LIVRE BLANC • MARS 2018

Les bonnes pratiques DevOps et CA Continuous Delivery Automation

Packaging applicatif, enchaînements et modèle de déploiement standard

Table Des Matières

Section 1 3

La complexité : une évolution majeure

Section 2 3

Croissance de la matrice applicative, moins de maîtrise

Section 3 4

Développement et déploiements applicatifs avec CA Continuous Delivery Automation : vitesse et maîtrise

Section 4 4

Application, processus et modèles d'environnement

- Modèles applicatifs
- Modèles de processus
- Modèles d'environnement

Section 5 7

Conclusion

SECTION 1

La complexité : une évolution majeure

Applications et services comprennent aujourd'hui davantage de composants et d'intégrations. S'ajoute à cela une concurrence économique toujours plus acharnée et l'on constate rapidement à quel point la mission des équipes de développement et d'exploitation est devenue complexe.

Les équipes de développement sont sous pression pour livrer toujours plus de fonctionnalités (en particulier des évolutions mineures) sur des cycles très courts, à l'aide de méthodes agiles et d'approches spécifiques telles que l'intégration en continue.

Les environnements de production ne cessent de croître, supportant une quantité extraordinaire de serveurs et de services. La production, un environnement totalement physique il y a quelques années, est presque intégralement virtualisé aujourd'hui, ce qui permet d'assembler un serveur en quelques minutes. Enfin, la production se voit dans l'obligation de superviser un volume de ressources bien plus important.

Les équipes de développement délivrent plus vite en découpant les services, en parallélisant l'écriture du code et en consolidant les applications, multipliant au passage les étapes de configuration et les interdépendances.

Les méthodologies agiles impliquent que les développeurs passent moins de temps sur les projets internes qui, pourtant, pourraient alléger certains aspects de leur travail. D'un côté, les équipes de production rechignent à pallier les problèmes issus du développement. De l'autre, les développeurs refusent de s'investir sur des tâches de configuration et de déploiement. Au fil du temps, ces deux équipes se rejettent mutuellement la responsabilité : « Ce n'est pas le code, c'est la machine », « ce n'est pas la machine, c'est le code ». Le mouvement DevOps voit ainsi le jour, stipulant que chaque équipe doit comprendre la mission de l'autre, Cette transition impose que les équipes de production s'impliquent davantage dans les phases initiales du développement.

L'utilisation des outils de gestion de configuration comme Puppet et Chef est une aide précieuse. Au-delà, le plus dur reste à faire. Ce livre blanc explique le processus d'automatisation du déploiement applicatif à travers DevOps, les prérequis nécessaires à ce processus et en quoi ces prérequis s'appliquent au développement, à la production, ou les deux , sur toute la durée du cycle de vie.

SECTION 2

Croissance de la matrice applicative, moins de maîtrise

Le cycle de vie d'une application est un processus itératif. Une application est en phase continue de développement, test, correction, mise à jour et redéveloppement en étapes distinctes : développement, test fonctionnel, test d'intégration, pré-production et production.

La terminologie et numérotation assignées à chacune de ces étapes varient en fonction de l'application et de l'entreprise. Le processus reste le même. La progression d'une étape vers une autre implique le déploiement partiel ou total de l'application vers les serveurs d'un environnement donné. Dans certains cas, il peut être nécessaire d'installer des versions antérieures ou d'en installer sur des sites parallèles (pour des besoins de test).

Pour l'équipe DevOps, la difficulté augmente à chaque ajout de composant. Toute nouvelle déclinaison d'une configuration ou d'un environnement cible entraîne la multiplication des composants, environnements et paramètres de déploiement dans la matrice. Si l'on considère la multitude de sites hébergeant le développement d'applications à intégrer, il n'est pas surprenant de constater que de nombreuses entreprises perdent la maîtrise de leurs projets. Ces dernières années ont révélé un certain nombre de défaillances majeures dans les systèmes critiques de très grandes entreprises. Banques, télécoms et géants des services internet s'écroulent, victimes d'évolutions non-maîtrisées ou non-anticipées. La presse informatique cite « une gestion catastrophique des projets de déploiements » et « des mises à jour bâclées » pour expliquer ces échecs retentissants.

