Cyborg Intelligence

DevOps

Stratégie & méthodes

Programme

- Jour 1: Théorie Matin / Pratique Après midi en groupe
- Jour 2: Théorie Matin / Pratique Après midi en groupe
- Jour 3: Pratique et présentation matin / Questionnaire à 13h35

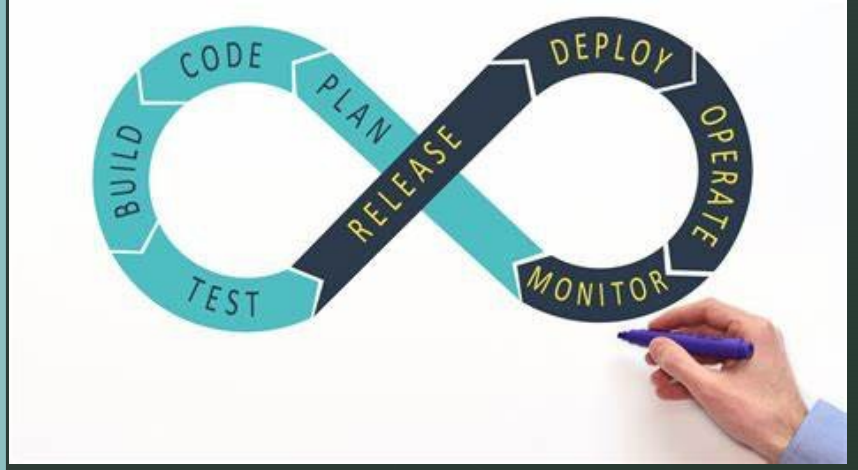
Tour de Table

Objectifs Formation

- Intégrer les principes « DevOps », les concepts et les techniques qui vous permettront d'augmenter le ROI réalisé lors de la conception et de la livraison de logiciels en Agile
- Apprendre à choisir les outils et à les configurer
- Savoir identifier les aspects positifs positifs et négatifs de la culture « DevOps »

DevOps - Définitions

- DevOps est un ensemble de pratiques qui **met l'accent sur la collaboration et la communication** entre les développeurs de logiciels et les professionnels des opérations informatiques, **en automatisant** le processus de livraison de logiciels et les changements d'infrastructure.
- Le terme DevOps est né de l'union du «development» et des «operations» dont l'objectif est favoriser **une meilleure communication entre les deux équipes**.



DevOps - Définitions

- DevOps vise à créer une culture et un environnement dans lesquels la conception, les tests et la diffusion de logiciels peuvent être réalisés rapidement, fréquemment et efficacement.

- **DevOps n'est pas seulement une méthodologie, c'est une véritable philosophie de travail.**

Concepts Clés

- Aujourd'hui, les entreprises évoluent vers **une approche dynamique orientée client** pour le développement et la livraison de leurs applications.

- Dans un environnement où les clients se tournent vers des transactions numériques à l'ère du mobile, **le rôle des développeurs** d'applications devient **incontournable dans l'expérience client**.

- En parallèle, la **tendance à l'agilité** a été une source d'inspiration pour DevOps dont l'un des points-clés agiles **favorise les professionnels et leurs interaction** plutôt que les processus et les outils.

Concepts Clés

- Au cours des dernières années, les **équipes de développement et d'exploitation** ont amélioré significativement leur façon de travailler.

- Mais aujourd'hui, le besoin de réaligner ces deux équipes se renforce. Le mouvement DevOps **naît de ce besoin de réalignement**

Concepts Clés

- DevOps est à l'origine d'une philosophie qui transforme complètement la façon dont les professionnels de l'informatique perçoivent la stabilité et le fonctionnement du système qu'ils gèrent, ainsi que leur propre rôle dans le flux de valeur ajoutée du début à la fin.
- Le **Cloud Computing** et les **réseaux logiciels** SDN, Software-Defined Network) sont deux éléments qui ont accéléré la **destruction des silos** lesquels séparaient les équipes de développement et d'exploitation.

Cloud : la démocratisation Devops

Concepts Clés

- La **rivalité “development VS operations” est en train de disparaître**, quoique progressivement, car les deux services sont de plus en plus conscients de faire partie de la même équipe et d’œuvrer vers le même objectif.

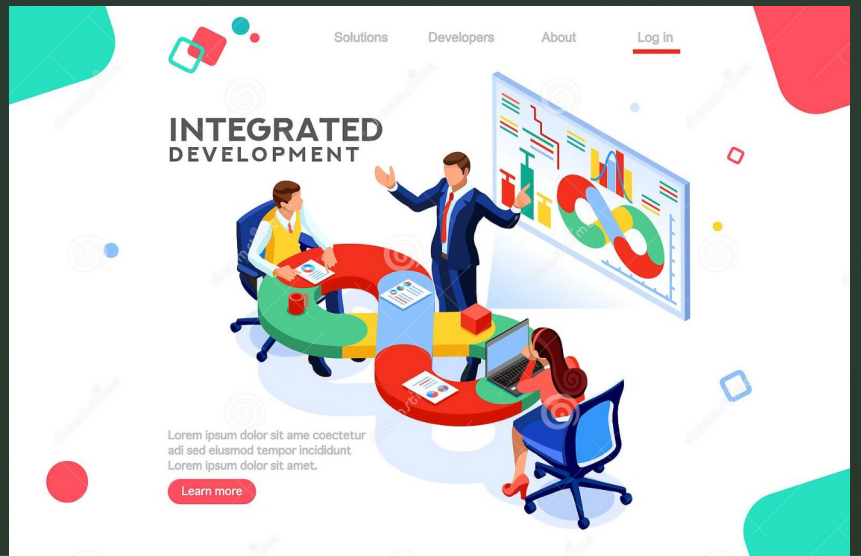
- DevOps permet aux sociétés d’être **réactif au Time to market avec célérité et donc, de satisfaire les besoins clients.**

Concepts Clés

- Le mouvement DevOps rassemble, et continue d'intégrer, de nombreux principes et bonnes pratiques qui peuvent être adoptés par **des organisations IT de toutes tailles (internes ou externes).**
- Toutes ces expériences ont créé **une approche qui vise à améliorer la façon dont l'IT apporte de la valeur ajoutée à ses clients.**

Concepts Clés

- Les professionnels certifiés DevOps satisfont cet objectif grâce à une amélioration de la communication et de la collaboration entre les équipes informatiques ainsi qu'une meilleure intégration des techniques, processus et personnes.



8 Avantages

- 1. Amélioration de la qualité du code, des produits et des services (réduction des anomalies, taux de réussite des changements plus important, etc.)
- 2. Efficacité accrue (par exemple, optimisation du temps consacré aux activités qui créent de la valeur ajoutée: une valeur ajoutée sans précédent pour le client)
- 3. Amélioration du délai de mise en place sur le marché>
- 4. Meilleur alignement entre l'informatique et les métiers

