopte de nouvelles technologies, méthodologies ou approches, cette adoption doit être motivée par un besoin métier. Pour développer une étude de rentabilité à l'adoption de DevOps, vous devez comprendre le besoin métier associé. Ce chapitre porte sur les fondamentaux nécessaires à l'élaboration de l'analyse du besoin.

Description du besoin métier pour DevOps

Les organisations veulent créer des applications ou services innovants pour résoudre les problèmes métier. L'intention peut être de résoudre des problèmes métier internes (tels que créer un système de gestion de la relation client plus performant) ou d'aider leurs clients ou utilisateurs finaux (en fournissant une nouvelle application mobile).

Cependant, de nombreuses organisations ne mènent pas à bien leurs projets logiciels, et leurs échecs sont généralement liés aux défis inhérents au développement et à la livraison des logiciels. Bien que la majorité des entreprises aient conscience que le développement et la livraison des logiciels sont des activités essentielles, une étude récente d'IBM dans le secteur indique que seulement 25 % d'entre elles ont le sentiment que leurs équipes sont efficaces. Ces échecs dans la réalisation se traduisent par des pertes d'opportunités commerciales.

Ces difficultés sont amplifiées par une évolution majeure des types d'applications que les entreprises doivent fournir, des systèmes d'enregistrement aux systèmes d'engagement.

- **Systèmes d'enregistrement :** les applications logicielles traditionnelles sont de grands systèmes qui fonctionnent comme des systèmes d'enregistrement contenant d'énormes volumes de données et/ou de transactions, et qui sont conçus pour être très fiables et stables. Comme ces applications n'ont pas besoin d'être modifiées fréquemment, les entreprises peuvent satisfaire leurs clients ou leurs propres besoins métier en fournissant une ou deux nouvelles versions majeures chaque année.
- **Systèmes d'engagement :** avec l'avènement des communications mobiles et l'évolution des applications Web, les systèmes d'enregistrement sont complétés par des systèmes d'engagement auxquels les clients peuvent accéder directement et utiliser pour interagir avec l'entreprise. Ces applications doivent être simples à utiliser, très performantes et pouvoir être modifiées rapidement pour répondre à l'évolution des besoins des clients et des contraintes du marché.

Comme les systèmes d'engagement sont utilisés directement par les clients, l'expérience utilisateur, la rapidité de livraison et l'agilité revêtent une importance extrême. En d'autres termes, ils nécessitent d'adopter une approche DevOps.

Les systèmes d'engagement ne sont pas des systèmes isolés et ils sont souvent liés à des systèmes d'enregistrement. Par conséquent, lorsque les systèmes d'engagement changent, les systèmes d'enregistrement doivent changer aussi. En fait, tout système nécessitant une mise à disposition rapide des innovations requiert DevOps. Ces innovations sont issues principalement des technologies émergentes, telles que le Cloud Computing, les applications mobiles, le Big Data et les réseaux sociaux, qui peuvent concerner tous les types de systèmes. Nous aborderons ces technologies émergentes à la lumière de DevOps dans les chapitres 4 et 5.

Identification de la valeur métier de DevOps

DevOps applique les principes Agile et Lean à l'ensemble de la chaîne logistique logicielle. Il permet à une entreprise d'optimiser la rapidité de livraison d'un produit ou d'un service, de l'idée initiale à la version en production, aux retours client et aux améliorations apportées en réponse à ces retours.

Comme DevOps améliore la manière dont une entreprise apporte de la valeur à ses clients, fournisseurs et partenaires, il constitue un processus métier essentiel, et non une simple fonctionnalité informatique.

DevOps génère un retour sur investissement significatif dans trois domaines :

- Amélioration de l'expérience client
- Accroissement de la capacité à innover
- Accélération du retour sur investissement

Les sections suivantes portent sur ces trois domaines.

Amélioration de l'expérience client

L'amélioration de l'expérience client (différenciée et engageante) permet de fidéliser les clients et d'accroître les parts de marché. Pour apporter cette expérience, une entreprise doit obtenir et répondre en continu aux retours client, ce qui nécessite des mécanismes pour recevoir ces informations de toutes les parties prenantes impliquées dans l'application logicielle à délivrer : clients, secteurs d'activité, utilisateurs, fournisseurs, partenaires, etc.

Dans l'environnement actuel des systèmes d'engagement (voir " Description du besoin métier pour DevOps" dans les pages précédentes de ce chapitre), cette capacité à réagir et s'adapter avec agilité améliore l'expérience et la fidélité du client.

Accroissement de la capacité à innover

Les organisations actuelles appliquent des approches Lean pour renforcer leur capacité à innover. Leurs objectifs visent à réduire le gaspillage et la reprise de travail et à dédier les ressources aux activités à plus forte valeur ajoutée.

Les *tests A-B* sont un exemple de pratiques courantes dans l'approche Lean ; dans ce cas de figure, des entreprises demandent à un petit groupe d'utilisateurs de tester et d'évaluer deux versions logiciels ou plus, ayant des fonctionnalités différentes. Ensuite, celui qui fournit les meilleures fonctionnalités est déployé vers tous les utilisateurs et la version rejetée est annulée. Ces tests A-B sont réalistes uniquement avec des mécanismes efficaces et automatisés, tels que ceux qu'offre DevOps.

Accélération de la production de valeur

L'accélération de la production de valeur implique de développer une culture, des pratiques et une automatisation afin de délivrer les logiciels rapidement, efficacement et de manière fiable jusqu'à la mise en production. DevOps, adopté comme fonctionnalité métier, apporte les outils et la culture nécessaires à la planification efficace, à la prévisibilité et au succès des nouvelles versions logicielles.

La définition de *valeur* change d'une entreprise à l'autre, et même d'un projet à l'autre, mais l'objectif de DevOps vise à fournir cette valeur plus rapidement et plus efficacement.

Fonctionnement de DevOps

Le mouvement DevOps a produit divers principes qui ont évolué dans le temps et qui continuent d'évoluer. Des fournisseurs de solutions, notamment IBM, ont développé leurs propres variantes de DevOps. Cependant, tous ces principes adoptent une approche holistique de DevOps, et les organisations de toutes tailles peuvent les adopter. Ces principes sont les suivants :

- Développement et test sur des systèmes similaires à ceux de la production
- Déploiement avec des processus réutilisables et fiables
- Surveillance et validation de la qualité opérationnelle
- Amplification des boucles de retour

Ces principes sont expliqués plus en détail dans les sections suivantes :

Développement et test sur des systèmes de type 'production'

Ce principe provient du concept *shift left,* de DevOps dans lequel les problèmes de la production sont traités en amont dans le cycle de vie de distribution de logiciel jusqu'au développement (voir l'illustration 1-1).

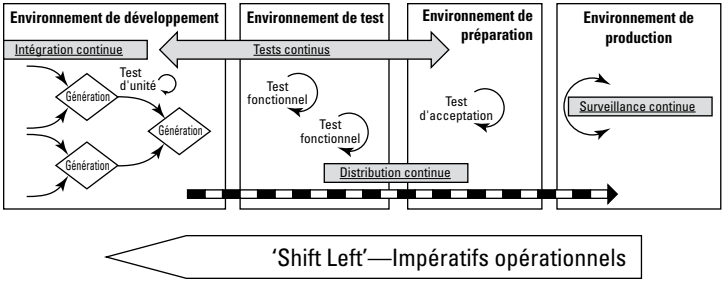


Illustration 1-1 : le concept Sift-left traite en amont les opérations dans le cycle de vie de développement

L'objectif vise à permettre aux équipes de développement et d'assurance qualité (QA) de développer et de tester l'application par rapport à des systèmes qui se comportent comme des systèmes de production afin de déterminer le comportement et les performances de l'application avant son déploiement.

La première exécution de l'application sur un système de type production doit intervenir le plus tôt possible dans le cycle de vie pour résoudre deux problèmes potentiels majeurs. Cela permet d'une part, de tester l'application dans un environnement similaire à l'environnement de production réel où l'application sera déployée, et d'autre part de tester et de valider les processus de mise en production eux-mêmes en amont.

Du point de vue du développement, également, ce principe est très efficace. Il permet à l'équipe des opérations de déterminer en amont dans le cycle la manière dont se comportera son environnent lors de la prise en charge de l'application, et de créer un environnement d'application optimisé.

Déploiement de processus réutilisables et fiables

Comme l'indique le terme, ce principe permet aux équipes de développement et des opérations de prendre en charge un processus de développement de logiciel agile (ou au minimum itératif) jusqu'à la production. L'automatisation est essentielle pour créer des processus itératifs, fréquents, réutilisables et fiables. Par conséquent, l'organisation doit créer un pipeline de livraison qui permet le déploiement et les tests de manière automatisée et en continu. Les pipelines de livraison sont expliqués plus en détail dans le chapitre 3.

Les déploiements fréquents donnent également la possibilité aux équipes de tester les processus de déploiement eux-mêmes et donc de réduire les risques d'échec de déploiement lors de la mise en production.

Surveillance et validation de la qualité opérationnelle

En règle générale, les organisations surveillent efficacement les applications et les systèmes dans l'environnement de production, car elles disposent des outils qui capturent les mesures des systèmes de production en temps réel. Mais elles les surveillent d'une manière isolée et déconnectée : ce principe place en amont la surveillance dans le cycle de vie en exigeant que les tests automatisés soient exécutés plus tôt et plus fréquemment dans le cycle de vie pour surveiller les caractéristiques fonctionnelles et non-fonctionnelles de l'application. Lorsqu'une application est déployée et testée, des mesures de qualité doivent être capturées et analysées. La surveillance fréquente permet d'être alerté en amont des problèmes opérationnels et de qualité qui peuvent apparaître dans l'environnement de production.

Ces mesures doivent être capturées dans un format compréhensible et utilisable par toutes les parties prenantes.

Amplification des boucles de retour

L'un des objectifs de DevOps est de permettre aux organisations de réagir et de procéder aux modifications plus rapidement. Dans la livraison des logiciels, cet objectif suppose qu'une organisation obtienne rapidement un retour et apprenne tout aussi rapidement de chaque action qu'elle exécute. Ce principe implique que les organisations créent des canaux de communication pour que les parties prenantes puissent accéder aux différents retours et agir en conséquence.

- Le développement peut agir en ajustant ses plans projet ou ses priorités.
- La production peut agir en améliorant les environnements de production.
- L'entreprise peut agir en modifiant les plans de mise en production des nouvelles versions.

Chapitre 2

Examen des fonctionnalités DevOps

Dans ce chapitre

- Description de l'architecture de référence DevOps
- Quatre chemins d'adoption de DevOps

Les fonctionnalités qui constituent DevOps sont un large ensemble qui couvre le cycle de vie de distribution des logiciels. Le point à partir duquel une organisation peut initier une démarche DevOps dépend de ses objectifs métier — les défis qu'elle tente de relever et les insuffisances qu'elle doit combler dans ses fonctionnalités de livraisons de logiciels.

Ce chapitre examine une architecture de référence DevOps et les différentes manières qui permettent à une entreprise d'entreprendre une démarche DevOps.

Chemins d'adoption de DevOps

Une *architecture de référence* fournit un modèle d'une solution éprouvée en utilisant un ensemble de méthodes et fonctionnalités préférées. L'architecture de référence DevOps traitée dans ce chapitre, aide les utilisateurs à accéder et utiliser les instructions, les directives et d'autres documents nécessaires afin de structurer et concevoir une plateforme DevOps qui s'adapte aux personnes, processus et technologies (voir le chapitre 3).

Une architecture de référence peut fournir des fonctionnalités via divers composants. Chacune de ces fonctionnalités peut être couverte par un seul composant ou un groupe de composants fonctionnant conjointement. Par conséquent, vous pouvez voir l'architecture

de référence DevOps, représentée dans l'illustration 2-1, du point de vue des principales fonctionnalités qu'elle est supposée fournir. Au fur et à mesure que l'architecture définie se concrétise, ces fonctionnalités seront supportées par un groupe de personnes compétentes, de pratiques définies et d'outils d'automatisation.

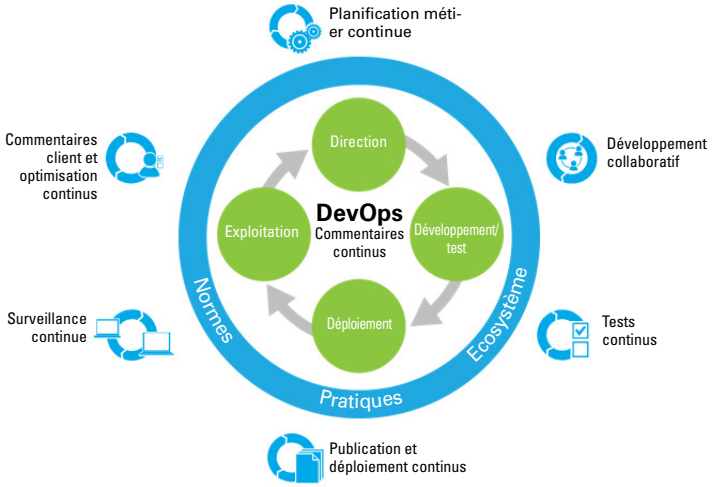


Illustration 2-1 : l'architecture de référence DevOps.

L'architecture de référence DevOps de l'illustration 2-1 propose les quatre groupes de chemins d'adoption suivants :

- Pilotage
- Développement/test
- Déploiement
- Opération

Les sections suivantes du chapitre examinent en détail ces chemins d'adoption.

Pilotage

Ce chemin d'adoption est constitué d'une seule pratique axée sur la définition des objectifs métier et leur ajustement en fonction des retours des clients : *planification métier continue*.