SECTION 3

Développement et déploiements applicatifs avec CA Continuous Delivery Automation : vitesse et maîtrise

L'objectif du mouvement DevOps est d'en finir avec ces problèmes. Ce mouvement est né d'une prise de conscience : les entreprises perdent la maîtrise de leurs projets informatiques précisément au point de convergence du développement et de la production. A l'origine, les équipes DevOps avaient pour mission de combler le « fossé culturel » qui sépare le développement et l'exploitation en poussant les deux équipes à collaborer le plus possible pendant toute la durée du cycle de vie d'une application. C'est encore le cas aujourd'hui.

Cet objectif sera désormais une des fonctions essentielles du DevOps.

Néanmoins, la taille des environnements, plus particulièrement le nombre de serveurs requis par l'exploitation, demeure une préoccupation significative. Des produits pour la gestion de hauts volumes de configurations et de versions multiples, comme Puppet (PuppetLab) et Chef (OpsCode) sont nécessaires pour supporter les infrastructures des environnements de taille importante. Cependant, même si ces solutions sont utiles pour standardiser un parc serveur, elles sont inadaptées au support de processus applicatifs nativement multi-environnements ou multicouches. Plus simplement, les outils de gestion des configurations opèrent à la demande : c'est l'environnement et les besoins de chaque serveur qui justifient leur utilisation. Inversement, l'automatisation des déploiements applicatifs (CA Continuous Delivery Automation) utilise une méthodologie précise. La première solution est davantage liée à l'environnement (« de quoi ai-je besoin sur chaque serveur de mon environnement ? »). L'autre s'articule autour du cycle de vie, autrement dit la définition des étapes à suivre pour promouvoir une version ou correctif d'un environnement vers un autre. Dans le contexte de la livraison continue, les outils CA Continuous Delivery Automation apportent davantage de maîtrise, d'efficacité et une utilisation efficace de la chaîne d'outillage DevOps sur toutes les couches de l'environnement IT, et pour tous les environnements où les applications et leurs dépendances respectives sont déployées. Les outils CA Continuous Delivery Automation associent modélisation et fonctionnalités d'automatisation particulièrement conçues pour s'intégrer à tout type de cycle de vie applicatif et de technologies de gestion.

SECTION 4

Application, processus et modèles d'environnement

Une solution CA Continuous Delivery Automation doit vous permettre de modéliser un processus de livraison de logiciel automatisé qui permettra aux différents intervenants de suivre les progrès des builds provenant des processus d'intégration continue à travers les multiples étapes du test et du déploiement. Les solutions CA Continuous Delivery Automation vous permettent de vous modéliser des composants physiques, des environnements et des étapes de déploiement. Ces modèles sont applicables facilement, et adaptés aux changements, à de nouveaux critères et de nouvelles situations.

Modèles applicatifs

La modélisation des applications est un prérequis fondamental de toute solution CA Continuous Delivery Automation digne de ce nom. Les applications réunissent de nombreux composants binaires, paramètres de configuration, scripts et services dépendants. Les composants proviennent de sources multiples : référentiels, serveurs de développement, partage de fichiers, serveurs FTP.... Chaque composant doit contenir une référence à la version du déploiement courant.

La création d'un modèle logique de votre application insère une couche d'abstraction entre les composants physiques et votre processus de déploiement, permettant aux mêmes mécanismes d'automatisation sous-jacents de s'exécuter dans tout type d'environnement. Certaines solutions de déploiement automatisés ou CA Continuous Delivery Automation mappent directement les composants physiques aux composants applicatifs modélisés et/ou aux cibles de déploiement des environnements.

Chaque nouvelle version d'application nécessitera une conception continuelle de votre modèle logique et impliquera de concevoir à nouveau votre processus de déploiement. Cette cartographie (et non pas modélisation) n'a pas les avantages auxquels on pourrait s'attendre pour la détection des problèmes, puisque les mécanismes d'automatisation eux-mêmes doivent être pris en compte dans la résolution du problème.