8 Avantanges

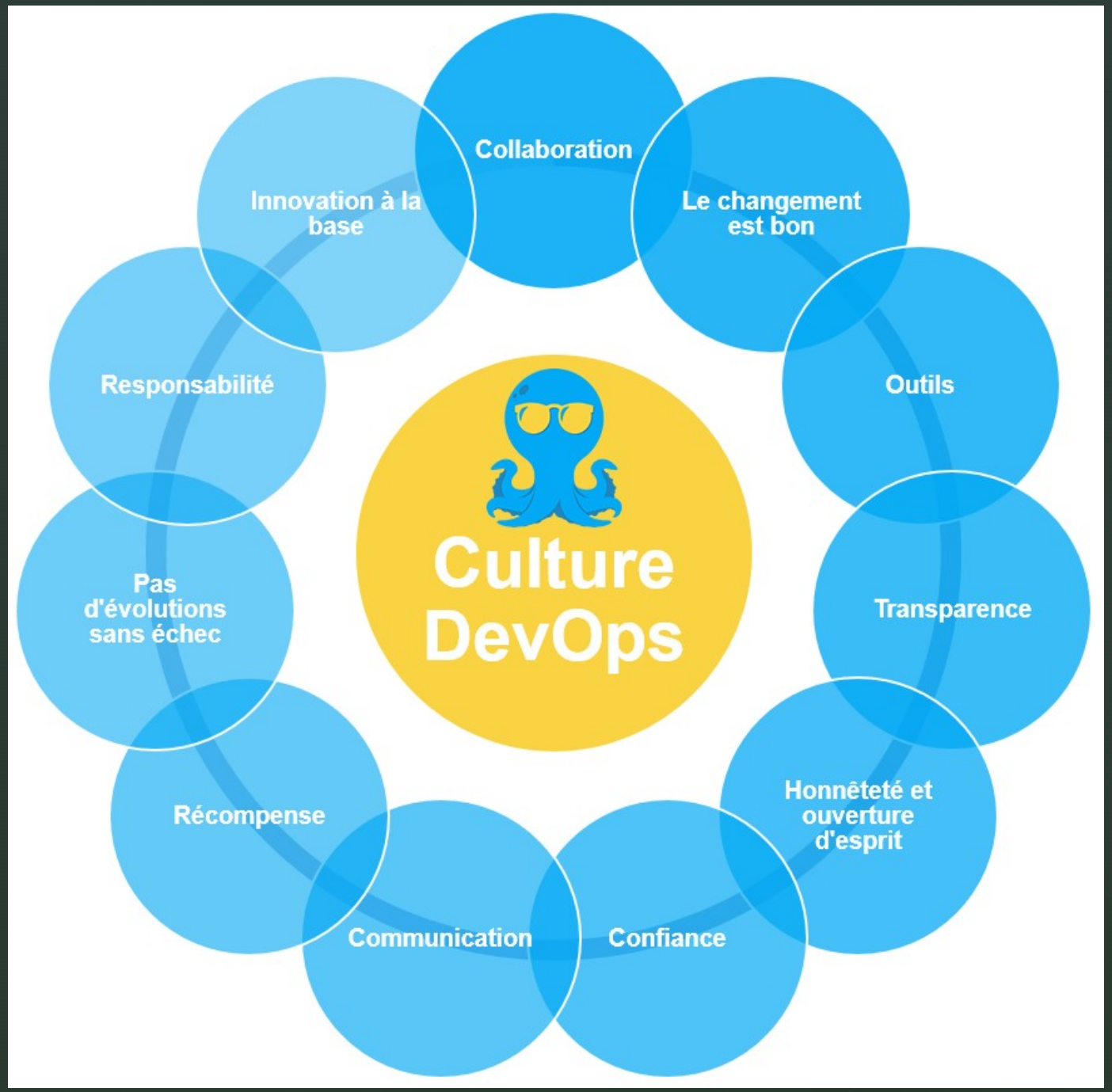
- 5. Des versions de plus petite taille fournies très rapidement et très fréquemment
- 6. Amélioration de la productivité, satisfaction du client, satisfaction du personnel
- 7. Moins de risques et moins de retours arrière
- 8. Réduction des coûts à long terme

Culture Devops

- Le mouvement DevOps s’appuie sur l’adoption et l’intégration de trois principales démarches ou méthodes actuelles
  - Les méthodes Agile de développement logiciel telles que Scrum.
  - La gestion des Services IT (ITSM) liée aux bonnes pratiques préconisées par ITIL.
  - Lean qui permet d’optimiser le travail et améliorer la qualité de la production

Culture Devops

- DevOps n’est pas une méthode ou un changement de processus. Il demande un **changement de la culture organisationnelle**. Les objectifs conflictuels des départements IT rendent ce changement culturel difficile.
- Les **équipes d’exploitation** cherchent la **stabilité** alors que les **développeurs** demandent **des changements fréquents** et les **testeurs** sont là pour **minimiser les risques**.
- La collaboration et l’intégration intelligente de ces équipes est **un challenge décisif à l’adoption de DevOps dans une entreprise.**


Collaboration
Le changement est bon
Outils
Transparence
Honnêteté et ouverture d'esprit
Confiance
Communication
Récompense
Pas d'évolutions sans échec
Responsabilité
Innovation à la base
Culture DevOps

Métiers du DevOps

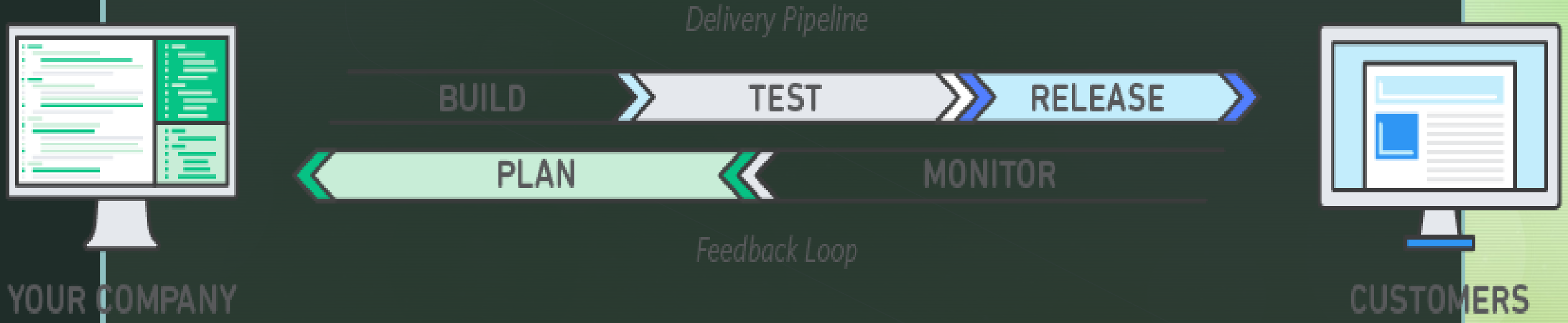
- En raison d'un développement orienté client par les sociétés et l'accroissement de l'adoption de DevOps, cette **compétence DevOps** (ou double compétence: développeur et ingénieur système) est de plus en plus recherchée par les entreprises.
- Une étude dévoile et témoigne le **top 5 des fonctions en tension** sur le marché de l'emploi du **secteur de l'IT et du digital**

Métiers du DevOps

- Le poste d’ingénieur en informatique industrielle
- La **compétence DevOps**
- Consultant fonctionnel
- Agent de contrôle qualité logicielle
- Ingénieur IT applicatif

DevOps selon AWS

DevOps selon AWS



DevOps selon AWS

- DevOps est une combinaison de philosophies culturelles, de pratiques et d'outils qui améliore la capacité d'une entreprise à livrer des applications et des services à un rythme élevé.

- Il permet de faire évoluer et d’optimiser les produits plus rapidement que les entreprises utilisant des processus traditionnels de développement de logiciels et de gestion de l’infrastructure. Cette vitesse permet aux entreprises de mieux servir leurs clients et de gagner en compétitivité.

DevOps selon AWS

- Dans un modèle DevOps, les équipes de développement et d'opérations ne sont plus isolées. Il arrive qu'elles soient fusionnées en une seule et même équipe.

- Les ingénieurs qui la composent travaillent alors sur tout le cycle de vie d'une application, de la création à l’exploitation, en passant par les tests et le déploiement, et développent toute une gamme de compétences liées à différentes fonctions.

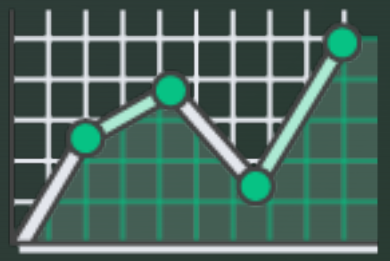
DevOps selon AWS

- Dans certains modèles DevOps, les équipes d'assurance qualité et de sécurité peuvent également s'intégrer étroitement au développement et aux opérations, ainsi qu'à l'ensemble du cycle de vie des applications.

- Lorsque la sécurité est au cœur de l'activité d'une équipe DevOps, on parle parfois de DevSecOps.

DevOps selon AWS

- Ces équipes utilisent des pratiques pour automatiser des processus qui étaient autrefois manuels et lents.
- Elles exploitent une pile technologique et des outils qui les aident à faire fonctionner et à faire évoluer les applications de façon rapide et fiable.
- Ces outils aident également les ingénieurs à accomplir de façon autonome des tâches (par exemple, le déploiement de code ou la mise en service d'infrastructure) qui nécessiteraient normalement l'aide d'autres équipes, ce qui augmente encore davantage la productivité de l’équipe d’ingénieurs.