Aujourd'hui, les entreprises doivent être agiles et pouvoir réagir rapidement aux retours des clients. La réalisation de cet objectif repose sur la capacité d'une organisation à faire les choses de manière juste et appropriée. Malheureusement, les approches traditionnelles en matière de livraison de logiciels sont trop lentes par rapport à la vitesse actuelle du monde des affaires, en partie parce que ces approches dépendent de processus de développement personnalisés et manuels, et que les équipes travaillent de manière isolée en silos. Les informations requises pour planifier et re-planifier rapidement, tout en optimisant la capacité de fournir de la valeur, sont généralement fragmentées et incohérentes. Souvent, le retour d'utilisation pertinent n'est pas reçu suffisamment en amont pour atteindre le niveau de qualité approprié et la véritable valeur à fournir.

En outre, les équipes éprouvent des difficultés à prendre en compte les retours susceptibles de prioriser les investissements et du coup à collaborer en tant qu'organisation capable de gérer correctement l'exécution d'un un modèle de livraison continue. Certains équipes considèrent la planification comme une tâche de gouvernance intrusive qui les ralentit, et non pas comme une activité qui leur permet de fournir de la valeur rapidement.

Une capacité à délivrer le logiciel plus rapidement rend l'entreprise plus agile, mais il faut également gérer la vitesse avec l'assurance que ce qui sera délivré correspondra à ce qui était attendu. Vous ne pouvez distribuer rapidement des logiciels si vous ne croyez pas en la justesse de vos objectifs métier, vos mesures de suivi et d'analyse et votre infrastructure.

DevOps permet de rapprocher ces perspectives antagonistes en aidant les équipes à définir ensemble les objectifs métier et à les modifier en continu en fonction des retours de manière à améliorer à la fois l'agilité et les résultats d'entreprise. En parallèle, les entreprises doivent gérer les coûts. En identifiant et éliminant les gaspillages dans le processus de développement, les équipes deviennent plus efficaces mais améliorent également les problèmes de coût. Cette approche permet aux équipes d'établir un équilibre optimal entre tous ces points, dans toutes les phases du cycle de vie DevOps vers un modèle de distribution continue.

Développement/test

Ce chemin d'adoption met en œuvre deux pratiques : le développement collaboratif et les tests continus. Comme tel, il forme le cœur des fonctionnalités de développement et d'assurance qualité (QA).

Développement collaboratif

La distribution des logiciels dans une entreprise implique un grand nombre d'équipes multifonctionnelles, des analystes métier, des architectes d'entreprise et logiciels, des développeurs, des spécialistes Assurance-Qualité, des responsables des opérations, des experts en sécurité, des fournisseurs et des partenaires. Les utilisateurs de ces équipes travaillent sur différentes plateformes et peuvent être répartis entre plusieurs sites. Le *développement collaboratif* leur permet de collaborer en fournissant un ensemble commun de pratiques et une plateforme commune qu'ils peuvent utiliser pour créer et délivrer les logiciels.

L'une des principales fonctionnalités dans le développement collaboratif est l'*intégration continue* (voir l'illustration 2-2), une pratique dans laquelle les développeurs de logiciels intègrent en continu ou fréquemment leur travail à celui des autres membres de l'équipe de développement.

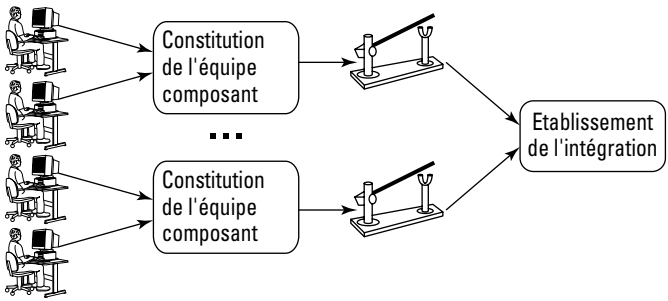


Illustration 2-2 : la collaboration via l'intégration continue.

L'intégration continue a été vulgarisée par la mouvance Agile. Le concept consiste pour les développeurs à intégrer leur travail à celui des autres développeurs de leur équipe, puis à tester le travail intégré. Dans le cas des systèmes complexes constitués de plusieurs systèmes ou services, les développeurs intègrent également régulièrement leur travail à d'autres systèmes et services. L'intégration régulière du travail produit permet d'identifier et de révéler en amont les risques d'intégration. Dans les systèmes complexes, elle dévoile les risques connus et inconnus — à la fois techniques et relatifs à la planification.

Tests continus

L'intégration continue (voir la section précédente) a plusieurs objectifs :

- Permettre les tests et la vérification continus du code
- Vérifier que le code produit et intégré à celui des autres développeurs et des autres composants de l'application fonctionne et s'exécute comme prévu
- Tester en continu l'application en cours de développement

Les *tests continus* impliquent de tester au plus tôt et en continu tout au long du cycle de vie, ce qui permet de réduire les coûts, de raccourcir les phases de test et de disposer de retours en continu sur la qualité. Ce processus, appelé également *shift-left testing* vise à intégrer les activités de développement et de test pour que la qualité soit incorporée aussi tôt que possible dans le cycle de vie et non pas repoussée à une phase ultérieure. Cela est facilité par l'adoption de fonctionnalités telle que l'automatisation des tests et la virtualisation des services. La virtualisation des services est une nouvelle fonctionnalité de simulation des environnements de type production qui rend possibles les tests continus.

Déploiement

Le chemin d'adoption Déploiement constitue la source des principales fonctionnalités de DevOps. La livraison et le déploiement continus font passer le concept d'intégration continue à l'étape suivante. La pratique qui permet la livraison et le déploiement permet également de créer un pipeline de distribution (voir le chapitre 3). Ce pipeline facilite le déploiement continu des logiciels vers les environnements d'assurance qualité, puis vers la production de manière automatisée et efficace. L'objectif de la livraison et du déploiement continus vise à fournir aux clients et aux utilisateurs les nouvelles fonctionnalités dès que possible.

La plupart des outils et des processus qui constituent le cœur de la technologie DevOps existent pour faciliter l'intégration, la livraison et le déploiement en continu. Ces points sont abordés en détail dans les chapitres suivants.

Opérations

Le chemin d'adoption des opérations inclut deux pratiques qui permettent aux entreprises de surveiller le fonctionnement des applications mises en production et d'avoir des retours clients. Ces données permettent aux entreprises d'interagir avec agilité et de changer leurs plans métier, si nécessaire.

Surveillance continue

La *surveillance continue* fournit aux équipes des Opérations, d'Assurance-Qualité, de Développement, aux responsables de secteur et aux autres parties prenantes des données et des mesures sur les applications à différentes étapes du cycle de livraison.

Ces mesures ne sont pas limitées à la production. Elles permettent aux parties prenantes de réagir en améliorant ou changeant les fonctionnalités à délivrer et/ou les plans métier nécessaires.

Retours continus des clients et optimisation

Les deux types d'informations les plus importants que peut obtenir une équipe chargée de la livraison du logiciel sont la manière dont les utilisateurs utilisent l'application et leurs commentaires sur son utilisation. Avec les nouvelles technologies, les entreprises peuvent capturer en temps-réel le comportement et les difficultés des clients lors de l'utilisation de l'application. Les parties prenantes peuvent s'appuyer sur ces retours afin de prendre les mesures appropriées dans l'objectif d'améliorer les applications et l'expérience des clients. Les responsables de secteur peuvent revoir leurs plans, le développement peut ajuster les fonctionnalités qu'il décide de livrer et le service des opérations peut améliorer l'environnement dans lequel l'application est déployée. Cette boucle de retours continus est un composant essentiel de DevOps, puisqu'elle permet aux entreprises d'être plus agiles et réactives aux besoins des clients.

Chapitre 3

Adoption de DevOps

Dans ce chapitre

- Efficacité des individus
- Rationalisation des processus
- Choix des bons outils

L'adoption d'une nouvelle fonctionnalité nécessite d'élaborer un plan qui englobe les personnes, les processus et les technologies. L'adoption de nouvelles fonctionnalités est vouée à l'échec — notamment dans une entreprise impliquant de nombreuses parties prenantes, potentiellement réparties — sans tenir compte de ces trois aspects à adopter.

Ce chapitre porte sur les aspects DevOps liés aux personnes, aux processus et aux technologies.

Bien que le terme *DevOps* fasse référence aux fonctionnalités de développement et des opérations, DevOps est une fonctionnalité d'entreprise qui englobe toutes les parties prenantes d'une organisation, notamment, les métiers, l'architecture, la conception, le développement, l'assurance qualité (QA), les opérations, la sécurité, les partenaires et les fournisseurs. L'exclusion d'une de ces parties prenantes — internes ou externes — se traduira par une implémentation incomplète de DevOps.

Savoir où commencer

Cette section explique comment démarrer avec DevOps, notamment créer la culture appropriée, identifier les défis économiques et déterminer les goulots d'étranglement à éliminer.

Identification des objectifs métier

La première tâche de création d'une culture consiste à amener les personnes à suivre la même direction et à atteindre le même objectif. Cela implique d'identifier les objectifs métier communs des équipes et de l'entreprise dans son ensemble. Il est plus important de récompenser l'ensemble de l'équipe en fonction des résultats de l'entreprise plutôt que des incitations d'équipe conflictuelles. Lorsque les personnes connaissent l'objectif commun et savent comment l'avancement de sa réalisation sera mesuré, les défis provenant des équipes et des utilisateurs ayant leurs propres priorités sont moindres.

DevOps n'est pas un objectif. Il donne les moyens d'atteindre des objectifs.

Les chapitres 4 et 5 mettent en lumière plusieurs nouveaux défis que DevOps permet de surmonter. Votre entreprise peut utiliser ces défis comme point de départ pour identifier les objectifs à atteindre. Ensuite, elle peut développer un ensemble de jalons communs vers l'atteinte de ces objectifs à l'attention des différentes équipes et parties prenantes.

Identification des goulots d'étranglement dans le pipeline de distribution

Les plus importantes sources d'inefficacités dans le pipeline de distribution peuvent être regroupées dans les catégories suivantes :

- Tâches inutiles (avoir à communiquer de manière répétitive les mêmes informations et connaissances)
- Reprise de travail inutile (défauts non détectés lors des tests ou de la production impliquant de réaffecter le travail à l'équipe de développement)
- Surproduction (fonctionnalités développées qui ne sont pas nécessaires)

L'un des principaux goulots d'étranglement dans le pipeline de distribution est le déploiement de l'infrastructure. L'adoption de DevOps accélère la distribution des applications et impose à l'infrastructure d'être plus réactive. Les environnements SDE

(Software-Defined Environment) permettent de capturer l'infrastructure comme un type de modèle programmable et réutilisable qui accélère de ce fait les déploiements. Consultez la section «L'infrastructure programmable» dans les pages suivantes de ce chapitre pour plus d'informations.

En outre, vous pouvez vouloir optimiser le pipeline avec un flux régulier de bout en bout. Le rythme de chaque processus doit être identique pour éviter les goulots d'étranglement. Pour atteindre cet équilibre, vous devez instrumenter ou mesurer le pipeline de distribution à des points clés afin de réduire le temps d'attente dans les files d'attente, optimiser le travail en cours et ajuster en permanence la capacité et le flux.

Les individus dans DevOps

Cette section porte sur l'aspect humain de l'adoption de DevOps et la création de la culture nécessaire.

La culture DevOps

A l'origine, DevOps est un mouvement culturel ; il concerne les individus. Une organisation peut adopter les processus ou les outils automatisés les plus efficaces possibles, mais sans les personnes qui doivent exécuter ces processus et utiliser ces outils, ils sont inutiles. La création d'une culture DevOps, par conséquent, est au centre de l'adoption de DevOps.

Une culture DevOps se caractérise par un haut niveau de collaboration dans les rôles, par une concentration sur les objectifs de l'entreprise et non pas sur ceux de départements, sur la confiance et l'importance que revêt l'apprentissage par l'expérimentation.

La création d'une culture DevOps ne revient pas à adopter un processus ou un outil. Elle nécessite (il n'y a pas de meilleur terme) une ingénierie sociale d'équipes d'individus ayant chacun des prédispositions, des expériences et des défauts. Cette diversité peut compliquer la création d'une culture.

Les pratiques de transformation Lean et Agile, telles que Scaled Agile Framework (SAFe), Disciplined Agile Delivery (DAD) et Scrum, sont au cœur de DevOps, et si votre organisation les applique déjà, vous pouvez les exploiter pour adopter une culture DevOps.

Mesure d'une culture

Mesurer une culture est extrêmement compliqué. Comment pouvez-vous mesurer précisément l'amélioration de la collaboration ou l'état d'esprit ? Vous pouvez mesurer directement les attitudes et l'état d'esprit des équipes en procédant à des enquêtes, mais ces dernières peuvent avoir un fort taux d'erreurs statistiques, car généralement les équipes sont petites.

Inversement, vous pouvez mesurer indirectement en déterminant la fréquence à laquelle un membre de l'équipe de développement communique avec un membre de l'équipe des opérations ou de l'équipe QA pour collaborer à la résolution d'un problème sans passer par des canaux officiels ou différents niveaux hiérarchiques.

Collaboration et communication entre les parties prenantes — c'est bien la culture DevOps.

La création d'une culture DevOps implique que les responsables de l'organisation collaborent avec leurs équipes pour créer un environnement et une culture de collaboration et de partage. Les responsables doivent éliminer les barrières de coopération que les personnes érigent elles-mêmes. Les mesures standard permettent de féliciter les équipes des opérations pour le temps de fonctionnement et la stabilité en production, et les développeurs pour les nouvelles fonctionnalités livrées, mais elles dressent potentiellement ces groupes les uns contre les autres. Pour le service des opérations, la meilleure protection est que l'équipe de production n'accepte pas les modifications par exemple, et le service Développement est peu enclin à s'intéresser aux problèmes de qualité. Remplacez ces mesures par une responsabilité partagée pour fournir de nouvelles fonctionnalités rapidement et en toute sécurité.

