Avis d'expert

7 facteurs clés pour réussir votre démarche DevOps

Par Maxime Orand, Consultant, Hardis Group

Si la démarche DevOps a, sur le papier, tout pour séduire, force est de constater que malgré toutes les bonnes volontés, elle peine parfois à convaincre dans les faits. En cause, un manque de préparation et des étapes non respectées. 7 essentiels à connaître avant de se lancer (ou pour corriger le tir d'une démarche embourbée).

1 – Modéliser

À elle seule, la réduction des cycles ne suffit pas à totalement fluidifier les processus. Au contraire, des livraisons multiples sont autant de points de frictions potentiels. Pour réduire ce risque, il convient donc d'identifier au plus tôt les options d'optimisation. Ce qui passe en particulier par une modélisation de l'ensemble des processus de livraison, mais également par une définition précise et sans ambiguïté des responsabilités et des livrables associés à chaque étape du cycle. En d'autres termes, l'automatisation et la rapidité de DevOps ne dispensent en aucun cas du formalisme de livraison et de documentation, et de la mise en place de points de contrôle humains.

Cet effort de transparence doit en outre être l'occasion de mettre certaines tâches en lumière et ne plus les sacrifier sur l'autel des exigences du planning. Il s'agit bien évidemment des tâches d'exploitation, en toute fin de chaîne de production donc, mais dont dépend, au même titre que les autres processus, le service global rendu.

2 – Privilégier la culture à l'outillage

Trop souvent, l'évolution des méthodes se borne à la dotation d'outils. Or, l'évolution de la culture est non seulement tout aussi importante dans la réussite, mais elle est même préliminaire : l'outillage n'étant qu'un moyen de mise en œuvre. Les modélisations préalablement évoquées étant mises en place, il s'agit alors de s'interroger sur les ressources humaines (existence, formation, recrutement…), l'organisation, la gouvernance et plus généralement l'information des équipes quant aux objectifs attendus. En bref, même si elles s'avèrent chronophages et complexes, les étapes d'accompagnement au changement et de réorganisation des équipes sont essentielles pour une parfaite adhésion à la démarche.

À ce titre, les cérémonies de partage et de décision constituent une composante majeure de ce changement de culture : ces rencontres permettent aux différents groupes de travail d'interagir et de communiquer efficacement entre eux, afin qu'ils puissent partager leurs problématiques et s'aligner sur les enjeux de chaque itération. La mise en place de nouveaux outils et de mécanismes d'automatisation ne peut pleinement porter ses fruits qu'une fois les changements de culture opérés.

3 – Concevoir un référentiel commun

Par définition, chaque équipe, selon son métier (développement, test, intégration, exploitation), voit chaque processus depuis son propre spectre. Or, pour favoriser les synergies et assurer la réussite de DevOps, il est essentiel que toutes les équipes aient la même vue d'ensemble : définition d'un langage commun, contextualisation des méthodes de travail de chaque dans l'ensemble du cycle de livraison, etc. Ce qui se traduira également sur le terrain par la mise en place d'outils collaboratifs : suivi de bugs et d'incidents, planning de livraisons et planning projet, engagements de service, back logs partagés.

Un environnement de travail commun à toutes les équipes donc, mais également des informations accessibles à tous, un dispositif favorable à une meilleure sensibilisation de chacun aux impératifs et contraintes des autres. L'objectif étant de créer des zones de chevauchement (développeurs ayant accès aux logs et au monitoring, exploitants réalisant eux-mêmes des tests…), afin de dégager une compréhension réciproque entre équipes, tout en assurant une meilleure responsabilisation de tous.

4 - Respecter la séquentialité de la démarche DevOps

De l'agilité au DevOps, il n'y a pas qu'un seul pas. Si l'objectif est d'atteindre l'opérationnel continu, la démarche passe préalablement par 3 étapes essentielles : l'intégration continue, la livraison continue et le déploiement continu. Chacune étant dépendante des précédentes. En effet, ce n'est qu'une fréquence élevée des livraisons (issues des étapes d'intégration et de livraison continues) qui entraînera le besoin d'automatisation des déploiements puis, dans la foulée, d'automatisation des opérations (plan de reprise d'activité, monitoring, administration, restauration).

Vouloir tout mettre sur pied en même temps n'a donc aucun sens, et risque même d'être contre-productif : mieux vaut en effet stabiliser une étape pour assurer la réussite de la suivante. Il n'est toutefois pas question d'atteindre la maturité complète d'une étape avant d'aborder la suivante : la chaîne doit être abordée dans son ensemble dès le début, afin d'anticiper les dépendances entre outils, et sensibiliser les équipes de développement à l'utilisation qui sera faite de leur code.