DevOps selon AWS

Rapidité

- Avancer plus rapidement pour accélérer le rythme des innovations pour vos clients, améliorer votre capacité d'adaptation au marché et gagner en efficacité et en croissance.
- Avec le modèle DevOps, ces objectifs sont à la portée de vos équipes de développement et d'opérations. Par exemple, les microservices et la livraison continue permettent aux équipes de s'approprier les services et de les mettre à jour plus rapidement.

DevOps selon AWS

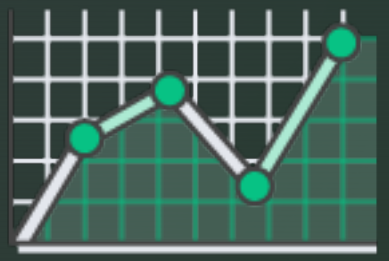
Livraison rapide

- Augmentez le rythme et la fréquence des publications de façon à innover et à optimiser vos produits plus rapidement. Plus vite vous publiez de nouvelles fonctionnalités et corrigez des bogues, plus vite vous pouvez répondre aux besoins de vos clients et gagner en compétitivité.
- L'intégration continue et la livraison continue sont des pratiques qui automatisent le processus de publication de logiciel, de la création au déploiement.

DevOps selon AWS

Fiabilité

- Assurez la qualité des mises à jour d'applications et des changements d'infrastructure afin de livrer en toute confiance vos produits à un rythme accéléré tout en continuant de proposer une expérience positive aux utilisateurs finaux.
- Utilisez des pratiques comme l'intégration continue et la livraison continue pour vous assurer que chaque changement est fonctionnel et sûr. Les pratiques de supervision et de journalisation vous aident à rester informé sur les performances en temps réel.



DevOps selon AWS

Évolutivité

- Opérez et gérez vos processus d'infrastructure et de développement à grande échelle. L'automatisation et la cohérence vous aident à gérer les systèmes complexes ou changeants de manière efficace et moins risquée.
- Par exemple, l'infrastructure en tant que code vous aide à gérer vos environnements de développement, de test et de production de façon répétitive et plus efficace.

DevOps selon AWS

Collaboration améliorée

- Créez des équipes plus efficaces avec un modèle culturel DevOps, qui insiste sur des principes comme la prise de responsabilité. Les équipes de développement et d'opérations collaborent étroitement, partagent de nombreuses responsabilités et combinent leurs flux de travail.
- Cela leur permet de limiter les pertes d'efficacité et de gagner du temps (par exemple en réduisant les délais de transfert entre les équipes de développement et d'opérations et en écrivant du code prenant en compte l'environnement dans lequel il est exécuté).

DevOps selon AWS

Sécurité

- Avancez rapidement tout en gardant le contrôle et en préservant la conformité. Vous pouvez adopter un modèle DevOps sans sacrifier la sécurité, en utilisant des politiques de conformité automatisées, des contrôles plus rigoureux et des techniques de gestion de la configuration.
- Par exemple, avec l'infrastructure en tant que code et la politique en tant que code, vous pouvez définir et suivre la conformité à n'importe quelle échelle.

DevOps selon AWS

- Les logiciels et Internet ont transformé le monde et les secteurs d'activité, du commerce au divertissement en passant par les banques.
- Les logiciels ne se contentent plus de soutenir les entreprises : ils sont aujourd'hui un composant essentiel de leurs activités. Les entreprises interagissent avec leurs clients à travers des logiciels livrés en tant que services ou applications en ligne et sur toutes sortes d'appareils.

DevOps selon AWS

- Elles peuvent également utiliser les logiciels pour gagner en efficacité opérationnelle en transformant chaque aspect de la chaîne de valeurs, comme la logistique, les communications et les opérations.
- Autant les entreprises spécialisées dans les biens physiques ont transformé leurs méthodes de conception, de création et de livraison de produits à l'aide de l'automatisation industrielle tout au long du 20e siècle, autant les sociétés modernes doivent adapter leur façon de créer et de livrer les logiciels.

DevOps selon AWS

- La transition vers DevOps implique un changement de culture et d'état d'esprit. Pour simplifier, le DevOps consiste à éliminer les obstacles entre deux équipes traditionnellement isolées l'une de l'autre : l'équipe de développement et l'équipe d'opérations.
- Certaines entreprises vont même jusqu'à ne pas avoir d'équipes de développement et d'opérations distinctes, mais des ingénieurs assurant les deux rôles à la fois.

DevOps selon AWS

- Avec DevOps, les deux équipes travaillent en collaboration pour optimiser la productivité des développeurs et la fiabilité des opérations.
- Elles ont à cœur de communiquer fréquemment, de gagner en efficacité et d’améliorer la qualité des services offerts aux clients.
- Elles assument l'entière responsabilité de leurs services, et vont généralement au-delà des rôles ou postes traditionnellement définis en pensant aux besoins de l'utilisateur final et à comment les satisfaire.

DevOps selon AWS

- Les équipes d'assurance qualité et de sécurité peuvent également s'intégrer étroitement aux équipes de développement et d’opérations.

- Quelle que soit leur structure organisationnelle, les organisations adoptant un modèle DevOps disposent d’équipes qui considèrent l’ensemble du cycle de développement et d'infrastructure comme faisant partie de leurs responsabilités.

DevOps selon AWS

- Il existe quelques pratiques clés pouvant aider les organisations à innover plus rapidement par le biais de l'automatisation et de la rationalisation des processus de développement de logiciels et de gestion de l'infrastructure.
- La plupart de ces pratiques sont rendues possibles par l'utilisation d’outils appropriés.
- Une pratique fondamentale consiste à réaliser des mises à jour très fréquentes, mais à petite échelle.

DevOps selon AWS

- Ainsi, les entreprises innovent plus rapidement pour leurs clients. Ces mises à jour sont généralement de nature plus incrémentielle que les mises à jour occasionnelles associées aux pratiques de publication traditionnelles.
- Le recours à des mises à jour fréquentes, mais à petite échelle, limite les risques associés à chaque déploiement.

DevOps selon AWS

- Les équipes ont ainsi plus de facilité à détecter les bugs, car pouvant identifier le dernier déploiement ayant provoqué l'erreur.

- Bien que la cadence et la taille des mises à jour soient variables, les entreprises utilisant un modèle DevOps déploient des mises à jour beaucoup plus fréquemment que les entreprises utilisant des pratiques traditionnelles de développement de logiciels.

DevOps selon AWS

- Les organisations peuvent également utiliser une architecture de microservices pour rendre leurs applications plus flexibles et favoriser des innovations plus rapides.

- L'architecture de microservices fragmente de grands systèmes complexes en des projets simples et indépendants.

DevOps selon AWS

- Les applications sont divisées en plusieurs composants (services) individuels, chaque service étant associé à une mission ou fonction spécifique et exploité indépendamment des autres services et de l'application tout entière.
- Cette architecture réduit les coûts de coordination liés aux mises à jour d'applications, et quand chaque service est tenu par de petites équipes agiles prenant pleine responsabilité de chaque service, les entreprises avancent plus rapidement.

DevOps selon AWS

- Cependant, la combinaison des microservices et d'une fréquence de publication plus élevée entraîne une augmentation considérable du nombre de déploiements, ce qui peut poser des problèmes opérationnels.
- Les pratiques DevOps comme l'intégration et la livraison continues résolvent donc ces problèmes et permettent aux entreprises d’assurer des livraisons rapides, fiables et sûres.

DevOps selon AWS

- Les pratiques d'automatisation de l'infrastructure, comme l'infrastructure en tant que code et la gestion de la configuration, aident à préserver le caractère Elastic et la réactivité des ressources informatiques face aux modifications fréquentes.
- En outre, le recours à la supervision et à la journalisation aide les ingénieurs à suivre les performances des applications et de l'infrastructure de manière à réagir rapidement en cas de problème.