Les responsables de l'organisation doivent, en outre, encourager la collaboration en améliorant la visibilité. La création d'un ensemble d'outils de collaboration communs est essentiel, notamment, lorsque les équipes sont réparties géographiquement et ne peuvent pas se rencontrer physiquement pour collaborer. Fournir à toutes les parties prenantes la visibilité et l'état d'un projet est indispensable pour créer une culture DevOps basée sur la confiance et la collaboration.

La création d'une culture DevOps implique parfois que les individus changent. Il peut être nécessaire de changer de poste ceux qui sont réfractaires au changement — à savoir à l'adoption de la culture DevOps.

L'équipe DevOps

Les arguments pour et contre l'existence d'une équipe DevOps distincte sont aussi vieux que le concept lui-même. Certaines organisations, telles que Netflix, ne disposent pas d'équipes Développement et Opérations ; une seule équipe «NoOps» est responsable de ces deux activités. D'autres organisations ont mis en place avec succès des équipes de liaison DevOps qui résolvent les conflits éventuels et promeuvent la collaboration. Une telle équipe peut être un groupe d'outils ou de processus existants ou bien une nouvelle équipe constituée par des représentants de toutes les équipes ayant un intérêt dans l'application à fournir.

Si vous décidez de créer une équipe DevOps, votre principal objectif vise à la faire fonctionner comme centre d'excellence qui facilite la collaboration sans ajouter de la bureaucratie ou qu'elle ne devienne l'équipe dédiée à la résolution de tous les problèmes DevOps — une évolution qui irait à l'encontre de l'adoption d'une culture DevOps.

Le processus dans DevOps

Dans la section précédente, nous avons vu le rôle des individus et de la culture dans l'adoption de DevOps. Les processus définissent ce que font ces personnes. Votre organisation peut avoir instauré une culture efficace de collaboration, mais si les individus font ce qu'elles font, les bonnes et les mauvaises choses, mais de la mauvaise manière, il existe une probabilité d'échec.

Un grand nombre de processus sont identifiés avec DevOps — ils sont trop nombreux pour être couverts dans ce manuel. Cette section porte sur les principaux processus à la lumière de leur adoption dans l'entreprise.

DevOps comme processus métier

DevOps comme fonctionnalité a un effet sur toute l'entreprise. Elle accroît l'agilité de l'entreprise et améliore la livraison des fonctionnalités aux utilisateurs. Vous pouvez élargir le champ de DevOps en l'apparentant à un *processus métier :* un ensemble d'activités ou de tâches qui produisent un résultat donné (service ou produit) pour les clients.

Dans l'architecture de référence présentée dans le chapitre 2, le processus métier DevOps implique de se saisir des fonctionnalités, de l'idée (généralement identifiée avec les propriétaires du métier) jusqu'à la production en passant par le développement et les tests.

Bien que ce processus ne soit pas suffisamment mature pour être capturé dans un ensemble de flux de processus simples, vous devez capturer les flux de processus que votre organisation utilise déjà pour fournir les fonctionnalités. Ensuite, vous pouvez identifier les domaines d'amélioration en renforçant les processus eux-mêmes et en introduisant l'automatisation (voir la section "Les technologies dans DevOps" dans les pages suivantes de ce chapitre).

Processus de gestion des changements

La *gestion des changements* est un ensemble d'activités qui consistent à contrôler, gérer et suivre les changements en identifiant les produits de travail qui sont susceptibles de changer, et les processus utilisés pour implémenter le changement. Le processus de gestion des changements qu'utilise une organisation est une partie inhérente du flux plus vaste de processus de DevOps. La gestion des changements détermine la manière dont les processus DevOps absorbent et régissent aux demandes de modification et aux retours des clients.

Les organisations qui ont adopté la gestion du cycle de vie d'application (ALM) disposent déjà de processus bien définis et (probablement) automatisés de gestion des changements.

La gestion des changements doit inclure des processus qui offrent les fonctionnalités suivantes :

- Gestion des éléments de travail
- Cycles de vie des éléments de travail configurables
- Gestion de configuration logicielle
- Planification (agile et itérative)
- Contrôle d'accès aux artefacts basé sur les rôles

La gestion traditionnelle des changements tend à se réduire à la gestion des demandes ou des défauts, avec une capacité limitée à suivre les événements entre les demandes de changement ou les défauts et le code ou les exigences associés. Ces approches ne fournissent pas une gestion intégrée des éléments de travail dans le cycle de vie ni une fonctionnalité intégrée de suivi de tous les types de données. Cependant, DevOps implique que toutes les parties prenantes puissent avoir une visibilité et collaborer sur tous les changements dans le cycle de vie du développement logiciel.

Une gestion des changements orientée DevOps ou ALM inclut des processus qui permettent de gérer tous les éléments de travail de tous les projets, tâches et données associées — et pas simplement ceux affectés par une demande de changement ou des défauts. Il inclut également des processus qui permettent à l'entreprise de lier les éléments de travail à tous les artefacts, actifs de projet et autres éléments de travail créés, modifiés, référencés ou supprimés par les utilisateurs qui travaillent dessus. Ces processus fournissent aux membres de l'équipe un accès à base de rôles à toutes les informations de changement et prennent en charge les efforts de développement de projet interactif et agile.

Techniques DevOps

L'adoption de DevOps implique d'inclure un certain nombre de techniques parmi lesquelles :

- Amélioration continue
- Planification des versions
- Intégration continue
- Livraison continue
- Test continus
- Surveillance et retours continus

Ces techniques sont examinées en détail dans la section suivante.

Amélioration continue

Dans une vision Lean, l'adoption du processus n'est pas une action ponctuelle ; il s'agit d'un processus qui évolue en permanence. Une entreprise doit disposer de processus intégrés qui identifient les domaines d'amélioration au fur et à mesure que l'organisation devient plus mature et qu'elle apprend des processus qu'elle a adopté. La plupart des entreprises possèdent des équipes dédiées à l'amélioration des processus, qui s'attachent à améliorer les processus en fonction des observations et des leçons apprises. D'autres permettent aux équipes qui adoptent les processus d'évaluer et de déterminer elles-mêmes leurs propres procédures d'amélioration des processus. Quelque que soit la méthode utilisée, l'objectif vise à instaurer l'amélioration continue.

Planification des versions

La planification des versions est une fonction métier essentielle qui repose sur la nécessité pour l'entreprise d'offrir des fonctionnalités

aux clients et sur les délais de satisfaction de ce besoin. Par conséquent, les entreprises ont besoin de mettre en place des processus bien définis de planification et de gestion des versions qui permettent d'instaurer des feuilles de route de sorties de versions, des plans de projet, des planifications de livraison et une traçabilité de bout en bout dans ces processus.

La plupart des entreprises actuelles exécutent cette tâche en utilisant des feuilles de calcul et en tenant des réunions (souvent très longues) avec toutes les parties prenantes de l'entreprise pour suivre toutes les applications en cours de développement, leur état de développement et leurs plans de sorties. Cependant, des processus bien définis et une gestion automatisée éliminent ces feuilles de calcul et ces réunions et permettent des sorties de versions plus rationnelles et surtout plus prévisibles. En appliquant les pratiques Lean et Agile, les versions publiées sont plus petites et plus fréquentes, et elles permettent d'accorder une plus grande attention à la qualité.

Intégration continue

L'intégration continue (décrite dans le chapitre 2) ajoute une valeur considérable à DevOps en permettant aux grandes équipes de développeurs, travaillant sur des composants multi-technologies sur des sites distribués, de délivrer du logiciel avec agilité. Elle permet de s'assurer que le travail de chaque équipe est intégré en continu à celui des autres équipes de développement, puis validé. Par conséquent, l'intégration continue réduit les risques et identifie les problèmes en amont dans le cycle de vie du développement logiciel.

Déploiement continu

L'intégration continue débouche naturellement sur le déploiement automatisé : un processus d'automatisation du déploiement des logiciels dans les environnements de test, de test système, de pré-production et de production. Bien que certaines organisations s'arrêtent à la production, celles qui adoptent DevOps utilisent généralement le même processus dans tous les environnements pour améliorer l'efficacité et réduire les risques liés à des processus incohérents.

Dans les environnements de test, l'automatisation de la configuration, l'actualisation des données, le déploiement du logiciel dans l'environnement de test, puis l'exécution de tests automatisés accélèrent les cycles de retour des résultats des tests vers l'équipe du développement.

Généralement, l'adoption du déploiement automatisé est la partie la plus importante de l'adoption de DevOps. Pour beaucoup d' utilisateurs de DevOps, DevOps est limité à l'automatisation du déploiement. Du coup, la plupart des outils désignés comme outils DevOps

se limitent uniquement à ce processus. Cependant, comme vous pouvez le noter dans ce manuel, DevOps couvre un champ beaucoup plus large. L'automatisation du déploiement est un composant important de DevOps, mais ce n'est pas le seul composant.

Selon les besoins métier et les défis de votre entreprise, vous pouvez commencer l'adoption de DevOps en utilisant n'importe lequel des processus ou chemins d'adoption décrits dans le chapitre 2.

Tests continus

Les tests continus ont été présentés dans le chapitre 2. D'un point de vue du processus, vous devez adopter des processus dans trois domaines pour mettre en place les tests continus.

- Test de provisionnement et de configuration de l'environnement
- Gestion des données de tests
- Test d'intégration, fonctionnels, de performance et de sécurité

Dans une organisation, les équipes d'assurance-qualité doivent déterminer les processus à adopter pour chacun de ces domaines. Les processus qu'elles adoptent varient en fonction du projet, des besoins particuliers en tests et des exigences sur les contrats de niveaux de service (SLAs). Par exemple, il peut être nécessaire d'exécuter un plus grand nombre de tests de sécurité sur les applications destinées au client que sur les applications internes. Les tests de provisionnement d'environnements et de gestion des données sont les défis les plus importants des projets qui utilisent les méthodologies Agile et l'intégration continue, par rapport aux projets qui suivent une méthodologie Waterfall et exécutent des tests uniquement une fois tous les deux ou trois mois. De même, les exigences de test fonctionnels et de performances pour les applications complexes constituées de composants ayant des cycles de déploiement différents, diffèrent de celles des applications Web monolithiques simples.

Surveillance et retours continus

Les retours des clients revêtent différentes formes : tickets ouverts par le client, demandes formelles de modification, réclamations informelles ou évaluations sur les magasins d'applications (app stores). Du fait du succès des réseaux sociaux et des magasins d'applications (voir le chapitre 5), les entreprises ont besoin de processus bien définis pour traiter les retours d'une multitude de sources différentes et pour les intégrer dans les plans de déploiement des logiciels. Ces processus doivent également être suffisamment agiles pour s'adapter au marché et à l'évolution des réglementations.

Mesure de l'adoption des processus

Vous pouvez mesurer le succès de l'adoption des processus en mesurant si un ensemble de métriques d'efficacité et de qualité s'améliorent au fil du temps. Ces types de métriques impliquent deux conditions :

- Vous devez identifier les métriques d'efficacité et de qualité appropriées. Ces métriques doivent présenter un réel intérêt pour l'entreprise.
- Vous devez définir le niveau de base par rapport auquel vous pourrez constater les améliorations.

Vous pouvez utiliser n'importe lequel des nombreux modèles déjà existants pour mesurer la maturité des processus. Pour les processus DevOps, de nouveaux modèles, tels que IBM DevOps Maturity Model, peuvent évaluer la maturité spécifique de vos organisations à DevOps. Plus d'informations sur le modèle de maturité IBM sont disponibles sur le site `ibm.biz/adoptingdevops`.

Les retours peuvent aussi provenir de la surveillance des données. Ces données sont issues des serveurs sur lesquels s'exécutent l'application (serveurs de développement, de QA ou de production) ou bien d'outils de mesure intégrés à l'application, qui capturent les actions de l'utilisateur.

Une surcharge de données peut se produire. Par conséquent, les entreprises doivent utiliser des processus de capture et d'utilisation des données qui améliorent les performances des applications et les environnements où elles sont exécutées.

Les technologies dans DevOps

Les technologies permettent aux individus de se concentrer sur le travail de création à forte valeur ajoutée tout en délégant les tâches routinières à l'automatisation. Elles donnent également la possibilité aux équipes d'ajuster et d'optimiser le temps et leurs capacités.

Si une organisation crée et gère plusieurs applications, tout ce qu'elle fait doit pouvoir être répété de manière fiable pour garantir la qualité dans chacune des applications. Il ne faut pas tout recommence depuis le début à chaque nouvelle sortie de version ou correction d'erreur des applications. Elle doit réutiliser les actifs, le code et les pratiques pour réduire les coûts et être efficace.

La standardisation de l'automatisation rend également les individus plus efficaces (voir "Les individus dans DevOps" dans les

pages précédentes de ce chapitre). Les organisations peuvent être confrontées au remplacement des employés, des sous-traitants ou des fournisseurs de ressources ; les personnes peuvent changer de projet. Par contre, un ensemble d'outils communs permettra aux utilisateurs de travailler sur n'importe quel projet, et de nouveaux membres d'équipes n'auront à apprendre à maîtriser qu'un seul ensemble d'outils — un processus efficace, économique, réutilisable et évolutif.

Infrastructure programmable

L'*infrastructure programmable* est une fonctionnalité importante de DevOps qui permet aux organisations de gérer l'échelle et la rapidité auxquels les environnements doivent être provisionnés et configurés pour permettre le déploiement en continu.

Autour de la notion d'infrastructure programmable, gravite la notion des *environnements SDE (Software-Defined Environments).* Alors que l'infrastructure programmable est liée à la capture des définitions et des configurations de nœud en tant que code, les environnements SDE utilisent des technologies qui définissent des systèmes entiers constitués de plusieurs nœuds — pas simplement leurs configurations, mais également leurs topologies, rôles, relations, règles de charges de travail et comportement.