Les modèles applicatifs devraient fournir un package qui se comporte comme une liste de pièce (BoM) pour chaque processus de déploiement, représentant un ensemble de différents composants versionnés devant être déployés ensemble. Les packages devraient par ailleurs être livrés avec un traitement spécifique vous permettant de les accepter ou de les rejeter avant leur promotion. Cela vous conférerait une visibilité inégalée sur la productivité et l'efficacité de vos pratiques de livraison continue, et permettrait d'améliorer le processus.

Modèles de processus

Les solutions CA Continuous Delivery Automation ont recours à deux types de mécanismes d'automatisation : déclaratif (statut final souhaité) et enchaînement (comment parvenir à ce statut). Les modèles déclaratifs prédisent comment le processus de déploiement se réalisera sans l'intervention d'un fournisseur, conférant la flexibilité nécessaire aux équipes DevOps pour déterminer leurs propres bonnes pratiques ou s'adapter aux changements technologiques. Les modèles d'enchaînement, en revanche, confèrent aux équipes DevOps toute la flexibilité et la visibilité nécessaires à la livraison du logiciel.

Les enchaînements réalisent le gros du travail dans une solution CA Continuous Delivery Automation. Ils se substituent aux interventions manuelles nécessaires à l'installation, la mise à jour, l'application de correctifs ou la désinstallation et peuvent être réutilisés de façon illimitée sur l'intégralité de l'environnement d'hébergement de l'application. Les solutions CA Continuous Delivery Automation matures vous permettent de modéliser des enchaînements visuellement sur un canevas, comme un diagramme Microsoft Visio®-. Les enchaînements sont assemblés par simple glisser-déposer des étapes/tâches, provenant d'une vaste bibliothèque de comportements et d'intégration standards à la plupart des applications, allant de pair avec la chaîne d'outillage de l'intégration/livraison continue (CI/CD) et des systèmes dépendants.

Les exemples ci-après montrent l'étendue des types d'infrastructure et d'outils auxquels les enchaînements prêts à l'emploi s'appliquent :

- Serveurs applicatifs : Oracle Weblogic, IBM WebSphere et JBoss/IIS
- Serveurs de base de données : Oracle and Microsoft SQL Server®
- Serveurs d'intégration : BizTalk
- Technologies de conteneurs, comme Docker, Kubernetes et LXC
- Fournisseurs de Cloud : AWS, Microsoft AZURE™ et autres
- Commandes OS, gestionnaires de package et systèmes de contrôle de sources comme GitHub et Bitbucket
- Outils d'intégration continue : Jenkins, Bamboo et Travis CI

Les intégrations prêtes à l'emploi garantissent la cohérence et l'exactitude du processus de déploiement des évolutions vers les différents serveurs. Elles ont un autre avantage : elles peuvent être exécutées pour chaque compte utilisateur requis (par association), en respectant les contraintes de sécurité. Enfin, leur bonne exécution est systématiquement vérifiée.

Les enchaînements ont d'autres caractéristiques importantes :

Génériques: un enchaînement doit être totalement indépendant de l'environnement, des privilèges, des permissions et du contenu qu'il exécute. Cela permet à un plan donné d'être réutilisé pour les différentes versions de package et d'être exécuté sur n'importe quel environnement, du test à la production.

Contrôle de version: la parallélisation du développement des composants et services sur plusieurs équipes produit de nombreuses versions. Les enchaînements fonctionnent sur le même mode et il n'est pas rare que le code et les enchaînements soient développés par les mêmes équipes. Dans le cas de plusieurs équipes travaillant en parallèle, maintenir la cohérence des enchaînements et des versions passe forcément par le contrôle de version. Il est donc absolument nécessaire que votre plateforme d'automatisation autorise plusieurs équipes à collaborer sur des enchaînements de déploiement et versions multiples.