DevOps selon AWS

Liste des bonnes pratiques DevOps

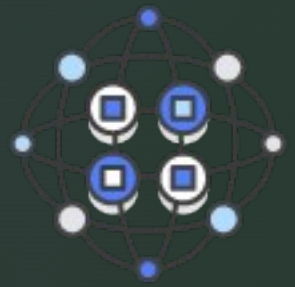
- Intégration continue
- Livraison continue
- Microservices
- Infrastructure en tant que code
- Surveillance et journalisation
- Communication et collaboration

DevOps selon AWS

- L'intégration continue est une méthode de développement de logiciel dans laquelle les développeurs intègrent régulièrement leurs modifications de code à un référentiel centralisé, suite à quoi des opérations de création et de test sont automatiquement menées.
- Les principaux objectifs de l'intégration continue sont : trouver et corriger plus rapidement les bogues, améliorer la qualité des logiciels et réduire le temps nécessaire pour valider et publier de nouvelles mises à jour de logiciels.

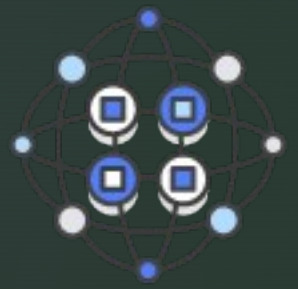
DevOps selon AWS

- La livraison continue est une méthode de développement de logiciels dans laquelle les changements de code sont automatiquement générés, testés et préparés pour une publication dans un environnement de production.
- Cette pratique étend le principe de l'intégration continue en déployant tous les changements de code dans un environnement de test et/ou un environnement de production après l'étape de création. Une bonne livraison continue permet aux développeurs de toujours disposer d'un artéfact prêt au déploiement ayant suivi un processus de test normalisé.



DevOps selon AWS

- L'architecture de microservices est une approche de conception qui consiste à diviser une application en un ensemble de petits services.
- Chaque service est exécuté par son propre processus et communique avec les autres services par le biais d'une interface bien définie et à l'aide d'un mécanisme léger, typiquement une interface de programmation d'application (API) HTTP.



DevOps selon AWS

- Les microservices sont conçus autour de capacités métier ; chaque service est dédié à une seule fonction.

- Vous pouvez utiliser différents frameworks ou langages de programmation pour écrire des microservices et les déployer indépendamment, en tant que service unique ou en tant que groupe de services.

DevOps selon AWS

- L'infrastructure en tant que code est une pratique qui implique la mise en service et la gestion de l'infrastructure à l'aide de code et de techniques de développement de logiciels, notamment le contrôle des versions et l'intégration continue.
- Le modèle de cloud axé sur les API permet aux développeurs et aux administrateurs système d'interagir avec l'infrastructure de manière programmatique et à n'importe quelle échelle, au lieu de devoir installer et configurer manuellement chaque ressource.

DevOps selon AWS

- Ainsi, les ingénieurs peuvent créer une interface avec l'infrastructure à l'aide d'outils de code et traiter l'infrastructure de la même manière qu'un code d'application.
- Puisqu'ils sont définis par du code, l'infrastructure et les serveurs peuvent être rapidement déployés à l'aide de modèles standardisés, mis à jour avec les derniers correctifs et les dernières versions ou dupliqués de manière répétable.

DevOps selon AWS

Gestion de la configuration

- Les développeurs et les administrateurs système utilisent du code pour automatiser la configuration du système d'exploitation et de l'hôte, les tâches opérationnelles et bien plus encore.
- Le recours au code permet de rendre les changements de configuration répétables et standardisés. Les développeurs et les administrateurs système ne sont donc plus tenus de configurer manuellement les systèmes d'exploitation, les applications système ou les logiciels de serveurs.

DevOps selon AWS

- Les entreprises surveillent les métriques et les journaux pour découvrir l'impact des performances de l'application et de l'infrastructure sur l'expérience de l'utilisateur final du produit.
- En capturant, catégorisant et analysant les données et les journaux générés par les applications et l'infrastructure, les organisations comprennent l'effet des modifications ou des mises à jour sur les utilisateurs, afin d'identifier les véritables causes de problèmes ou de changements imprévus.

DevOps selon AWS

- La surveillance active est de plus en plus importante, car les services doivent aujourd'hui être disponibles 24 h/24 et 7 j/ 7, et la fréquence des mises à jour d'infrastructure augmente sans cesse.
- La création d'alertes et l'analyse en temps réel de ces données aident également les entreprises à surveiller leurs services de manière plus proactive.

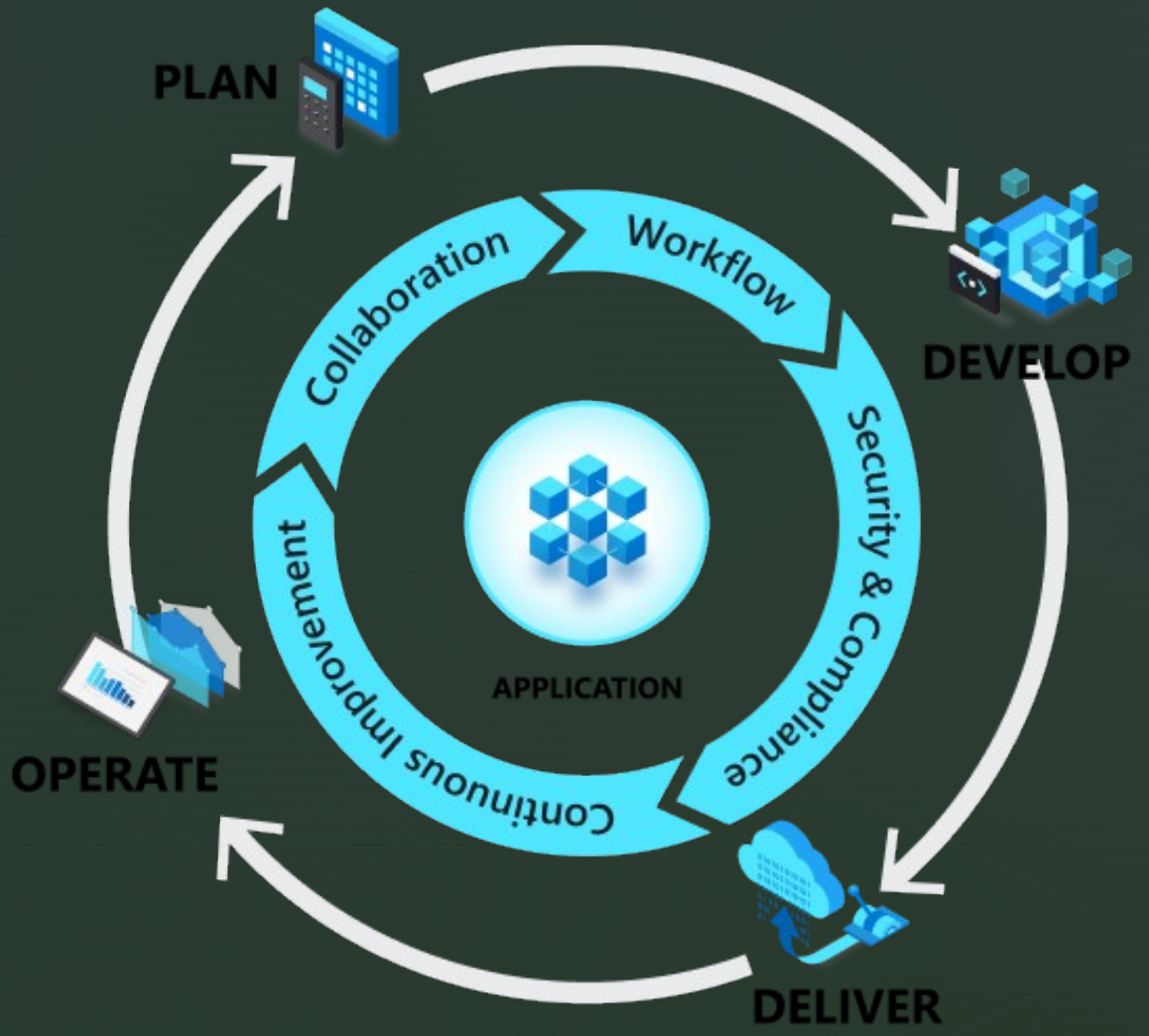
DevOps selon AWS

- L'instauration d'une meilleure collaboration et d’une meilleure communication au sein de l'organisation est un des principaux aspects culturels de DevOps.
- Le recours aux outils DevOps et l'automatisation du processus de livraison des logiciels établit la collaboration en rapprochant physiquement les flux de travail et les responsabilités des équipes de développement et d’opérations.