Trois types d'outils d'automatisation sont disponibles pour gérer

- **Outils d'applications ou de middleware :** ces outils peuvent généralement gérer comme du code, les serveurs d'applications et les applications qui s'y exécutent. Ces outils spécialisés disposent également de bibliothèques de tâches d'automatisation types pour les technologies qu'ils prennent en charge. Ils ne peuvent pas exécuter les tâches de bas niveau, telles que configurer un système d'exploitation, mais ils peuvent automatiser complètement les tâches de niveau serveur et applicatif.
- **Outils d'environnement de déploiement :** ces outils appartiennent à une nouvelle catégorie d'outils qui peuvent déployer à la fois des configurations d'infrastructure et du code applicatif.
- **Outils génériques :** ces outils ne sont pas spécialisés pour une technologie spécifique et peuvent être scriptés pour exécuter divers types de tâches, de la configuration d'un système d'exploitation sur un nœud virtuel ou physique à la configuration des ports de pare-feu. Ils nécessitent une plus grande quantité de travail initial que les outils d'applications ou middleware, mais ils peuvent traiter un plus grand nombre de tâches.

En utilisant un outil de gestion et de déploiement d'environnement tel qu'IBM UrbanCode Deploy with Patterns, les organisations peuvent concevoir, déployer et réutiliser rapidement des environnements et accélérer le pipeline de distribution.

Pipeline de distribution

Un *pipeline de distribution* est constitué des étapes par lesquelles passe une application, du développement à la production. L'illustration 3-1 montre un ensemble d'étapes types. Ces étapes peuvent varier en fonction de l'organisation, mais également de l'application, selon les besoins, le processus de distribution du logiciel et la maturité de l'entreprise. Le niveau d'automatisation peut également varier : certaines organisations automatisent complètement leurs pipelines de distribution. D'autres soumettent leur logiciel à des vérifications et étapes manuelles du fait d'exigences réglementaires ou propres à la société. Il n'est pas nécessaire de traiter toutes les étapes immédiatement. Commencez par traiter les parties critiques de l'organisation — pas tout d'un coup — puis incluez progressivement chacune des étapes.

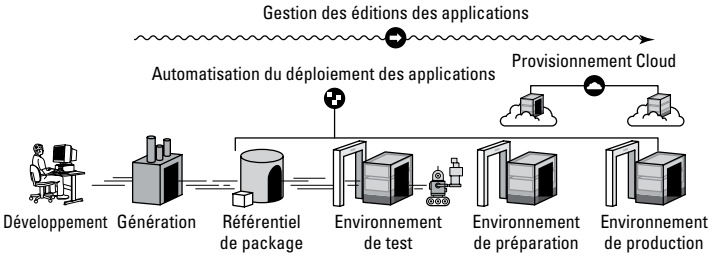


Illustration 3-1 ; étapes d'un pipeline de distribution DevOps type.

Un pipeline de distribution type comporte les étapes décrites dans les sections suivantes :

Environnement de développement

Le développement d'une application se déroule dans un *environnement de développement* qui fournit des outils permettant aux développeurs d'écrire et de tester le code. Mis à part les outils IDE (Integrated Development Environment) que les développeurs utilisent pour écrire le code, cette étape inclut des outils qui permettent le développement collaboratif, tels que des outils de gestion de configuration logicielle, de gestion des éléments de travail, de collaboration, de test unitaires et de planification de projet. Les outils dans cette étape sont généralement des outils multiplateformes et multi-technologies basés sur le type de développement à réaliser.

Etape de génération

L'*étape de génération* est l'étape au cours de laquelle le code est compilé pour créer et tester les fichiers binaires à déployer. Plusieurs outils de génération peuvent être utilisés à ce stade en fonction de besoins multiplateformes et multi-technologies. Les équipes de développement utilisent généralement des serveurs de génération pour faciliter un plus grand nombre de générations nécessaires en permanence dans le cadre d'une intégration continue.

Référentiel de package

Un *référentiel de package* (appelé également *référentiel d'actifs* ou *référentiel d'artefacts*) est un mécanisme de stockage commun pour les fichiers binaires créés pendant la génération. Ces référentiels doivent également contenir les actifs associés aux fichiers binaires pour faciliter leur déploiement, tels que les fichiers de configuration, les fichiers d'infrastructure programmable et les scripts de déploiement.

Environnement de test

Un *environnement de test* est l'emplacement dans lequel les équipes d'assurance qualité, les équipes d'acceptation utilisateur et celles de développement/test exécutent les tests. Divers types d'outils sont utilisés au cours de cette étape, en fonction des besoins de qualité. Voici quelques exemples :

- **Test de gestion de l'environnement :** ces outils facilitent le provisionnement et la configuration des environnements de test. Ils incluent les technologies d'infrastructure programmable et (si l'environnement est dans le Cloud) les outils de provisionnement et de gestion du Cloud.
- **Gestion de la donnée de test :** pour une organisation qui veut mettre en place les tests continus, la gestion des données de test est une fonction essentielle. Le nombre de tests pouvant être exécutés et leur fréquence d'exécution sont limités par le volume de données disponible pour les tests et la vitesse à laquelle les données peuvent être actualisées.
- **Test d'intégration, fonctionnel, de performance et de sécurité :** des outils automatisés sont disponibles pour chacun de ces types de tests. Ces outils doivent être intégrés à un outil ou un référentiel de gestion des actifs de test commun où tous les scénarios de test, scripts de test et résultats associés peuvent être stockés et la traçabilité effectuée vers le code, les exigences et les défauts.
- **Virtualisation des services :** les applications actuelles ne sont pas des applications monolithiques simples. Il s'agit de

systèmes complexes qui dépendent d'autres applications, serveurs d'applications, bases de données et même d'applications et de sources de données tierces. Malheureusement, lors des tests, ces composants peuvent ne pas être disponibles ou peuvent être coûteux. Les solutions de virtualisation de services simulent le comportement — la fonctionnalité et les performances — des composants d'une application pour permettre les tests de bout en bout de l'application dans son ensemble. Ces outils créent des 'bouchons' de simulation (composants virtuels) des applications et services nécessaires à l'exécution des tests. Le comportement et les performances de l'application peuvent être testés lorsqu'elle interagit avec ces parties de code du bouchon de simulation. IBM Rational Test Virtualization Server fournit ces fonctions de virtualisation des tests.

Environnements de promotions et de production

Les applications sont déployées dans les environnements de promotions et de production. Les outils utilisés au cours de ces étapes incluent les outils de gestion et de provisionnement de l'environnement. Les outils pour l'infrastructure programmable jouent également un rôle essentiel dans ces étapes du fait de l'étendue des environnements au cours de ces étapes. Avec l'avènement des technologies de virtualisation et de Cloud, les environnements de promotions et de production peuvent être constitués aujourd'hui de centaines, voire de milliers de serveurs. Les outils de surveillance permettent aux organisations de surveiller les applications déployées dans l'environnement de production.

Automatisation du déploiement et gestion des versions

La gestion de l'automatisation du déploiement des applications d'un environnement de promotion à l'autre implique d'utiliser des outils spécialisés dont certains sont abordés dans les sections suivantes.

Automatisation du déploiement

Les outils d'automatisation du déploiement sont les principaux outils dans l'univers DevOps. Ces outils exécutent l'orchestration des déploiements et identifient la version déployée et le nœud concerné au cours de n'importe quelle étape du pipeline de la génération et du déploiement. Ils peuvent également gérer les configurations d'environnement de toutes les étapes au cours desquelles les composants de l'application doivent être déployés.

Mesure de l'adoption des technologies

Mesurer le retour sur investissement des outils et des technologies est plutôt simple. Généralement, vous pouvez mesurer les améliorations créées par l'automatisation. En outre, les outils automatisés permettent d'améliorer la scalabilité et la fiabilité des tâches de déploiement — ce qui n'est pas toujours possible avec des outils manuels. En fin de compte, l'utilisation d'un ensemble d'outils automatisés intégrés facilite la collaboration et la traçabilité, et améliore la qualité.

Les outils d'automatisation du déploiement gèrent les composants logiciels déployés, les composants middleware et les configurations middleware devant être mise à jour, les composants de base de données qui doivent être modifiés et les modifications de configuration des environnements sur lesquels ces composants doivent être déployés. Ces outils permettent de définir et automatiser également les processus pour exécuter les modifications de ces composants et configurations. IBM UrbanCode Deploy est un outil de déploiement de ce type.

Gestion des versions d'applications

L'orchestration des plans de versions et des déploiements associés nécessite une coordination entre les équipes métier, de développement, d'assurance qualité et enfin celles des opérations. Les outils de gestion et planification de versions permettent aux organisations de planifier et réaliser les sorties de versions, fournissent un portail de collaboration unique pour toutes les parties impliquées dans une version de l'application et permettent la traçabilité d'une version de l'application et de ses composants dans tous les environnements du pipeline de génération et de déploiement d'une application. IBM UrbanCode Release offre ces fonctions de gestion de versions applicatives.

Chapitre 4

Comment le Cloud accélère DevOps

Dans ce chapitre

- Utilisation du Cloud comme facilitateur de DevOps
- Description des déploiements de la pile matérielle/logicielle complète
- Les différents modèles de service Cloud
- Découverte du Cloud hybride

DevOps et le Cloud agissent mutuellement comme deux catalyseurs et facilitateurs. Lorsqu'une entreprise adopte le Cloud, la proposition de valeur d'exploitation du Cloud pour héberger une activité DevOps devient évidente. La flexibilité, la résilience, l'agilité et les services qu'apporte une plateforme Cloud permettent de rationaliser un pipeline de déploiements d'applications hébergé sur le Cloud. Les environnements, du développement aux tests jusqu'à la production, peuvent être provisionnés et configurés selon le besoin et lorsqu'il y en a besoin. Ce processus réduit les goulots d'étranglement relatifs aux environnements dans le processus de déploiement. Les organisations renforcent également l'exploitation des plateformes Cloud pour réduire les coûts des environnements de développement et de tests ou pour fournir une expérience de développement rationalisée moderne pour leurs utilisateurs. Ces éléments constituent des scénarios métiers extrêmement incitatifs pour adopter le Cloud avec et pour DevOps.

Ce chapitre explore les différents modèles de Cloud pour DevOps et examine la proposition de valeur de DevOps comme charge de travail sur le Cloud.

Utilisation du Cloud comme facilitateur pour DevOps

DevOps a pour principal objectif de réduire les goulots d'étranglement dans le pipeline de distribution pour le rendre plus efficace. L'un des principaux goulots d'étranglements auxquels font face les entreprises concerne la disponibilité et la configuration des environnements. Il n'est pas rare de voir les utilisateurs, notamment les développeurs et les testeurs, demander un environnement par le biais d'un système de gestion de tickets traditionnels, la satisfaction de ces demandes pouvant prendre des jours, voire des mois.

DevOps offre l'avantage, entre autres, de pouvoir développer et tester dans un environnement similaire à la production. Au goulot d'étranglement de la disponibilité de l'environnement s'ajoute le risque d'avoir un environnement disponible différent de celui de production. Cette discordance peut se limiter à de simples différences dans la configuration de l'environnement — au niveau du système d'exploitation ou du middleware, ou bien elle peut être beaucoup plus grave avec la présence d'un type de système d'exploitation ou de middleware dans les environnements de développement, totalement différent de celui de l'environnement de production.

L'absence de disponibilité des environnements se traduit par des délais d'attente potentiellement longs pour les utilisateurs. La discordance entre les environnements de développement et de production peut poser des problèmes de qualité significatifs, car les développeurs ne peuvent pas vérifier le comportement de l'application telle qu'elle sera déployée dans l'environnement de production, ou pire, ne peuvent pas valider que le processus de déploiement validés sur les environnements de tests fonctionneront lorsqu'il faudra déployer dans l'environnement de production.

Le Cloud résout ces problèmes de la manière suivante :

- La rapidité du provisionnement d'environnement sur les plateformes Cloud peut fournir aux utilisateurs un accès d'usage en libre-service avec une disponibilité d'environnement et un accès à la demande.
- La possibilité de provisionner et de décomissionner dynamiquement ces environnements en fonction des besoins, améliore la gestion des environnements et diminue les coûts en réduisant la nécessité d'avoir des environnements de tests statiques permanents.

- La possibilité d'exploiter des technologies qui s'appuient sur des "modèles" et qui donnent la possibilité aux organisations de définir et versionner les environnements comme du logiciel, permet de fournir des environnements qui répondent exactement aux besoins des utilisateurs — et qui correspondent surtout à des environnements de type production.
- Du point de vue de l'automatisation, la disponibilité des technologies d'automatisation du déploiement des applications, telles que IBM UrbanCode Deploy, permet avec un seul outil de provisionner l'environnement Cloud et de déployer les versions appropriées vers ces environnement en fonction des besoins et quand cela est nécessaire. Ces technologies peuvent également configurer rapidement l'environnement et l'application pour répondre aux besoins des utilisateurs.
- La disponibilité des technologies de virtualisation de services, telles qu'IBM Rational Test Virtualization Server, fonctionnant conjointement avec les environnements Cloud, permet de simuler les services nécessaires aux tests sans avoir à provisionner des instances réelles de ces services.

L'illustration 4-1 montre comment les environnements Cloud fonctionnent conjointement avec les technologies d'automatisation du déploiement et de virtualisation de services pour fournir des environnements de développement/test de bout en bout.

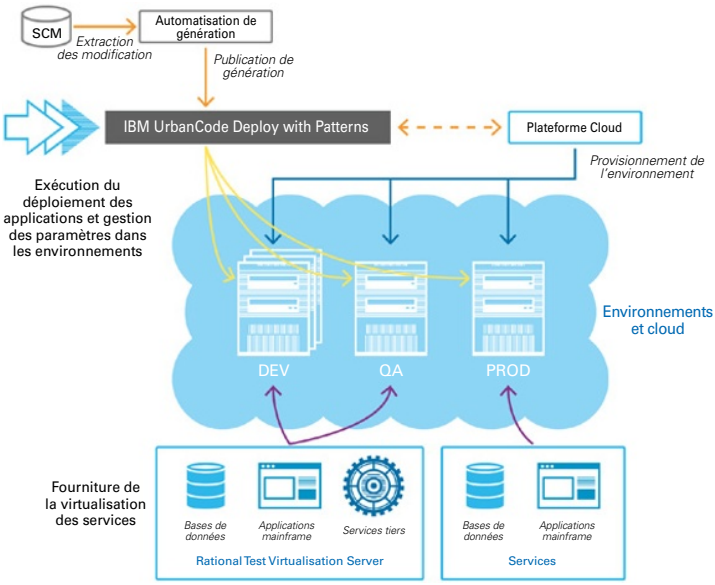


Illustration 4-1 : environnement de développement/test de bout en bout dans le Cloud.