5 – Mesurer et améliorer la performance

Chaque opportunité de revoir les différents processus doit être saisie : revue de sprint, revue des bugs (qualité), revue des déploiements, revue des incidents, revue de performance, etc. Des revues systématiques où chaque individu peut (et doit) être contributeur de l'amélioration continue. Se renvoyer la responsabilité est inutile : les constats doivent être partagés et donner systématiquement lieu à des actions qui devront être suivies et priorisées au niveau du projet.

Dans ce contexte, il est nécessaire de mettre en place des indicateurs de performances pour suivre le projet dans le temps et créer des boucles d'amélioration continue : livraisons effectuées vs attendues, nombre de bugs détectés vs incidents en production, durée des déploiements, utilisation de l'infrastructure, etc. Et pour chaque problème identifié, de se poser la question de l'automatisation de son identification, voire de sa correction. Les investissements à effectuer pour traiter un problème en amont sont souvent plus rentables que les « bricolages » réalisés en bout de chaîne.

6 – Tenter l'organisation matricielle et la transparence

On l'a vu : dans une équipe DevOps, la définition des responsabilités de chacun en toute transparence permet d'identifier facilement les rôles. Pour aller encore plus loin et développer les notions de partage et de communication, l'organisation matricielle se révèle extrêmement pertinente : à chaque groupe de développeurs en charge d'une fonctionnalité ou d'un bloc de composants du projet, sont associés un testeur et un profil DevOps, ces deux derniers faisant en parallèle partie des groupes Qualité et Exploitation. Les zones de chevauchement sont ainsi « incarnées » et la cohésion d'ensemble assurée.

Ce qui renforce également la communication, fondamentale dans la réussite du DevOps, tant la rétention ou la dissimulation d'information peuvent impact particulièrement grave dans ce type d'organisation. Les managers doivent donc à la fois imposer une communication très fréquente et adopter un comportement positif : pointer du doigt une responsabilité ne sert à rien. Trouver des solutions ensemble est beaucoup plus efficace. En matière de DevOps, la transparence n'est donc pas une recommandation mais un impératif.

7 – Intégrer les enjeux des opérations et du support

Jusque-là principalement focalisés sur l'intégration et le déploiement continu, les méthodes agiles et le DevOps s'invitent progressivement dans les phases dites de production. Désormais acteurs du dispositif, les exploitants doivent formuler clairement leurs exigences, attentes et cas d'usage, et sensibiliser les autres équipes aux problématiques rencontrées en production. Cela doit faire partie intégrante des phases de collecte des exigences non fonctionnelles, notamment dans le cadre de l'automatisation des opérations.

De la même façon, le support ne doit pas se contenter de regarder se succéder les incidents : il s'agit d'enrichir les cas d'usage testés par les scripts de l'équipe qualité, et proposer de nouveaux axes de monitoring aux développeurs et exploitants. En bref, par la collecte des exigences ou par l'amélioration continue, l'organisation doit permettre la prise en compte de cet aspect du cycle de vie de l'application.

A propos de Hardis Group

Entreprise de conseil, de services du numérique et éditeur de logiciels, Hardis Group accompagne ses clients dans la transformation de leur business model, de leur chaine de valeur numérique et d'exécution logistique. La société les aide à gagner compétitivité et en performance opérationnelle, en concevant et intégrant des solutions métiers, technologiques et digitales adaptées à leurs besoins et enjeux.

Hardis Group a développé des expertises dans les secteurs de l'assurance, de la distribution, de l'industrie et énergie, de la e-santé, ou encore de la prestation logistique. Expertises qui lui permettent aujourd'hui de proposer des réponses globales, dans une approche agile de co-construction, d'innovation et d'amélioration continue.

Depuis sa création en 1984, la société construit sa croissance sur une approche pragmatique ainsi que des valeurs d'efficacité et d'engagement fort, tant auprès de ses 2 500 clients que de ses 850 collaborateurs. En 2016, Hardis Group a réalisé un chiffre d'affaires de 83,3 millions d'euros. Le groupe, dont le siège social est situé à Grenoble, dispose de cinq autres agences à Lyon, Paris, Lille, Bordeaux et Nantes.

www.hardis-group.com

Contacts presse

Anjuna
Elodie Cassar
elodie.cassar@anjuna.fr
Tel : +33 9 64 15 31 27
GSM : +33 6 80 53 82 94

Hardis Group
Hélène Leclercq
helene.leclercq@hardis.fr
Tél.: +33 4 76 70 98 41