DevOps selon AWS

- Partant de cela, ces équipes instaurent des normes culturelles fortes autour du partage des informations et de la facilitation des communications, et ce par le biais d'applications de messagerie, de systèmes de suivi des problèmes ou des projets.
- Cela permet d'accélérer les communications entre les équipes de développement et d’opérations et même d'autres services, par exemple le marketing et les ventes, afin d'aligner plus étroitement chaque composant de l'organisation sur des objectifs et des projets communs.

DevOps selon Microsoft Azure



DevOps selon Microsoft Azure

- DevOps influence le cycle de vie des applications tout au long de leurs phases de planification, de développement, de livraison et d'exploitation.
- Ces phases reposent les unes sur les autres et ne sont pas spécifiques à un rôle. Au sein d'une culture DevOps, chaque rôle est, dans une certaine mesure, impliqué dans les différentes phases.

DevOps selon Microsoft Azure

Planifier

- Au cours de la phase de planification, les équipes DevOps imaginent, définissent et décrivent les fonctionnalités des applications et des systèmes qu'elles créent.
- Elles suivent leur progression à des niveaux de granularité faibles et élevés, des tâches portant sur un seul produit aux tâches couvrant des portefeuilles de plusieurs produits.
- La création de backlogs, le suivi des bogues, la gestion du développement logiciel agile avec Scrum, l'utilisation de tableaux Kanban et la visualisation des progrès réalisés grâce aux tableaux de bord illustrent les moyens dont disposent les équipes DevOps pour planifier avec agilité et visibilité.

DevOps selon Microsoft Azure

Développement

- La phase de développement comprend tous les aspects du codage (écriture, test, révision et intégration du code par les membres de l'équipe), ainsi que la génération de ce code dans des artefacts de build pouvant être déployés dans divers environnements.
- Les équipes DevOps entendent innover rapidement, sans sacrifier la qualité, la stabilité et la productivité.
- Pour ce faire, elles utilisent des outils hautement productifs, automatisent des étapes simples et manuelles, et effectuent des itérations par petits incréments moyennant des tests automatisés et une intégration continue

DevOps selon Microsoft Azure

Fournir

- La livraison consiste à déployer des applications dans des environnements de production de manière cohérente et fiable. La phase de livraison englobe également le déploiement et la configuration de l'infrastructure de base entièrement régie qui constitue ces environnements.
- Lors de la phase de livraison, les équipes définissent un processus de gestion des mises en production ponctué d'étapes d’approbation manuelle claires.

DevOps selon Microsoft Azure

Fournir

- Elles définissent également des portes automatisées que franchissent les applications entre les étapes jusqu'à leur mise à la disposition des clients.
- L'automatisation de ces processus les rend évolutifs, reproductibles, contrôlés. Ainsi, les équipes qui utilisent DevOps peuvent procéder à des livraisons plus fréquentes avec facilité, confiance et tranquillité d'esprit.

DevOps selon Microsoft Azure

Opérer

- La phase d'exploitation implique la maintenance, la supervision et le dépannage des applications dans les environnements de production.
- En adoptant les pratiques DevOps, les équipes veillent à assurer la fiabilité, la haute disponibilité du système, et visent à éliminer les temps d'arrêt tout en renforçant la sécurité et la gouvernance..

DevOps selon Microsoft Azure

Opérer

- Les équipes DevOps entendent identifier les problèmes avant qu'ils n'affectent l'expérience client, et à les atténuer rapidement lorsqu'ils se surviennent.
- Maintenir cette vigilance nécessite une riche télémétrie, des alertes exploitables et une visibilité totale sur les applications et le système sous-jacent

DevOps selon Microsoft Azure

- Si l'adoption de pratiques DevOps automatise et optimise les processus grâce à la technologie, elle repose essentiellement sur la culture de l'organisation, et les personnes qui en font partie.
- Le défi consistant à entretenir une culture DevOps implique de profonds changements dans la manière dont chacun travaille et collabore.
- Cela étant, lorsque les entreprises s’engagent sur la voie d'une culture DevOps, elles peuvent créer un environnement propice au développement d’équipes hautement performantes.

DevOps selon Microsoft Azure

Collaboration, visibilité et alignement

- La collaboration entre équipes, qui commence par la visibilité, est l’une des caractéristiques d’une culture DevOps saine.
- Différentes équipes telles que le développement et les opérations informatiques doivent partager leurs processus, priorités et préoccupations DevOps.
- Ces équipes doivent également planifier leur collaboration et s'aligner sur les objectifs et les indicateurs de réussite liés à l'entreprise.

DevOps selon Microsoft Azure

Évolutions en termes de portée et de responsabilité

- Au fur et à mesure que les équipes s'alignent, elles s'approprient et s'impliquent dans d'autres phases du cycle de vie, sans se limiter aux seules phases inhérentes à leurs rôles.
- Par exemple, les développeurs, en plus de l'innovation et de la qualité établies lors de la phase de développement, sont en charge des performances et de la stabilité que leurs modifications apportent à la phase d'exploitation.
- De même, les opérateurs informatiques assurent la gouvernance, la sécurité et la conformité des phases de planification et de développement.

DevOps selon Microsoft Azure

Cycles de mise en production plus courts

- Les équipes DevOps restent agiles en publiant des logiciels dans des cycles courts. Les cycles de mise en production plus courts facilitent la planification et la gestion des risques car la progression est incrémentielle, ce qui réduit également tout impact sur la stabilité du système.
- En outre, le raccourcissement du cycle de mise en production permet aux organisations de s’adapter et de réagir aux besoins évolutifs des clients, de même qu'à la pression de la concurrence

DevOps selon Microsoft Azure

Apprentissage continu

- Les équipes DevOps hautement performantes adoptent un état d'esprit de croissance.
- Elles effectuent un Fail-fast et intègrent ce qu'elles apprennent dans leurs processus, s’améliorant continuellement, renforçant la satisfaction des clients, accélérant leur capacité à innover et à s'adapter au marché.
- DevOps est un cheminement qui fait la part belle à la croissance.

DevOps selon Microsoft Azure

Pratiques DevOps

- En plus d'établir une culture DevOps, les équipes donnent vie à DevOps en implémentant diverses pratiques tout au long du cycle de vie des applications.
- Certaines de ces pratiques permettent d’accélérer, d’automatiser et d’améliorer une phase spécifique.
- D'autres couvrent plusieurs phases, aidant les équipes à créer des processus homogènes qui contribuent à renforcer la productivité.

DevOps selon Microsoft Azure

Intégration et livraison continues (CI/CD)

- L’intégration continue est une pratique de développement logiciel dans laquelle les développeurs fusionnent fréquemment les modifications de code dans la branche de code principale.
- L’intégration continue utilise des tests automatisés, qui s’exécutent chaque fois qu’un nouveau code est validé afin d’assurer la stabilité du code de la branche principale.

DevOps selon Microsoft Azure

- La livraison continue correspond au déploiement fréquent et automatisé de nouvelles versions d'application dans un environnement de production.
- L'automatisation des étapes requises à des fins de déploiement permet aux équipes de limiter les problèmes susceptibles de survenir lors du déploiement et de procéder à des mises à jour plus fréquentes.

DevOps selon Microsoft Azure

- Une fois en place, ces deux pratiques ouvrent la voie au processus CI/CD, qui comprend l’automatisation complète de toutes les étapes entre la validation du code et le déploiement de production.
- L'implémentation du processus CI/CD permet aux équipes de se concentrer sur la création du code et élimine la surcharge et les erreurs humaines potentielles liées aux simples étapes manuelles.

DevOps selon Microsoft Azure

- En outre, le processus CI/CD permet de déployer du code de façon plus rapide et moins risquée.
- Les déploiements interviennent alors plus fréquemment et plus progressivement, ce qui renforce l'agilité et la productivité des équipes, de même que leur confiance quant à leur code d'exécution.