Le Cloud sans DevOps c'est perdre tous les avantages qu'offre le Cloud. L'adoption de DevOps avec des environnements hébergés dans le Cloud permet d'offrir aux organisations qui développent et déploient des applications logicielles l'ensemble des fonctionnalités et des bénéfices qu'elles peuvent tirer du Cloud.

Déploiement de pile matérielle/logicielle complète

Le déploiement d'une application Cloud consiste à déployer l'application et à configurer l'environnement Cloud où elle s'exécute. Ces deux tâches peuvent être exécutées séparément, mais lorsqu'elles sont combinées, cette opération s'appelle le *déploiement de pile complète*. Ces deux approches sont traitées plus en détail dans cette section.

La première approche consiste à séparer le provisionnement d'environnement Cloud du déploiement de l'application. Dans ce cas, il n'existe pas un point unique d'orchestration des environnements Cloud et des applications qui y sont déployées. L'outil d'automatisation du déploiement d'application considère les environnements Cloud comme des environnements statiques. Ce scénario n'optimise pas les avantages du déploiement dans le Cloud.

La seconde approche consiste à utiliser l'outil d'automatisation du déploiement applicatif comme outil d'orchestration unique pour le provisionnement d'environnement Cloud et pour le déploiement des applications dans ces environnements provisionnés. Cela peut se faire en créant des «modèles» qui définissent l'environnement Cloud et sa topologie et qui associent les composants d'application et la configuration aux nœuds définis dans l'environnement Cloud.

Plusieurs technologies de modèles, telles que les modèles IBM Virtual System Patterns et OpenStack Hot, peuvent être utilisées pour définir des modèles d'environnements Clould. Les outils d'automatisation du déploiement, tels qu'IBM UrbanCode Deploy with Patterns, peuvent fournir un provisionnement de pile complète en utilisant ces modèles. Cela inclut le provisionnement de l'environnement Cloud défini dans le modèle et le déploiement de l'application dans ce nouvel environnement. Une fois l'environnement provisionné, des modifications d'application, de configuration et de contenu peuvent être déployées en continu dans l'environnement Cloud sous forme de mises à jour.

Les organisations peuvent également décider de procéder à un déploiement de pile complète lorsque les environnements et les

applications associées sont toujours provisionnés ensemble comme un seul et même actif à déployer. Dans ce cas aucune mise à jour n'est apportée à l'environnement existant.

Choix d'un modèle de service Cloud pour DevOps

Lors de l'adoption du Cloud, vous devez d'abord déterminer l'ensemble des responsabilités que vous souhaitez voir prises en charge par la plateforme Cloud et celles que vous souhaitaient prendre en charge vous-même. Il existe deux principaux modèles de service pour le Cloud : Infrastructure as a Service (IaaS) et Platform as a Service (PaaS).

Iaas

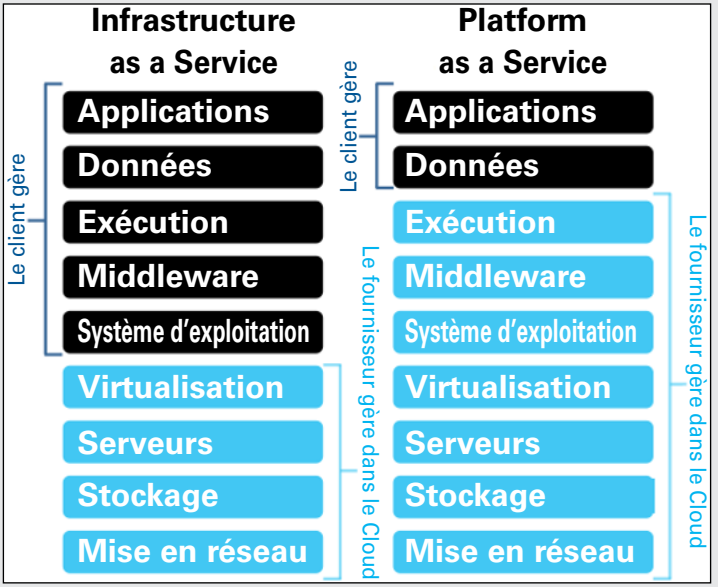
Lors de l'adoption du Cloud dans un modèle IassS, la plateforme Cloud gère l'infrastructure sous-jacente et fournit les fonctionnalités et les services qui permettent de gérer toute l'infrastructure virtualisée. L'installation, les correctifs et la gestion du système d'exploitation, le middleware, les données et l'application restent dans le domaine de responsabilité de l'utilisateur.

Dans le contexte de l'adoption de DevOps dans un contexte de travail dans le Cloud, le choix du modèle de service Cloud à utiliser détermine la manière dont DevOps sera adopté. Pour le modèle de service IaaS, l'organisation d'utilisateurs est chargée de gérer l'ensemble du pipeline de distribution. Tous les outils et intégrations du pipeline de déploiement deviennent la responsabilité des équipes utilisateurs, y compris l'acquisition des outils adéquats et leur intégration pour définir le pipeline de distribution. En outre, ils doivent s'assurer que la collaboration entre les équipes de développement et celles des opérations respecte bien l'approche DevOps. Ce n'est pas parce qu'une plateforme Cloud est utilisée comme service Cloud qu'il n'est plus nécessaire de supprimer les silos de responsabilités entre les développeurs qui fournissent le code et les équipes des opérations qui fournissent l'infrastructure.

Bien que le Cloud apporte une valeur considérable en fournissant une infrastructure IaaS aux équipes de livraison et déploiement des applications, cela n'empêche pas que ces équipes ont toujours besoin de toutes les fonctionnalités de DevOps pour pouvoir fournir la valeur que DevOps peut leur apporter.

Séparation des responsabilités

L'un des principaux problèmes liés à l'exploitation de DevOps sur le Cloud IaaS réside dans la définition de la séparation des responsabilités entre la plateforme Cloud et l'outil de déploiement d'application. Quel outil pour quelle responsabilité ? Une approche simple en la matière consiste à se placer du point de vue des actifs de la pile Cloud selon qu'ils aient besoin d'évoluer rapidement ou lentement. Le tableau ci-dessous montre les différentes couches de la pile matérielle/logicielle, des couches Système d'exploitation, stockage et réseau jusqu'à l'application.



Les couches de configuration d'application, de données et de middleware évoluent rapidement par nature. Elles évoluent souvent, car l'application, ses données et son utilisation se renouvellent sans arrêt. Le changement peut être très rapide pour une application qui est toujours en cours de développement. Les couches inférieures dans ce cas incluent le middleware (serveur d'applications, base de données, etc.), le système d'exploitation et le stockage, et elles ne changent pas si fréquemment. Comme la mise à jour et le réapprovisionnement de toutes les couches pour une simple modification qui affecte uniquement l'application, son contenu et la configuration ne seraient pas une méthode efficace, il est logique de séparer les responsabilités de ces couches à évolutions lente et rapide entre un outil de déploiement d'application et un outil de gestion Cloud. Les couches à évolution rapide sont gérées et automatisées par l'outil de déploiement d'application et les couches à évolution lente le sont par le logiciel de gestion Cloud fourni par la plateforme Cloud.

PaaS

En adoptant un modèle PaaS, vous êtes responsables uniquement de l'application et des données. Toutes les autres fonctionnalités sont fournies par la plateforme PaaS. De ce fait, l'expérience utilisateur s'en trouve nettement améliorée pour les équipes de déploiement d'applications. Les outils de développement et de test d'application sont désormais disponibles comme services sur la plateforme accessible aux utilisateurs. L'organisation distributrice d'applications n'est plus responsable de la gestion du pipeline de distribution. En fait, elle est incorporée dans PaaS et permet aux utilisateurs de se concentrer exclusivement sur la distribution rapide des applications. Tous les outils de développement et de test et le provisionnement d'infrastructure ne sont plus des services incombant aux utilisateurs, ce qui leur permet de se concentrer sur les tâches principales de déploiement des applications.

IBM Bluemix est un PaaS. IBM et ses partenaires gèrent la plateforme et les services qui y sont fournis. La plateforme inclut IBM DevOps Services — un ensemble de services fournissant toutes les fonctionnalités permettant aux équipes d'adopter DevOps, et plus particulièrement un pipeline de déploiement d'applications comme groupe de services. Les équipes de déploiement d'applications peuvent utiliser les services sans se préoccuper de la manière dont ceux-ci sont hébergés et fournis. Les services DevOps dans le Cloud incluent ce qui suit :

- Environnement de développement intégré (Integrated Development Environment) Web comme service
- Fabrication de l'application comme service
- Planification et gestion des tâches comme service
- Analyse de la sécurité comme service
- Déploiement comme service
- Surveillance et analyse comme service

La plateforme fournit également des environnements d'exécution évolutifs pour les applications exécutées dans différents environnements dans le cycle de vie de distribution — du développement, jusqu'aux tests, qualification et la production.

Description d'un Cloud hybride

Le terme Cloud hybride est très employé dans le domaine du Cloud. Il est probablement un peu galvaudé pour décrire des scénarios Cloud où plusieurs technologies Cloud coexistent ou dans lesquels une infrastructure Cloud et une infrastructure physique coexistent. Une méthode simple pour définir un Cloud hybride consiste à examiner cette multitude de scénarios de Cloud.

- **Infrastructure Cloud et infrastructure physique :** scénario de Cloud hybride très courant. Il s'agit du scénario standard, à moins que l'entreprise soit née dans le Cloud. Toutes les organisations ont des charges de travail et des applications qui s'exécutent dans leur infrastructure physique existante. Dans la plupart des cas, certaines de ces applications continuent de fonctionner sur l'infrastructure physique. Les applications mainframe et les systèmes d'enregistrements à forts volumes de données qui ne peuvent être migrés vers le Cloud du fait des contraintes technologiques et de coût sont des exemples types. Même si une organisation migre toutes ses charges de travail vers le Cloud, la migration ne peut pas être exécutée du jour au lendemain, et l'infrastructure physique coexistera avec l'infrastructure Cloud pendant une période plus ou moins longue.
- **Cloud sur site et Cloud hors site :** dans ce scénario, une entreprise peut adopter un Cloud hors site (public ou privé virtuel) pour certaines applications et charges de travail, et un Cloud sur site (privé) pour les autres. Il peut s'agir, par exemple, d'une organisation qui exploiterait un Cloud hors site économique pour gérer ses environnements de développement, et un Cloud sur site autogéré dans son propre centre de données pour toutes les charges de travail de production.
- **IaaS et PaaS :** ce scénario inclut les clients ayant adopté un modèle de service Cloud PaaS pour certaines charges de travail — nouveaux systèmes innovants d'applications de type engagement, par exemple — et IaaS pour un système plus traditionnel de charges de travail d'enregistrement.

Pour l'adoption de DevOps, l'existence d'un cloud hybride présente de nouveaux défis, car dans ce cas, il existe des pipelines de distribution d'applications qui couvrent des environnements de Cloud hybrides et physiques complexes. Exemples d'environnements Cloud hybride :

- Une organisation peut décider d'utiliser un Cloud public pour le développement, les tests et les autres environnements de non-production, et un Cloud sur site, ou même une infrastructure physique, pour la production.
- Une organisation peut disposer d'un système d'applications d'engagement déployé dans un environnement Cloud, alors que les systèmes d'applications d'enregistrements qui fournissent les services back-end pour les principales applications métier peuvent toujours résider dans l'infrastructure physique, par exemple un mainframe.
- Les organisations peuvent tirer parti d'un PaasS public pour l'expérimentation avec les applications innovantes et vouloir les placer dans un Cloud privé lorsqu'une expérimentation aboutit.
- Les organisations peuvent vouloir porter les charges de travail d'application sur plusieurs plateformes Cloud pour ne pas être dépendantes d'un seul fournisseur ou pour permettre de déployer les charges de travail critiques sur des Clouds de plusieurs fournisseurs.

Le déploiement d'applications dans ces divers environnements Cloud et physiques constitue la principale nécessité de l'adoption DevOps dans un Cloud hybride. Les applications, telles qu'IBM UrbanCode Deploy with Patterns, utilisent des modèles pour associer les applications et les configurations à plusieurs environnements, physiques ou Cloud, ce qui permet d'utiliser le déploiement automatisé dans les environnements Cloud hybrides.

Chapitre 5

Utilisation de DevOps pour relever les nouveaux défis

Dans ce chapitre

- Activation des applications mobiles
- Traitement des processus ALM
- Agilité à grande échelle
- Gestion des applications multi-tiers
- Regard sur DevOps dans l'entreprise
- Utilisation des chaînes logistiques
- Navigation sur l'Internet des objets

DevOps est apparu dans les entreprises *nées sur le Web* (entreprises qui ont créé à l'origine leur activité sur Internet), telles qu'Etsy, Flickr et NetFlix. Ces sociétés, tout en relevant des défis technologiques complexes à une très grande échelle, utilisaient une architecture plutôt simple — contrairement aux entreprises qui se sont développées autour de systèmes existants et par le biais d'acquisition et de fusions, avec des systèmes faisant appel à diverses technologies qui devaient fonctionner ensemble. Ces défis sont encore plus considérables du fait des exigences qu'imposent sur les entreprises actuelles les nouvelles technologies comme les mobiles et les modèles de livraisons d'applications telles que les chaînes logistiques logicielles.

Ce chapitre explore certains de ces défis auxquels les entreprises font face et que DevOps peut permettre de surmonter.