Sophistication et simplicité: les enchaînements de déploiement doivent être simples, tout en étant élaborés. Une série de processus automatisé est bien trop simpliste et rigide pour gérer la complexité, les nuances, et la tolérance aux pannes exigée par la gamme de technologies modernes. Les enchaînements doivent supporter les tentatives de relance, les échecs/réussites partiels, la prise de décision, la distribution de charge et bien plus encore. Les entreprises ont besoin de sophistication et de flexibilité, autant que de simplicité dans l'utilisation

Modèles d'environnement

Beaucoup d'erreurs sont le fait de disparités dans les paramètres d'environnement sur un déploiement donné et non d'anomalies dans les enchaînement. Le problème vient des logiciels et applications qui sont particulièrement sensibles aux variations (même minimes) d'environnements et de paramétrages. Un simple exemple : les différences entre la structure de répertoire de fichier entre les serveurs de production et de QA. Certaines sont plus difficiles à identifier, comme le port d'un service, les privilèges d'exécution ou la taille d'un cluster. La prise en compte de ces éléments est un véritable casse-tête. Des vérifications et réglages s'imposent pour chaque type de machine et le système. Dans le cas de centaines de serveurs à mettre à jour, une gestion mécanique de la matrice devient impossible et les scripts doivent être redéveloppés de façon systématique.

CA Continuous Delivery Automation propose un modèle permettant aux développeurs et opérateurs de définir leurs paramètres d'exécution de façon centralisée et automatique. Le modèle distribue lui-même les paramètres spécifiques à l'environnement (répertoires et versions) et les données de configuration (par exemple, les variables). Par exemple : cibles de déploiement, profils, connexions utilisateurs et plus encore. Non seulement ce type d'approche garantit la diffusion correcte des paramètres au moment voulu et pour la bonne exécution, mais il simplifie la définition et la gestion des enchaînements. Sans modèle prédéfini, l'utilisateur doit maintenir des enchaînements extrêmement complexes et stocker ses paramètres

d'exécution dans des sources externes, comme du XML ou des bases de données. Ainsi le système est moins sécurisé et devient plus complexe : l'utilisateur doit lire, analyser et segmenter différents chemins d'exécution formant le enchaînement global.

SECTION 5

Conclusion

L'automatisation du déploiement applicatif (CA Continuous Delivery Automation) demeure la seule méthode viable pour conserver une maîtrise totale sur une matrice complexe sur les variations d'applications, de déploiements et d'environnements. Sans contrôle et sans traçabilité, les processus de déploiement provoquent des dysfonctionnements systèmes graves, des impacts financiers qui se compte en millions, et rend l'objectif de DevOps inatteignable. CA Continuous Delivery Automation est le dernier obstacle dans la mise en œuvre de la livraison continue.

Afin de surmonter ces difficultés, il est impératif qu'une solution CA Continuous Delivery Automation offre les trois fonctions suivantes :

- Le packaging applicatif et des chemins de promotion, garants d'un déploiement sans erreur.
- Des enchaînements, qui confère uniformisation, et cohérence au processus de déploiement.
- Des modèles d'environnement qui garantissent l'assignation des paramètres et des configurations spécifiques à chaque environnement.

Même si d'autres aspects sont à prendre en compte, une solution CA Continuous Delivery Automation doit impérativement proposer ces trois fonctions pour remplir vos objectifs : rapidité, réduction des coûts et maîtrise totale.

Pour plus d'informations **ca.com/fr/automation**

Connectez-vous à CA Technologies

CA Technologies fournit les logiciels qui aident les entreprises à opérer leur transformation numérique. Dans tous les secteurs, les modèles économiques des entreprises sont redéfinis par les applications. Partout, une application sert d'interface entre une entreprise et un utilisateur. CA Technologies aide ces entreprises à saisir les opportunités créées par cette révolution numérique et à naviguer dans l'« Economie des Applications ». Grâce à ses logiciels pour planifier, développer, gérer la performance et la sécurité des applications, CA Technologies aide ainsi ces entreprises à devenir plus productives, à offrir une meilleure qualité d'expérience à leurs utilisateurs et leur ouvre de nouveaux relais de croissance et de compétitivité sur tous les environnements : Mobile, Cloud, Distribué ou Mainframe. Pour plus d'informations : www.ca.com/fr.

CS200-374355_0618