Intégration Continue (CI)

- Avec l'intégration continue, les développeurs fusionnent aussi souvent que possible leurs changements de code vers la branche principale.
- Dès qu'une modification est validée, des processus automatisés de build et de test sont exécutés pour valider l'exactitude de la modification. Les défauts sont détectés le plus tôt possible dans le cycle de développement – là où leur impact et leur coût sont les plus faibles : un « shift left » des défauts.
- L'automatisation des tests est la clé de l'intégration continue pour s'assurer que les nouveaux commits ne corrompent pas l'application lorsqu'ils sont intégrés dans la branche principale.

Déploiment Continu (CD)

- La livraison continue est une extension de CI, où le logiciel est conçu, configuré et paramétré de manière à ce qu'il puisse être mis en production automatiquement et à tout moment.
- Le déploiement continu va encore plus loin que la livraison continue, en orchestrant automatiquement le déploiement des applications au client à chaque changement.
- En accélérant la boucle de retours clients, les gains peuvent être énormes – en termes de qualité logicielle, de délais de projet, de résultats et de coûts de développement.

DevOps selon Microsoft Azure

- **Gestion de version**
- La gestion de version consiste à gérer le code dans les versions, à suivre les révisions et l'historique des modifications afin de faciliter l'examen et la récupération du code.
- Cette pratique est généralement implémentée à l'aide de systèmes de gestion de version tels que Git, qui permettent à plusieurs développeurs de collaborer sur la création du code.
- Ces systèmes proposent un processus clair pour fusionner les modifications de code intervenant dans les mêmes fichiers, gérer les conflits et restaurer les modifications apportées aux états antérieurs.

DevOps selon Microsoft Azure

- La gestion de version est une pratique DevOps fondamentale.
- Elle facilite la collaboration des équipes de développement, répartit les tâches de codage entre leurs membres et stocke l'intégralité du code pour simplifier sa récupération, si besoin.
- La gestion de version est également indissociable d'autres pratiques telles que l'intégration continue et l'infrastructure en tant que code.

AGILE

Agile software development

- Agile est une approche de développement logiciel qui met l'accent sur la collaboration d'équipe, les commentaires des clients et des utilisateurs, ainsi que la capacité à s'adapter aux changements moyennant des cycles de mise en production courts.
- Les équipes qui utilisent Agile apportent des modifications et des améliorations constantes aux clients, recueillent leurs commentaires, puis en tirent des enseignements et s'adaptent en fonction des attentes et des besoins de ces clients.

AGILE

- Agile diffère sensiblement des infrastructures traditionnelles telles que l'infrastructure en cascade, qui se caractérise par de longs cycles de mise en production définis par des phases séquentielles.

- Kanban et Scrum sont deux infrastructures très populaires associées à Agile.

Infrastructure as a code

- L’infrastructure en tant que code définit les ressources système et les topologies de manière descriptive, ce qui permet aux équipes de gérer ces ressources comme du code.

- Ces définitions peuvent également être stockées et versionnées dans des systèmes de gestion de version permettant leur révision et leur restauration, là encore comme du code.

Infrastructure as a code

- Utiliser l’infrastructure en tant que code aide les équipes à déployer des ressources système de manière fiable, reproductible et contrôlée.
- De plus, l'infrastructure en tant que code permet d’automatiser le déploiement et réduit le risque d’erreur humaine, notamment au sein d'environnements aussi vastes que complexes.

Infrastructure as a code

- Fiable et reproductible, cette solution de déploiement d'environnement offre aux équipes la possibilité de gérer des environnements de développement et de test identiques aux environnements de production.
- La duplication des environnements dans différents centres de données et plateformes cloud gagne également en simplicité et en efficacité.

Configuration Management

- La gestion de la configuration fait référence à la gestion de l'état des ressources d'un système, ce qui englobe les serveurs, machines virtuelles et bases de données.

- Moyennant des outils de gestion de la configuration, les équipes peuvent déployer les modifications de façon systématique et contrôlée, réduisant ainsi les risques de modification de la configuration système.

Configuration management

- Les équipes utilisent des outils de gestion de la configuration pour suivre l'état du système et éviter toute dérive de configuration, à savoir toute dérive de configuration d'une ressource système par rapport à son état souhaité au fil du temps.
- Combinées à l'infrastructure en tant que code, la définition et la configuration système sont faciles à modéliser et à automatiser, ce qui aide les équipes à exploiter des environnements complexes à grande échelle.

Supervision continue

- Une supervision continue offre une visibilité complète en temps réel sur les performances et l’intégrité de toute la pile d’applications, de l’infrastructure sous-jacente exécutant les applications aux composants logiciels de niveau supérieur.
- Cette visibilité comprend la collecte de données de télémétrie et de métadonnées, ainsi que la définition d'alertes correspondant à des conditions prédéfinies qui doivent susciter l'attention d'un opérateur.

Configuration management

- La télémétrie englobe les données d'événement et les journaux issus de différentes parties du système et stockés à un emplacement où ils peuvent être analysés et interrogés.
- Les équipes DevOps hautement performantes veillent à définir des alertes exploitables et explicites, et à collecter des données de télémétrie enrichies afin de mieux les exploiter.
- Fortes de ces informations, les équipes peuvent résoudre les problèmes en temps réel et améliorer les applications lors des cycles de développement ultérieurs.

Liens DevOps - Editeurs

- https://azure.microsoft.com/fr-fr/products/github/
- https://aws.amazon.com/fr/devops/

Augmenter la valeur Ajoutée Agile DevOps

Maximiser la valeur. se concentre sur la création de meilleurs logiciels et rapidement

- Développer un produit et livrer par incréments
- Des résultats de qualité alignés sur les besoins de l'entreprise
- Flexible et réactif aux changements

DevOps se concentre sur les processus de développement logiciel

- Time to market
- Aligne l'objectif entre le développement, les opérations et les moteurs commerciaux.
- Consolide le pipeline de construction et d'intégration pour obtenir un gain de productivité (cloud et plateforme agonistique).

Objectifs Devops

- Collaboration entre équipes Dev & Ops
- Suivre un cycle de vie de A à Z
- Mesurer les anomalies
- Gérer les anomalies en temps réel
- Détecter plus facilement des pannes
- Utiliser une méthodolgie qui optimise les performances IT & métier

Structure Equipe Devops

- Un responsable de projet DevOps
- Un gestionnaire de projet agile / coach agile
- Des développeurs et testeurs
- Un release manager : celui qui se charge des déploiements
- Un ingénieur en infrastructure
- Un ingénieur sécurité
- Un expert en industrialisation : celui qui automatise les déploiements…

Collaboration entre équipe

- Développer sa collaboration est clé dans un projet DevOps
- L'agilité inhérente au travail collaboratif permet de réagir immédiatement en cas de changement impromptu dans un projet (nouvelles tâches, modification du planning, changement dans l'équipe) et de s'affranchir des process lourds qui nuisent à l'avancement d'un projet.
- L'objectif étant que chaque membre de l'équipe puisse contribuer à la réalisation de l'objectif premier : créer et déployer les meilleurs produits possibles de manière optimale.

Mise en place d'un environnement automatisé

- L'approche DevOps entraîne la mise en place de processus standardisés et automatisés.
- L'intérêt est donc de pouvoir automatiser le maximum d'actions avec une chaîne d'outils adaptée : on parle d'**infrastructure as code**. L'objectif étant de pouvoir s'appuyer sur des **outils d'automatisation** permettant d'optimiser et simplifier la gestion et la mise en production des produits ou logiciels.
- Plusieurs **outils open source** permettent de créer et mettre en place un processus standardisé et automatisé : Orchestration Ansible, virtualisation Docker, Puppet, Gitlab ou encore Jenkins…

Gouvernance pour ajuster l'équilibre

- La gouvernance permet de trancher sur les projets
- Des méthodologies existent pour gouverner:
  - ITIL
  - ITSM
  - Lean Management
  - Agile

Gestion des version DevOps

- **DevOps** est axé sur les techniques de **gestion** de projet d'Agile et le support de microservices.