Applications mobiles

Dans une entreprise, en règle générale, les applications mobiles ne sont pas des applications autonomes. Leur logique applicative sur l'appareil mobile est très limitée, et ces applications font surtout office d'interfaces avec les applications d'entreprise déjà utilisées dans l'entreprise. Ces applications d'entreprise back-end peuvent être aussi bien des systèmes de traitement de transactions que des portails employés ou encore des systèmes d'acquisition de clients. Le développement et la fourniture d'applications mobiles sont complexes et nécessitent de fournir un ensemble de services dépendants de manière coordonnée, avec toute la fiabilité et l'efficacité nécessaires.

Pour les applications mobiles d'entreprise, les cycles de publication et les publications de nouvelles fonctions doivent être coordonnés avec les applications et services qui interagissent avec elles. Par conséquent, l'adoption de DevOps doit englober impérativement les équipes dédiées aux applications mobiles, ainsi que les autres équipes de développement de logiciels dans l'entreprise.

DevOps et les magasins d'applications

Le déploiement des applications mobiles sur les magasins d'applications est un aspect propre à ces applications. La plupart des applications mobiles ne peuvent pas être déployées directement sur les appareils mobiles ; elles doivent passer par un magasin d'applications géré par le fournisseur. Apple a introduit cette formule de distribution avec son App Store (et verrouillé ses appareils pour empêcher l'installation directe des applications par les développeurs et les fournisseurs d'application). Les fabricants d'appareils, tels que Research In Motion, Google et Microsoft, qui autorisaient auparavant l'installation directe des applications, ont repris le modèle Apple.

Cette situation introduit une étape supplémentaire asynchrone dans le processus de déploiement. Les développeurs ne peuvent plus déployer à la demande les mises à jour d'une application. Même pour les correctifs critiques, les nouvelles versions d'application doivent passer par les processus de soumission et de vérification d'un magasin d'applications. La livraison continue se transforme en soumission et en attente. Le déploiement continu pour le développement et les tests restent disponibles, mais avec un environnement constitué de simulateurs des appareils sur lesquels les applications seront déployées ou des bancs d'appareils physiques.

Quatre-vingt pour cent des données d'entreprise proviennent du mainframe, et soixante-dix pour cent de toutes les transactions touchent un mainframe. L'ouverture d'un chemin d'accès mobile vers ces fonctions de mainframe peut changer la manière de mener l'entreprise et d'approcher les clients, mais l'objectif peut s'avérer compliqué. Vous pouvez être confronté à un manque de compétences, à des silos organisationnels ou à une hétérogénéité de plateformes qui allongent les cycles de livraison, génèrent des retards inutiles et consomment énormément de ressources. Pour fournir un accès mobile à ces applications, les entreprises utilisent DevOps, une approche de distribution de logiciel qui repose sur la rapidité et l'efficacité sans sacrifier la stabilité et la qualité.

Aucun concept ou principe DevOps spécifique ne s'applique uniquement aux applications mobiles. Cependant, les applications mobiles nécessitent d'autant plus de recourir à DevOps que leurs cycles de développement sont courts et leur évolution rapide inhérente;

Processus ALM

ALM (Gestion du cycle de vie des applications) est l'ensemble des processus déployés pour gérer le cycle de vie d'une application, de l'idée (besoin) à la réalisation de l'application qui est déployée et à la fin en maintenance. Ainsi, en considérant DevOps comme une fonctionnalité métier de bout en bout, ALM devient le concept fondamental qui sous-tend le processus DevOps. DevOps élargit le champ ALM pour inclure les responsables métier, les clients et les opérations dans le processus.

Le chemin d'adoption Développement/Test DevOps (voir le chapitre 2 pour plus d'informations) est plus en conformité avec les fonctionnalités ALM traditionnelles de gestion des besoins, des changements, du contrôle de version, de la traçabilité et de gestion des tests. Cependant, les autres fonctionnalités ALM, telles que le suivi et la planification, interviennent dans le chemin d'adoption Pilotage. Les tableaux de bord et les rapports sont quant à eux inclus dans le chemin d'adoption Exploitation.

Extension d'Agile

Le développement selon les principes Lean et Agile sont les points clés de l'approche DevOps — l'un des résultats étant la réduction du gaspillage avec des équipes plus performantes. L'efficacité et la répétition des meilleures pratiques accélèrent les cycles de développement en permettant aux équipes d'être plus innovantes et

plus réactives et donc d'augmenter la valeur client. L'extension des principes d'efficacité et d'agilité (Lean et Agile) au-delà de l'équipe de développement à une équipe d'équipes et à l'ensemble du cycle de vie de livraison des produits et des applications est au cœur de l'approche DevOps.

La plupart des équipes ont déjà adopté l'agilité et veulent étendre leurs processus actuels dans leur adoption à DevOps. En règle générale, de nombreux modèles courants sont disponibles à cet effet. Ces modèles comprennent Scaled Agile Framework (SAFe) et Disciplined Agile Delivery (DAD). Certaines entreprises ont aussi étendu efficacement leur processus Scrum à de très grandes équipes. Ces modèles ont pour vocation de fournir une méthodologie pour adopter l'agilité au niveau de l'entreprise. Cela implique de tenir compte non seulement du développement du code, mais également de l'architecture, du financement du projet et de la gouvernance des processus et des rôles qu'impose la gestion, en appliquant précisément les mêmes principes d'efficacité et d'agilité que ceux qui se sont avérés efficaces au niveau de l'équipe. Quelle que soit l'infrastructure utilisée pour étendre l'agilité, on utilisera les principes de base de l'agilité et on appliquera les meilleures pratiques pour les exploiter afin d'instaurer l'efficacité et l'efficience dans l'entreprise.

Applications multi-tiers

Dans une grande entreprise informatique type, il n'est pas rare de trouver des applications multi-tiers qui couvrent plusieurs plateformes, ayant chacune ses propres besoins de processus de développement, outils et compétences. Généralement, ces systèmes multi-tiers intègrent des applications Web, des ordinateurs et mobiles sur les systèmes frontaux et back-end, telles que les applications du commerce, les systèmes d'entrepôt de données, les applications exécutées sur des mainframes et les systèmes milieu de gamme. La gestion et la coordination des versions des différents composants de ces systèmes multiniveaux, dont la plupart peuvent se trouver sur différentes plateformes, peuvent s'avérer compliquées, même pour l'entreprise informatique la mieux organisée.

Une approche raisonnée consiste à suivre des processus de fabrication, de configuration et de déploiement cohérents automatisés dans toutes les étapes du développement. Ainsi, vous êtes assuré de générer tous les composants — et seulement les composants dont vous avez besoin. Elle garantit également l'intégrité de l'application à mesure que des modifications sont apportées et pendant le cycle de test, de contrôle qualité et de production du projet. IBM UrbanCode

Deploy dispose d'un modèle applicatif qui permet d'automatiser le déploiement complexe des applications multiniveaux.

La gestion d'outils distincts pour différentes équipes basées selon les plateformes est une réalité dans l'environnement multifournisseurs et multiplateformes actuel. Dans ce contexte, des plateformes ouvertes, telles que IBM Jazz permettent d'intégrer des outils hétérogènes pour fournir une solution unifiée. Des pratiques de déploiement cohérentes permettent aux équipes d'exécuter un déploiement fiable et réutilisable sur les plateformes pour générer une réelle valeur métier.

DevOps dans l'entreprise

L'entreprise actuelle dépend de la rapidité à laquelle l'informatique peut fournir le logiciel. Généralement, ces entreprises utilisent des systèmes d'applications d'enregistrement (développées en interne ou applications du commerce) déployées sur des systèmes mainframe ou milieu de gamme. Elles sont confrontées à une multitude de défis :

- Obstacles réguliers
- Complexité des processus
- Exploitation inadéquate des compétences
- Silos organisationnels
- Plateformes et outils qui allongent les cycles de livraisons de versions, génèrent des retards inutiles et gaspillent les ressources

DevOps au niveau de l'entreprise permet aux parties prenantes de la planification, du développement, des tests et des opérations de distribuer en continu le logiciel dans l'entreprise. Actuellement, les entreprises déploient des applications qui sont réellement interplateformes, qu'il s'agisse d'appareils mobiles ou de mainframes. L'approche DevOps du développement utilise les principes d'efficacité pour créer un pipeline de distribution efficient et efficace qui permet de déployer, tester et distribuer les applications, car il améliore la qualité et accélère le développement tout en réduisant ses coûts.

Compte tenu de la vraie nature multiplateforme des entreprises actuelles, avec l'existence d'applications mobiles, de Cloud, réparties et mainframe — qui doivent être toutes créées, intégrées, déployées et exploitées, l'efficacité, la rationalisation et la collaboration qu'apporte DevOps deviennent un véritable différenciateur concurrentiel essentiel.

Chaîne logistique

Face au recours croissant à l'externalisation et aux partenariats stratégiques pour apporter à l'entreprise les compétentes et capacités dont elle a besoin, les chaînes logistiques logicielles deviennent la norme. Une *chaîne logistique* est un système d'organisations, de personnes, de technologies, d'activités, d'informations et de ressources intervenant dans le processus de livraison d'un produit ou d'un service d'un fournisseur chez un client : Les divers fournisseurs dans la chaîne peuvent être internes ou externes à l'entreprise.

Dans une entreprise qui a opté pour un modèle de chaîne logistique pour la fourniture de ses logiciels, l'adoption de DevOps peut relever du défi, car les relations entre les fournisseurs sont plus gérées par des contrats et des accords sur les niveaux des services que par la collaboration et la communication. Néanmoins, une telle entreprise peut toujours adopter DevOps. Les principales équipes du projet conservent la propriété des fonctionnalités de planification et de mesure, les autres fonctionnalités étant partagées entre les autres fournisseurs. Dans le pipeline de distribution, différents fournisseurs peuvent détenir différents morceaux du pipeline. L'utilisation d'un ensemble d'outils communs et d'un référentiel d'actifs communs est donc essentielle. Un outil de gestion des éléments de travail, par exemple, fournit des fonctions de génération de rapports sur tous les éléments à la charge de chacun des fournisseurs, et permet de transférer la propriété des demandes de travail aux fournisseurs. L'utilisation d'un référentiel d'actifs communs fournit un mécanisme pour transmettre les actifs dans le pipeline pour permettre la livraison en continu.

Internet des objets

L'étape suivante importante pour DevOps est son évolution dans le domaine des systèmes et des appareils embarqués, où elle est généralement appelée *ingénierie continue*. Au début d'Internet, la plupart des données qui y étaient partagées étaient générées par des individus. Aujourd'hui, un nombre incalculable d'appareils connectés à Internet (capteurs et actionneurs, par exemple) génèrent largement plus de données que l'être humain. Ce réseau de périphériques interconnectés sur Internet s'appelle l'*Internet des objets*.

Dans cet espace, DevOps est potentiellement encore plus essentiel du fait de la coexistence du matériel et du logiciel incorporé qui s'y exécute. Les principes de DevOps sont reflétés dans l'ingénierie continue pour garantir que le logiciel embarqué dans les appareils

est de très haute qualité avec les spécifications techniques appropriées.

Le service «Opérations» dans l'ingénierie continue est remplacé par des techniciens matériel ou système qui conçoivent des matériels personnalisés pour les périphériques. La collaboration entre les équipes de développement et de test et les ingénieurs système est cruciale pour développer et fournir des matériels et des logiciels d'une manière coordonnée, bien que le développement des matériels et des logiciels doive suivre des cycles de livraison différents. Les besoins en développement et en test pour la livraison continue restent les mêmes. Des simulateurs sont utilisés pour tester le logiciel et le matériel pendant le développement.

Anti-modèles

Dans la pratique, il existe toujours des limitations à l'adoption des principes DevOps. Certaines de ces limitations sont des fonctions des secteurs et des environnements où évoluent une entreprise, telles que le respect des réglementations, les systèmes matériels complexes ou les capacités de distribution de logiciels immatures. Dans ce cas, DevOps doit être adopté à la lumière des *anti-modèles* (modèles inefficaces et contreproductifs) qui peuvent ne pas être acceptables pour une entreprise, en fonction de ses besoins métier.

Water-SCRUM-fall

Forrester (`www.forrester.com`), une société globale de conseil et d'études, utilise le terme *Water-SCRUM-fall* pour décrire l'état actuel d'adoption des méthodologies agiles de développement de logiciels. D'un point de vue DevOps, cela implique que, bien que les équipes de développement puissent avoir adopté des pratiques agiles, les équipes à la périphérie peuvent toujours utiliser des processus de type Waterfall qui ne permettent pas la livraison continue. Cette situation s'explique par la culture d'entreprise dans plusieurs entreprises. Une entreprise qui adopte DevOps doit intégrer des processus manuels dans des pratiques DevOps plus larges.

NoOps

Dans une organisation NoOps, le service Opérations est éliminé, et ses responsabilités sont fusionnées avec celle du service Développement. Netflix, un opérateur de télévision Web, prône cette méthode. NoOps peut bien fonctionner pour certaines entreprises, mais il convient d'attendre afin de déterminer si ce modèle organisationnel sera plébiscité.

Chapitre 6

Réussir DevOps : témoignage d'IBM

Dans ce chapitre

- Description des meilleures pratiques pour les dirigeants
- Organisation de l'équipe
- Identification des objectifs DevOps
- Noter la transformation DevOps
- Apprendre à partir des résultats DevOps

DevOps a été adopté dans l'ensemble de la société IBM et évolue régulièrement. Cette adoption résulte de la mise en place réussie d'une démarche DevOps initiée par IBM Software Group (SWG) Rational et qui est maintenant utilisée, entre autres, dans les divisions Watson, Tivoli, Global Business Services. Ce chapitre propose une étude de cas de l'adoption des fonctionnalités DevOps par l'équipe produit IBM Rational Collaborative Lifecycle Management d'IBM SWG.