- **DevOps** approche le cycle de vie entier du développement de logiciel par une automatisation reposant sur les normes de contrôle de **version**.

- **Git** est la solution de contrôle de **version** la plus populaire dans **DevOps**, suivi de Subversion (SVN)

Contrôle des versions

- S’assurer que chaque commit du contrôle des versions déclenche la création automatisée de packages pouvant être déployés dans n'importe quel environnement, en utilisant *uniquement* les informations du contrôle des versions.
- Faire en sorte de créer des environnements de test de type production à la demande en utilisant uniquement les scripts et les informations de configuration du contrôle des versions, et des packages à l'aide du processus automatisé décrit dans l'approche précédente.
- Une infrastructure de test et de production de scripts permettant aux équipes d'ajouter de la capacité ou d'effectuer une reprise après sinistre de manière entièrement automatisée.

Conception du IAC

- IAC (Infrastructure as a code)
- Le développement, les tests et le déploiement d'applications dans un environnement de production requièrent souvent que les développeurs attendent la production, et vice versa.
- Si les développeurs peuvent demander des conteneurs ou des machines virtuelles avec le même niveau de stabilité que celui appliqué au code via une requête automatisée, il devient alors possible de rendre le déploiement plus fluide et rapide, en sachant que la configuration du réseau et de la machine virtuelle est effectuée via un système contrôlé.
- Le versionnement est amélioré et plus facile à tracer.

Configuration

- En mettant en place une automatisation de l'infrastructure, l'entreprise va largement réduire les écarts entre les délais de livraison des équipes de développement et le déploiement sur l'infrastructure par les équipes d'exploitation.
- Il s'agit ici d'automatiser la configuration des systèmes et d'automatiser leur livraison.
- Une infrastructure agile a également pour ambition de permettre aux développeurs d'avoir accès rapidement à un environnement propice au déploiement de leurs systèmes.
- Les configurations nécessaires sont mises en place.

Intégration continue

- L'intégration continue est un ensemble de pratiques consistant à tester de manière automatisée chaque révision de code avant de le déployer en production.
- Lorsque le développeur code une fonctionnalité, il conçoit également le test associé et ajoute le tout à son dépôt de code.
- Le serveur d'intégration va ensuite faire tourner tous les tests pour vérifier qu'aucune régression n'a été introduite dans le code source suite à cet ajout. Si un problème est identifié, le déploiement n'a pas lieu et les Dev sont notifiés. Si aucune erreur n'est remontée, le serveur d'intégration peut déployer directement le code en production.

Objectifs de l’intégration continue

- Le principal apport de l’intégration continue est de garantir en production un code de qualité, et donc unemeilleure satisfaction des utilisateurs finaux.
- En effet, l'automatisation des tests de tout le code source à chaque ajout/modification de fonctionnalités permet d’éviter l’introduction de régressions en production. Les développeurs peuvent configurer les notifications du serveur d’intégration et ainsi être prévenus sur le service de leur choix en cas d’anomalies (webhook, email, etc…), gagnant ainsi un temps précieux.

Objectifs intégration continue

- Ainsi, avec l'intégration continue la phase de tests automatisés est complètement intégrée au flux de déploiement.
- En réalité, ce n'est pas la totalité du code qui est testée, car le développement des tests prend du temps.
- Mieux vaut avoir les principales fonctionnalités mises à l'épreuves plutôt que toute l'application mal testée.

Objectifs intégration continue

- L'intégration continue permet également aux Dev d'avoir un retour plus rapide sur leur développement. Il n'y a donc plus besoin d'attendre plusieurs semaines pour identifier les erreurs et les corriger.
- Enfin l'intégration continue favorise le travail en équipe. Souvent, plusieurs Dev travaillent sur des tâches séparées dans le cadre d'un même projet. Or, plus les équipes sont importantes, plus le risque d'introduire des erreurs lors de l'intégration est élevé.

Objectifs Intégration continue

- Et en cas de problème, le débuggage peut s'avérer très chronophage.
- En intégrant les révisions de code quotidiennement, le risque d'erreur est réduit au minimum.
- Les conflits au sein de l'équipe se font plus rares et les Dev n'ont plus peur de "casser le code" à chaque déploiement.

Quelques Outils d'intégration continue

- Jenkins
- Puppet
- Ansible
- Gitlab CI
- TeamCity
- Travis CI

Livraison Continue

- L'intégration continue n'est pas un prérequis pour faire du DevOps, c'est plutôt un idéal à atteindre, un premier pas vers le déploiement continu (CD).

- Les tests automatisés constituent un mur de qualité, certes non exhaustif, mais qui filtre la plupart des régressions s'il est bien conçu. Dev et Ops sont ainsi plus sereins et plus aptes à déployer régulièrement.

Livraison Continue

- Le déploiement continu est une pratique qui consiste à livrer chaque modification apporté au logiciel directement aux utilisateurs finaux.
- Aucune intervention humaine n’est nécessaire, tout se fait automatiquement. Seuls les changements qui échouent à un test ne sont pas directement déployés en production.

Livraison continue

- Cette pratique permet d’accélérer la boucle de feedbacks, et permet aussi aux dev de se concentrer sur le développement de l’application puisqu’il n’y a plus de date de release à anticiper.
- - L'intégration continue est comme un mur de qualité que le code doit franchir systématiquement avant d'être déployé en production.
- - Cette pratique garantit un code de qualité, des utilisateurs finaux satisfaits et des développeurs plus efficaces.
- - De nombreux logiciels, parfois gratuits, permettent sa mise en place.
- - L'intégration continue est un bon premier pas vers le déploiement continu.

Automatisation des tests

- Le but de DevOps est de créer des applications plus rapides, efficaces et réactives en réunissant l'équipe de développement et l'équipe d'exploitation.
- Il s'agit d'un changement culturel pour supprimer toutes les barrières entre Dev et Ops et fournir des livraisons de logiciels plus courtes et plus fréquentes, permettant ainsi aux organisations de répondre de manière très agile face aux demandes et aux attentes des clients en constante évolution.

Automatisation des tests

- Les équipes de test doivent aligner leur conception de test, leur automatisation ainsi que le développement de cas de test avec DevOps pour garantir que les modifications fréquentes apportées n'ont pas affecté le produit final.
- Contrairement à l'approche traditionnelle, les tests d'automatisation dans un environnement DevOps nécessitent de déplacer les scripts d'automatisation des tests vers un outil de contrôle d'une version d'entreprise.
- Ce système de tests centralisés au niveau de l'entreprise se traduit par une suite de tests intégrée qui offre une exécution et des rapports centralisés.

Automatisation des tests

- **New Relic** – New Relic offre une visibilité de bout en bout ainsi qu'une expérience client améliorée et une infrastructure dynamique. Cela permet également à l'équipe DevOps d'économiser leur temps consacré à la surveillance des applications.
- **Jenkins** – Jenkins est un outil d'automatisation DevOps utilisé pour vérifier l'exécution des tâches redondantes. Jenkins est un serveur open-source **CI / CD** (intégration continue / livraison continue) qui permet aux utilisateurs d'automatiser diverses phases impliquées dans le pipeline de livraison d'applications.

Automatisation des tests

- **Splunk** – Cet outil d'automatisation est utilisé pour accéder aux données de la machine. Il offre une efficacité opérationnelle à la fois au développement et aux équipes opérationnelles de DevOps. Il offre aux entreprises la possibilité d'être plus productives, compétitives, sûres et fiables.
- **Selenium** – Étant l'outil de test d'automatisation le plus populaire pour DevOps, Selenium est conçu pour répondre aux besoins spécifiques d'un large éventail de navigateurs différents. Il utilise des ressources moindres et prend en charge l'exécution de tests parallèles, ce qui réduit le temps global requis pour le processus de test. Les scénarios de test préparés peuvent également être exécutés sur n'importe quel système d'exploitation.

Autres Outils automatisation TESTS

- Cucumber
- Jasmine
- Junit
- JMeter
- Sont d'autres outils d'automatisation de test populaires utilisés par DevOps pour accélérer le développement et le déploiement d'applications.

Automatiser Test

- Dans le passé, le département QA était informé dès que le développement était terminé et qu'une nouvelle version était prête à être testée, que ce soit avec les méthodes en cascade ou Agile.
- L'équipe QA poursuivait ses tests de régression manuels, réalisait des tests basés sur les caractéristiques pour tous les nouveaux composants et ajoutait quelques tests exploratoires. Une fois les tests terminés, le build retournait en phase de développement pour être retravaillé, ou était publié par l'entreprise.

Automatisation des tests

- Cette approche n'est plus viable si elle est étendue à plusieurs équipes. La nécessité de constamment jongler avec des builds différents sur plusieurs environnements intermédiaires peut entraîner une certaine confusion.
- Par conséquent, les équipes QA doivent s'adapter et s'aligner davantage. Pour ce faire, voici comment procéder:

- Standardiser les environnements
- Automatiser les déploiments
- Automatiser les tests (y compris les tâches de pré-test et de post-test)
- Automatiser et regrouper les cas de test en ensembles efficaces (Smoke-Tests, suites de régression) pour obtenir une couverture de code de 100 %
- Réaliser des tests timeboxing fiables

Automatisation des tests

- Pour être clair, les équipes doivent automatiser autant que possible le processus de test afin de fonctionner automatiquement au besoin, de manière efficace et efficiente.
- L'automatisation réduit le travail manuel et rassemble les équipes QA et d'infrastructure informatique, notamment grâce à des outils spécifiques d'automatisation spécialisée et de CI/CD.
- En outre, un framework d'automatisation mature devient primordial pour que les équipes puissent rapidement écrire des scripts et ajouter de nouveaux cas de test.

Automatisation: Test manuel nécessaire

- Les cas de tests automatisés ont un réel défaut potentiel : ils sont écrits par des humains. Il ne s'agit pas d'être méchant, mais réaliste : un individu ne peut penser qu'à un nombre limité de scénarios.
- Et même si l'ensemble du code est couvert, les différentes manières de le traiter avec d'autres variables/données de test ou sur d'autres environnements de test peuvent le compromettre

Automatisation Manuelle des tests

- Afin de trouver le bon équilibre entre les tests manuels et l'automatisation, n'hésitez pas à utiliser des indicateurs de fonctionnalités.
- Ces indicateurs sont utilisés pour activer, désactiver ou masquer la fonctionnalité en cours de production. Ainsi, le code peut être envoyé en production et passer par tous les processus de test et de déploiement automatisés, garantissant un haut niveau de qualité.

Automatisation manuelle des tests

- Ensuite, lors de la production (ou en phase intermédiaire, etc.), les indicateurs de fonctionnalités peuvent être activés pour un certain pourcentage de la base utilisateur ou pour l'équipe QA, afin que des tests manuels soient mis en place.
- De cette façon, des informations supplémentaires peuvent être obtenues et communiquées à l'équipe de développement. La durée de ces tests n'a pas non plus d'incidence sur les nouveaux processus DevOps et constitue, pour cette raison, un réel avantage.

Finops: optimisation des coûts de l'infrastructure

- Le but de la culture FinOps est tout d'abord de permettre à une organisation de comprendre ses coûts liés au cloud.
- Cette compréhension va lui permettre à la fois de travailler à l'optimisation et à la réduction de ces coûts, ainsi que d'éclairer les décisions qui impliquent un compromis entre les opérations et la finance.

Finops: optimisation des coûts de l'infrastructure

- La phase d'information vise cinq objectifs : visibilité, allocation, analyse comparative, établissement d'un budget et prévision.
- L'utilisation du cloud induit des notions de flexibilité, de paiement à la demande qui complique l'utilisation d'outils financiers « classiques » : la culture FinOps essaye d'y remédier.
- Le contrôle des ressources et une surveillance précise via un système de tags ou de découpage par services permet d'avoir une vue au plus proche des opérations et de donner le plus haut niveau de visibilité aux parties prenantes business et financières.

Finops: optimisation des coûts de l'infrastructure

- Une fois que l'organisation dispose d'assez d'informations, elle peut définir des objectifs de réduction de ses coûts liés au cloud.
- Les pratiques FinOps permettent de mieux comprendre l'utilisation des ressources cloud de l'organisation, puis de réduire ses coûts en s'engageant à l'avance sur la réservation de ressources, plutôt que d'utiliser les ressources à la demande, plus onéreuses.
- Le diagnostic des pratiques de consommation et l'analyse comparative permettent aussi de déceler les secteurs dans lesquels l'impact d'un redimensionnement de l'infrastructure nécessaire peut être le plus fort.

Finops: optimisation des coûts de l’infrastructure

- La culture FinOps se base sur la communication transversale au sein d'une organisation, étayée par des informations factuelles précises.
- Le lancement de mesures d’optimisation chiffrées permet d’établir une évaluation continue des objectifs établis, sur la base des indicateurs remontés durant la phase d'information.
- Cette évaluation permet d’initier des discussions avec les acteurs de la finance, des opérations et de la gouvernance pour intégrer ces problématiques à la structure même de l’organisation, comme par exemple en créant un organe transverse dédié à ces problématiques (*Cloud Cost Center of Excellence - CCoE*).
- La culture FinOps se place alors à la convergence de tous ces acteurs, et permet d'aligner leurs objectifs.

Finops: optimisation des coûts de l'infrastructure

- Les outils cloud ont des structures de tarification complexes, centrées autour des deux variables de performance et de délai de disponibilité des ressources.
- Pour être capable de réduire ses coûts, il faut connaître ces structures de coût, et les croiser avec les besoins opérationnels (IaaS, PaaS).
- La mise en place de pratiques FinOps implique une expertise portant sur tous les services clouds correspondant aux besoins opérationnels.

Finops: optimisation des coûts de l'infrastructure

- Des bonnes pratiques décidées au niveau de l'entreprise permettent un premier pas dans la culture FinOps sans accroître les risques. En premier lieu, la rationalisation de l'utilisation des ressources (espace de stockage, délai et volume des sauvegardes, …) permet d'obtenir des premiers résultats.
- La pondération des différences de prix entre les cloud providers avec les besoins opérationnels et les risques posés par l'optimisation du code pour un cloud provider en particulier (*cloud vendor lock-in*) peut aussi représenter un facteur d'optimisation implémentable au niveau de l'organisation.

Finops: optimisation des coûts de l'infrastructure

- Les dépenses inutiles liées au cloud (*cloud waste*) s'élèveraient à 14 milliards de dollars en 2019. Ces dépenses peuvent être grandement réduites de deux façons, qui constituent le coeur des mesures opérationnelles FinOps :
- D'abord, s'assurant que la taille des ressources correspond aux besoins opérationnels. Une ressource prise une taille au-dessus du nécessaire coûte deux fois plus cher.
- Ensuite, en s'assurant que toutes les ressources provisionnées sont effectivement utilisées. Par exemple, des ressources qui seront utilisées uniquement en production, mais provisionnées aussi sur les environnements de développement et de staging.

Finops: optimisation des coûts de l'infrastructure

- Le fait d'avoir de la visibilité sur ses besoins permet d'utiliser des ressources réservées, qui peuvent coûter significativement moins cher (jusqu'à quatre fois moins cher – en théorie - chez certains cloud providers) que les ressources déployées à la demande.
- Cette partie peut être assistée par des outils de gestion des coûts sur le cloud (*Cloud cost management software*), qui peuvent être proposés par le cloud provider ou par des entreprises tierces.

Finops: optimisation des coûts de l'infrastructure

- Le développement d'applications serverless nécessite des compétences particulières, mais a un impact non négligeable sur le coût final de l'application.
- Garder en tête les enjeux du cloud lors du développement permet aussi d'intégrer des fonctions de gestion de l'allocation des ressources (*horizontal scaling*).

Haute Disponibilité

- Alignement de l’ensemble des équipes du système d’information.
- Amélioration de l’organisation et de la qualité des déploiements
- Tendre vers une approche d’industrialisation des process
- Travailler ensemble et faciliter les tâches pour arriver à une automatisation des processus de déploiement.
- Homogénéisation des configurations, versionning
- Monitoring des process
- IAAS PAAS ou open source

Questions

Merci