Cette volonté d'améliorer la livraison du produit IBM Rational Collaborative Lifecycle Management a ceci d'unique qu' elle a été menée de manière *ouverte* — l'équipe de livraison du produit délivre tous ses artefacts de développement et le résultat de son travail en cours, y compris tous les éléments de travail détaillés, de manière transparente sur `jazz.net`. Ce site Web est accessible au public, et les utilisateurs enregistrés peuvent consulter le travail planifié, le travail en cours et l'historique de tout le travail de développement réalisé pour les produits d'IBM Rational Collaborative Lifecycle Management.

Le rôle du management

La culture est un fil invisible dans une organisation. Elle repose sur des valeurs et des comportements que les managers et les employés font tous évoluer. Bien souvent, vous ne comprenez pas la culture d'une organisation tant que vous n'êtes pas impliqué dans un projet de transformation significative. Il y aura toujours les sceptiques qui attendent afin de voir s'il ne s'agit pas simplement de l'engouement du mois. Des leaders de la transformation émergeront. Il est donc important d'établir une approche pour comprendre ces dynamiques et identifier les rôles de chacun de manière à bien identifier les vrais inhibiteurs.

Pour traiter la dynamique culturelle, les responsables d'IBM SWG ont adopté diverses approches :

- **Sélectionner le leader approprié :** le leader a pour rôle de rassembler les différents points de vue afin d'orienter l'équipe vers un ensemble commun d'objectifs, d'inhibiteurs, de changements de processus et de décisions sur les premières étapes à mettre en œuvre.
- **Impliquer les parties prenantes :** tous ces changements doivent absolument être soutenus par la direction, les managers et chaque contributeur individuel des différentes disciplines du développement. Il faut nommer et impliquer des leaders et experts du changement parmi les représentants métier, les architectes, les développeurs, les testeurs et les opérations.
- **Mesurer les améliorations et les résultats :** il est essentiel de disposer d'un ensemble de mesures qui permettent de dégager l' efficacité et les résultats économiques de ces changements. Ces objectifs et mesures doivent placer la barre assez haut et rendre les personnes responsables en évitant le risque d' un désengagement.
- **Créer une dynamique avec les premiers succès :** la compréhension des inefficacités existantes et la mesure des améliorations apportées par DevOps dans chaque domaine créent une dynamique du changement.
- **Communiquer et écouter :** en tant que leader, il est important de comprendre la dynamique réelle de l'instauration du changement dans l'équipe. Le échanges 1 à 1 avec les individus et les interactions régulières en face à face entre les équipes techniques, les responsables et les leaders métier permettent d'évaluer l'adhésion de l'équipe, sa perception des inhibiteurs, et, tout aussi important, offrent une opportunité au management de partager ses perspectives sur les priorités et l'avancement.

Si vous êtes dirigeant d'entreprise, vous devez encourager les équipes et vous rendre disponible pour comprendre et éliminer les obstacles. Evoluer dans une équipe soudée avec des objectifs métier clairs est indispensable pour amener tout le monde à emprunter le même chemin.

Constitution de l'équipe

L'équipe produit IBM SWG Rational Collaborative Lifecycle Management fait partie d'une entité plus large qui développe des outils de développement de logiciels dans les domaines de la planification, la livraison des logiciels, la gestion de la qualité des logiciels et la surveillance et l'analyse des applications.

Cette équipe produit d'IBM SWG est une grande équipe organisée à l'échelle mondiale en quatre équipes produit sur 25 sites de 10 pays. Avant d'adopter l'approche DevOps, l'activité du groupe reposait sur une planification de sorties produit annuelles, y compris une période de trois à six mois pour déterminer ce qu'il fallait mettre dans la sortie de la version annuelle.

Définition des objectifs DevOps

L'équipe IBM SWG considéra qu'elle ne réagissait pas suffisamment rapidement à l'évolution du marché et des besoins des clients. Elle décida de raccourcir le cycle de sorties de versions, non seulement dans les phases de développement et de test, mais également pour la collaboration et les interactions avec les parties prenantes de l'entreprise et les clients. La décision fut prise de passer d'une planification annuelle à une planification trimestrielle.

Outre la nécessité d'accélérer son développement pour fournir de nouvelles fonctionnalités plus fréquemment, l'équipe devait également s'engager plus rapidement pour prendre en charge les modèles de distribution Cloud, le développement et les tests d'applications mobiles et d'autres fonctionnalités afin de prendre en compte les évolutions technologiques. L'équipe décida donc d'adopter les principes et pratiques de DevOps pour changer le développement des logiciels afin de livrer plus rapidement et plus fréquemment de la valeur pour ses clients.

Une modification d'une telle ampleur imposait un changement culturel dans l'organisation. Il fut donc décidé de créer quatre groupes de travail constitués de membres à la fois du management et de responsables techniques. Ces groupes de travail examinèrent complètement

les processus de sortie des logiciels et décidèrent de changer les méthodes de travail. Un ensemble de mesures et plans d'action fut mis en place pour traiter les principaux points du processus de développement. Une équipe d'experts en livraison continue fut créée, et une personne fut chargée de former les équipes et de communiquer les meilleures pratiques à l'ensemble de l'organisation.

L'adoption de DevOps par l'équipe IBM SWG commença en identifiant ces objectifs :

- Rationaliser le processus et introduire les nouvelles méthodologies.
- Améliorer l'usage des outils en termes de cohérence, de mise à l'échelle sur d'autres équipes, de traçabilité et de mesures
- Faire évoluer la culture vers l'amélioration permanente

Apprendre de la transformation DevOps

Cette section explique les étapes suivies par l'équipe d'IBM SWG pour faciliter sa transformation DevOps.

Etendre les pratiques Agile

Les pratiques Agile existantes furent étendues au-delà du développement et des tests pour inclure les clients, les représentants métier et les opérations afin d'éliminer les silos et d'améliorer les résultats. Ce modèle Agile plus large a permis aux équipes de collaborer d'apporter plus de cohérence et de qualité aux produits logiciels délivrés, d'apporter plus de valeur métier à l'entreprise, tout cela en utilisant un ensemble de processus qui s'intégraient à chaque étape de la chaîne de développement du produit.

La direction produit, la partie conception et les équipes de développement furent regroupés au sein d'une seule et même équipe. L'équipe de développement est constituée des rôles traditionnels de responsables de développement et de responsables d'équipes, mais également de la gestion des opérations et d' architectes pour prendre en charge la stratégie de cycle de vie de bout en bout.

Des ressources dédiées furent affectées à l'encadrement des équipes pour couvrir les livraisons agiles et continues à l'ensemble de l'organisation. La priorité fut donnée aux fonctionnalités par rapport aux composants du produit, ce qui a permis d'éliminer les silos traditionnels et d'assurer la disponibilité et l'automatisation d'un produit

livrable à chaque Sprint. En outre, ces équipes fonctionnelles furent renforcées par la désignation de responsables de développement dédiés. Des réunions Scrum furent organisées régulièrement au niveau de l'organisation du développement afin d'identifier et d'éliminer les blocages, suivre les principales mesures, utiliser des données de tableau de bord dynamiques et communiquer les informations importantes.

Afin de mieux appréhender les opportunités et évolutions du marché et de mieux prioriser les développements, un comité produit stratégique fut créé, constitué de la direction produit, des directeurs de développement, d'architectes et de responsables métier. Leurs responsabilités sont les suivantes :

- Allocation et financement du succès de l'exécution du programme
- Gestion, assistance et soutien à l'exécution du programme
- Etablissement d'une vision et d'une orientation à long terme pour le métier
- Priorisation des ensembles de *user stories* délivrées dans les versions annuelles pour s'aligner sur la vision à long terme.

Exploiter l"automatisation des tests

Pour éliminer les longs cycles de test back-end traditionnels et améliorer la qualité des éditions, une approche de test continu agile fut adoptée en utilisant l'automatisation et la virtualisation. Il fut décidé de mettre en place des itérations de quatre semaines se terminant par une démo, et des jalons de quatre semaines prenant fin avec une version livrée utilisable par le client. Les rétrospectives après chaque jalon et l'analyse des problèmes techniques ont permis d'éliminer les écueils dans les itérations futures. La devise de l'équipe d'IBM SWG était «Tester en amont et souvent».

L'équipe a adopté les meilleures pratiques suivantes pour l'automatisation des tests :

- Automatiser les tests répétitifs et nécessitant une charge de travail importante
- Automatiser les tests sur les parties fréquemment affectées par des dysfonctionnements
- Exécuter les tests automatisés à chaque génération de l'application : exécuter tôt et souvent

- Créer une automatisation des tests qui ne soit pas affectée par les évolutions de l'interface utilisateur — utilisation d'une infrastructure qui sépare l'interface utilisateur des tests
- Faciliter la création, la livraison et la gestion des tests automatisés de manière à assurer de leur appropriation par l'équipe chargée des fonctionnalités produit.
- Planifier et prendre en compte dans les estimations le travail de développement de l'automatisation des tests , de manière à s'assurer que les développeurs disposent du temps nécessaire pour le réaliser.
- Développer des mesures pour s'assurer que l'automatisation est utile (vous ne pouvez pas améliorer ce que vous ne mesurez pas)
- Evaluer constamment si l'automatisation détecte les dysfonctionnements et reprendre l'écriture des tests automatisés si elle n'en trouve pas.

Pour prendre en charge l'automatisation des tests, l'équipe a déployé IBM Rational Test Workbench pour les tests fonctionnels et les tests de performances. Pour exécuter les tests plus fréquemment, l'automatisation du déploiement des versions intermédiaires de l'application était essentielle. En utilisant IBM UrbanCode Deploy, l'équipe a réduit de 90 % les coûts de déploiement par le biais du déploiement automatique de l'application fabriquée couvrant également l'automatisation des paramètres nécessaires d'application et de configuration de serveur de base de données.

Créer un pipeline de livraison

L'équipe IBM SWG décida de créer un pipeline de distribution améliorant l'usage des outils comme 'services' afin de permettre aux développeurs de valider le code, de le tester et de le déployer dans un environnement de production en quelques 60 minutes au lieu d'un ou deux jours auparavant. Ce processus a permis de réduire les retours en arrière et d'optimiser la productivité.

L'équipe de déploiement détermina qu'un pipeline de distribution continue nécessitait d'appliquer les meilleures pratiques suivantes :

- Exécuter les tests au plus tôt (« Shift-left ») et automatiser au maximum
- Utiliser le même mécanisme de déploiement sur tous les environnements

- S'efforcer de maintenir constamment l'application en d'être livrée aux utilisateurs
- Traiter l'infrastructure comme une infrastructure programmable

L'illustration 6-1 montre les produits et les fonctions fournis dans le cadre du pipeline de livraison continue adopté par IBM SWG.

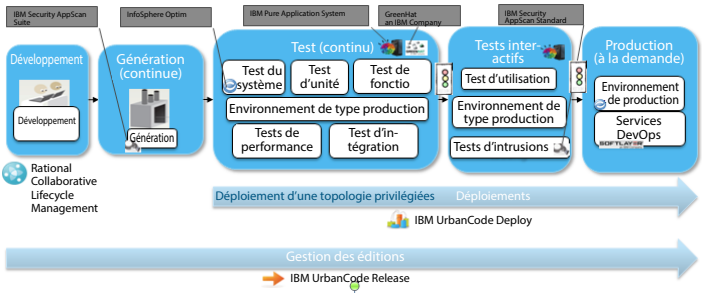


Illustration 6-1 : le pipeline de distribution continue.

Une meilleure pratique essentielle de l'implémentation d'un pipeline de livraison continue est de "traiter l'infrastructure comme une infrastructure programmable". Cela implique que le développeur peut écrire des scripts pour configurer l'infrastructure requise pour l'application dans le code de cette dernière. Auparavant, cela était réalisé généralement par un administrateur système ou une personne chargée des opérations, mais désormais avec le contrôle et les gains d'efficacité que cela apporte, le développeur peut le faire directement. Puppet, Chef et IBM UrbanCode Deploy with Patterns sont des exemples des nouvelles catégories d'outils d'automatisation qui font de l'infrastructure programmable une réalité concrète.

Désormais, l'équipe d'IBM SWG traite son infrastructure comme une infrastructure programmable et suit ces meilleures pratiques :

- Traiter les définitions de modèles de déploiement, les packages de scripts et les services comme du code
- Versionner tout
- Automatiser le déploiement des modèles de topologie vers le Cloud
- Gérer les versions de modèles dans plusieurs environnements Cloud
- Automatiser les tests des modèles
- Nettoyer les ressources de catalogue pour éviter la propagation

Expérimenter rapidement

Le concept de livraison continue couvre non seulement les activités de développement, telles que l'intégration continue et le déploiement continu, mais également l'activité plus fondamentale d'apprentissage qui peut être réalisée de manière optimale via l'expérimentation et la mesure de résultats.

Lorsque des fonctionnalités ou des services sont ajoutés à une application, vous n'êtes pas sûr que le client en tirera les avantages escomptés. C'est la raison pour laquelle IBM considère qu'il est important d'expérimenter en amont et souvent, de demander aux clients des retours sur les fonctionnalités qui leur conviennent et de supprimer celles qui n'offrent que peu d'avantages ou qui peuvent peut-être même présenter un problème. Cette stratégie est décrite dans le diagramme de l'illustration 6-2.

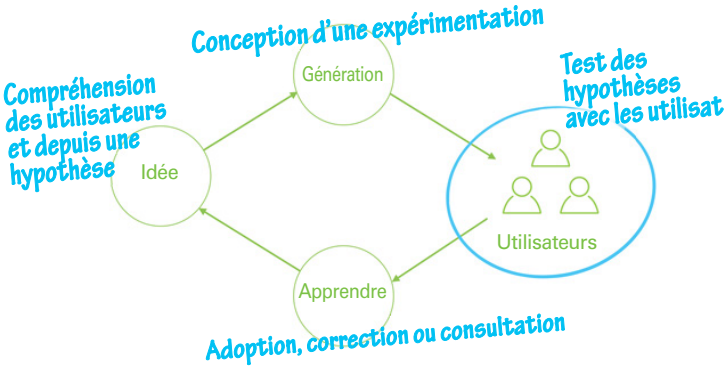


Illustration 6-2 : le développement reposant sur l'hypothèse.

L'équipe d'IBM SWG s'est enrichi de l'expérimentation fréquente et a développé les meilleures pratiques suivantes :

- Définir des mesures et des critères de succès/d'échec
- Déterminer ce qui fonctionne par expérimentation— des petits tests par un petit groupe d'utilisateurs afin de déterminer l'utilité d'une fonctionnalité
- Lancer en continu plusieurs expérimentations
- Prendre rapidement des décisions à partir de faits
- Distribuer rapidement pour expérimenter plus rapidement
- Créer un mécanisme pour permettre l'expérimentation au niveau du système (Google Analytics, IBM Digital Analytics, etc;)

- Envisager différents modèles d'expérimentation (tests A/B classiques, MAB Multi-Armed Bandit, etc).
- Suivre deux chemins simultanément pour les projets associés : expérimenter sur un projet Cloud et utiliser les données des expérimentations pour non seulement déterminer la direction du projet sur le Cloud, mais également celle des projets associé sur site.

Améliorer en continu

IBM SWG voulait créer une culture d'amélioration continue et exploiter des mesures d'efficacité et d'efficience pour s'en assurer. Les équipes gèrent leurs efforts d'amélioration continue comme un projet Agile. Dans ce contexte, l'amélioration continue repose sur le suivi des objectifs de maturité, des difficultés et des actions d'amélioration associées pour résoudre les problèmes. Les équipes suivent le travail d'amélioration continue comme n'importe quel autre travail de développement pour que l'investissement soit largement partagé par tous. Le développement et l'adoption de ces objectifs de maturité (par exemple, les capacités d'amélioration) peuvent prendre un ou plusieurs trimestres. La réduction ou l'élimination des problèmes importants peuvent prendre de nombreux mois. Mais dans tous les cas, les actions d'amélioration spécifiques doivent être toutes adaptées pour pouvoir être réalisées dans le délai d'un mois.

IBM SWG utilise les rétrospectives pour institutionnaliser l'amélioration continue. Une *rétrospective* est la vérification régulière de ce qui a fonctionné, de ce qui n'a pas fonctionné et des actions qui doivent être menées pour procéder à des améliorations. Si vous n'utilisez pas de rétrospectives, cela implique qu'un niveau de perfection dans le développement de logiciel n'a pas encore été atteint. Dans les grandes équipes, il peut exister une hiérarchie de rétrospectives. Pour IBM SWG, chaque équipe chargée d'un composant exécute une rétrospective, et ces rétrospectives sont prises en compte dans les rétrospectives du niveau de l'application qui sont ensuite elles-mêmes prises en compte dans les rétrospectives de niveau produit. Les actions des rétrospectives sont documentées avec les difficultés et les actions d'amélioration continue à exécuter pour les réduire ou les éliminer.

Et pour s'assurer que les équipes travaillent plus efficacement, l'équipe d'IBM SWG a défini des mesures métier et des mesures opérationnelles afin de déterminer l'efficacité de la transformation DevOps. Les mesures métier sont constituées d'améliorations mesurées dans les domaines suivants :

- Accélération de la livraison
- Amélioration de la satisfaction du client
- Réduction des coûts de maintenance tout en augmentant les investissements dans l'innovation
- Accroissement de l'adoption client

Les mesures opérationnelles ont un impact sur l'efficacité de l'équipe dans le temps et mesurent les éléments suivants :

- Délai de lancement d'un nouveau projet
- Délai de fabrication
- Délai des tests d'itération

Les résultats de DevOps

Une approche DevOps a permis à l'équipe d'IBM SWG d'obtenir des gains significatifs au niveau de l'amélioration de la satisfaction client et d'une meilleure adoption client et, au final, de connaître une croissance à deux chiffres de son chiffre d'affaires. Les délais plus courts ont stimulé les équipes de livraison au sein d'IBM, ce qui s'est traduit par des livraisons plus rapides des mises à jours des solutions sur site, ainsi que la livraison de nouveaux services Cloud, tels Bluemix, DevOps Services for Bluemix ou Collaborative Lifecycle Management as a Managed Service (CLM aaMS).

L'illustration 6-3, un exemple du succès de l'approche DevOps chez IBM, montre les résultats mesurés atteints par l'équipe produit IBM SWG Rational Collaborative Lifecycle Management.

Mesures de cycle de vie	2008	2010	2012-2014	Amélioration totale
Lancement du projet	30 jours	10 jours	2 jours	28 jours
Affinage du backlog	90 jours	45 jours	En cours	89 jours
Délai total vers le développement	120 jours	55 jours	3 jours	117 jours
Délai de génération composite	36 heures	12 heures	5 heures	700 %
Haute disponibilité BVT	S/O	18 heures	<1 heure	172 heures
Durée des tests d'itérations	5 jours	2 jours	14 heures	4 jours
Durée total du déploiement	2 jours	8 heures	4 heures-> 20 minutes	2 jours
Délai total vers la production	9 jours	3 jours	2 jours	7 jours
Délai entre les éditions	12 mois	12 mois	3 mois	9 mois

Illustration 6-3 : Améliorations mesurées de l'équipe IBM SWG.

Chapitre 7

Les dix mythes de DevOps

Dans ce chapitre

- A quoi sert DevOps
- A quoi ne sert pas DevOps

DevOps est un mouvement récent qui continue d'émerger, notamment dans les entreprises. A l'instar de n'importe quel mouvement ou n'importe quelle tendance, il fait naître des mythes et des illusions. Certains de ces mythes peuvent provenir de sociétés ou de projets qui ont tenté d'adopter DevOps et qui ont échoué. Ce qui peut est vrai dans un cas, peut ne pas être nécessairement vrai dans un autre. Voici quelques-uns des mythes courants sur DevOps — et les faits.

DevOps s'adresse uniquement aux entreprises «nées sur le Web».

Ce que l'on appelle généralement DevOps, provient des entreprises «nées sur le Web (sociétés qui proviennent d'Internet), telles que Etsy, Netfix et Flickr. Cependant, les grandes entreprises utilisent des principes et des pratiques DevOps pour distribuer les logiciels depuis des dizaines d'années. En outre, les principes DevOps actuels, décrits dans ce manuel, se caractérisent par un niveau de maturité qui permet de les appliquer aux grandes entreprises disposant de technologies multiplateformes et d'équipes réparties.

DevOps vise apprendre aux équipes des opérations à programmer

Les équipes d'opérations écrivent depuis toujours des scripts pour gérer les environnements et les tâches répétitives, mais avec l'évolution de l'infrastructure programmable, ces équipes ont eu besoin de

gérer ces énormes volumes de code avec les pratiques de l'ingénierie logicielle, telles que la gestion de version de code, le check-in, le check-out, le développement en parallèle et la fusion. Actuellement, une nouvelle version d'un environnement est générée par un membre d'une équipe d'opérations en créant une nouvelle version du code qui la définit. Cependant, cela n'implique pas que les équipes des opérations doivent apprendre à coder en Java ou C#. La plupart des méthodologies d'infrastructure programmable utilisent des langages tels que Ruby qui sont relativement simples à maîtriser, notamment pour les personnes qui savent créer des scripts.

DevOps s'adresse uniquement aux équipes de développement et des opérations

Bien que le terme laisse penser qu'il concerne le développement et les opérations, DevOps s'adresse à toute l'équipe. Toutes les parties prenantes dans la livraison de logiciel — les métiers, les utilisateurs, le management, les partenaires, les fournisseurs, etc. — sont concernées par DevOps.

DevOps n'est pas fait pour les entreprises ITIL

Certaines personnes craignent que les fonctionnalités DevOps, telles que la livraison continue, ne soient pas compatibles avec les vérifications et les processus prescrits par ITIL (Information Technology Infrastructure Library), un ensemble de meilleures pratiques pour la gestion des services informatiques. En fait, le modèle de cycle de vie ITIL est compatible avec DevOps. La majorité des principes définis par ITIL sont parfaitement compatibles avec les principes DevOps. Cependant, ITIL pâtit d'une mauvaise réputation dans certaines organisations du fait de son implémentation avec des processus Waterfall lents qui ne permettent pas d'effectuer rapidement des modifications et des améliorations. L'alignement des pratiques entre les équipes de développement et des opérations est l'essence même de DevOps.

DevOps n'est pas adapté aux secteurs réglementés

Les secteurs réglementés ont un besoin global de vérifications et protection et s'assurer de l'approbation des parties prenantes chargées de la conformité et de l'audit. L'adoption de DevOps améliore la conformité s'il est appliqué correctement. L'automatisation des flux de processus ainsi que l'utilisation d'outils disposant de fonctionnalités intégrées pour enregistrer les informations d'audit peuvent être bénéfiques.

Les organisations dans les secteurs réglementés auront toujours des points de contrôle ou points de passage nécessitant des interventions manuelles, mais ces éléments ne sont pas incompatibles avec DevOps.

DevOps n'est pas fait pour les développements externalisés

Les équipes externalisées doivent être considérées comme des prestataires ou des fournisseurs de fonctionnalités alimentant le pipeline de distribution DevOps. Cependant, les entreprises doivent s'assurer que les pratiques et les processus des équipes des fournisseurs sont compatibles avec ceux de leurs équipes de projet internes.

L'utilisation d'outils communs de planification de versions, de gestion des éléments de travail et de référentiel d'actifs améliore de manière significative la communication et la collaboration entre les équipes métier, les fournisseurs et les équipes projet, et elle permet d'appliquer les pratiques DevOps. L'utilisation d'outils de gestion de versions d'applications peut améliorer considérablement la capacité d'une organisation à définir et coordonner l'ensemble du processus de sorties de versions entre tous les participants.

Pas de DevOps sans Cloud

Quand on pense à DevOps, on pense souvent au Cloud du fait de sa capacité à provisionner dynamiquement les ressources d'infrastructure pour les développeurs et les testeurs, ce qui permet d'obtenir rapidement des environnements de test sans attendre des jours/semaines la réponse positive à une demande manuelle. Cependant,

le Cloud n'est pas nécessaire pour adopter les pratiques DevOps dès lors qu'une organisation dispose de processus efficaces pour provisionner les ressources nécessaires aux déploiements et tests des nouvelles modifications de l'application.

La virtualisation est facultative. Le déploiement continu vers des serveurs physiques est possible si les serveurs peuvent être configurés et déployés suffisamment rapidement pour satisfaire les équipes.

DevOps ne s'applique pas aux grands systèmes complexes

Les systèmes complexes ont besoin de la discipline et collaboration qu'apporte DevOps. Ces systèmes se caractérisent généralement par des composants logiciels et/ou matériels ayant chacun leurs propres cycles et jalons. DevOps facilite la coordination de ces cycles de distribution et la planification des versions au niveau système.

DevOps est uniquement une affaire de communication

Certains membres de la communauté DevOps ont inventé des termes humoristiques, tels que *ChatOps* (les équipes effectuent toutes les communications via des outils de communication, tels qu'Internet Relay Chat) et *HugOps* (DevOps se concentre uniquement sur la collaboration et la communication). L'origine de ces termes découle, à tort, que la communication et la collaboration peuvent résoudre tous les problèmes.

Certes la communication est un élément important de DevOps, mais une meilleure communication combinée à des processus sources de gaspillage n'améliore pas les déploiements

DevOps est synonyme de déploiement continu des modifications

Cette vision erronée provient des organisations qui déploient uniquement des applications. Certaines de ces entreprises indiquent fièrement sur leurs sites Web qu'elles déploient tous les jours en

production. Cependant, le déploiement quotidien est non seulement impossible dans les grandes entreprises qui déploient des applications complexes, mais peut être également impossible du fait des restrictions réglementaires ou de l'entreprise. DevOps ne se limite pas simplement au déploiement, et certainement pas au déploiement continu vers la production. L'adoption de DevOps permet aux organisations de mettre en production lorsqu'elles le décident et non pas en fonction d'une simple date marquée sur un calendrier.

Notes

Notes

Notes

Notes

Le monde dépend du logiciel

Dans le monde d'aujourd'hui,où l'évolution est toujours plus rapide, DevOps constitue l'approche qui convient aux entreprises qui veulent être agiles et *Lean*. Afin de répondre rapidement aux besoins des clients et du marché, DevOps permet d'atteindre la livraison en continu de vos innovations logicielles. Ce guide décrit DevOps et explique les réels avantages qu'il offre à votre entreprise. Vous découvrirez également comment une vue holistique de DevOps, qui couvre l'ensemble du cycle de vie des logiciels, de l'idéation et de la conception de nouvelles fonctionnalités métier jusqu'à l'implémentation et la mise en production, peut apporter un avantage concurrentiel dans un monde de en continu.

- ***Dépasser les attentes des clients : Offrir une expérience de qualité optimale***
- ***Satisfaire les besoins de l'entreprise : répondre à l'accroissement spectaculaire de la fréquence et du volume des livraisons de logiciels***
- ***Exploiter les nouvelles tendances technologiques : utiliser les technologies mobiles, le Cloud, le Big Data et les réseaux sociaux***
- ***Améliorer la collaboration : engager les parties prenantes dans le cycle de vie de la livraison***

Vous apprendrez :

- **Fonctionnalités de DevOps**
- **Comment utiliser les technologies mobiles, le Cloud et les autres technologies de pointe**
- **Comment DevOps s'aligne sur les processus de gestion du cycle de vie des applications (ALM)**
- **Comment gérer les applications multi-tiers**
- **Témoignage d'utilisation de DevOps par IBM**

WILEY

ISBN : 978-1-119-17744-9
Numéro de référence : RAM14025-FRFR-00

WILEY END USER LICENSE AGREEMENT

Go to www.wiley.com/go/eula to access Wiley’s ebook EULA.