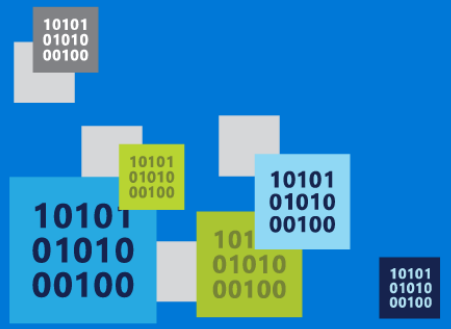
DevOps

Mouhanad EL FILALI
Directeur Conseils et Services @CGI
mhnd.elfilali@gmail.com



Qu'est-ce que DevOps ?

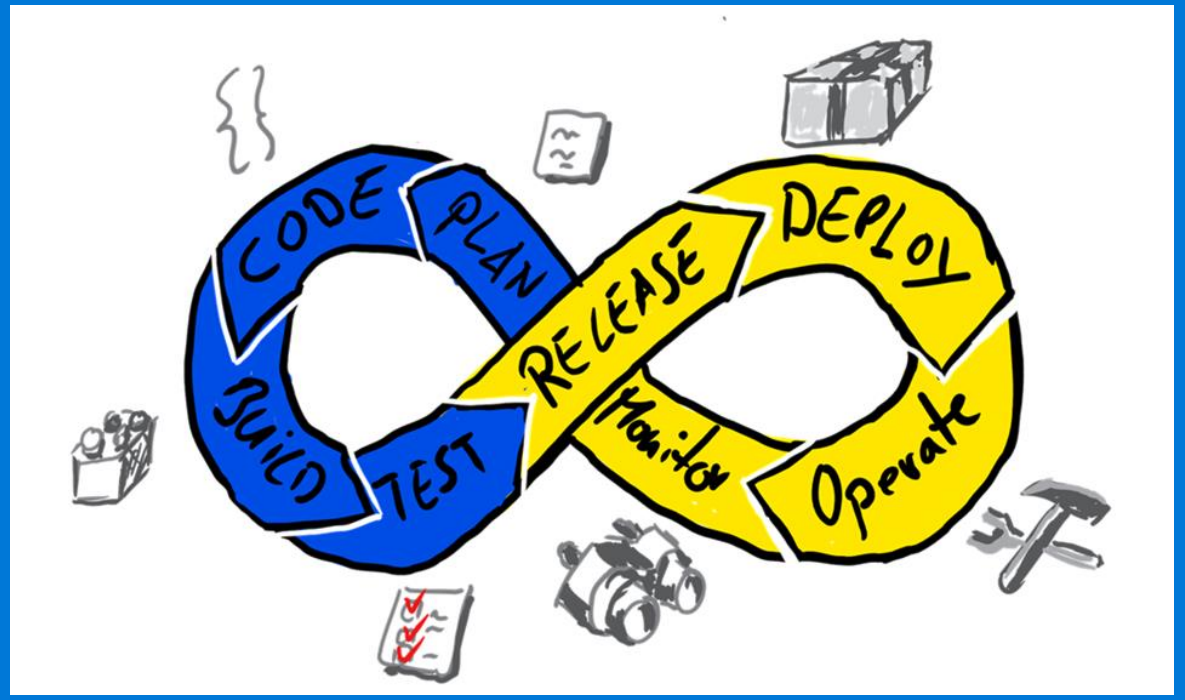
Avant de parler DevOps, connaissons-nous :

- Cycle de vie d’un logiciel
- L’agile
- ITIL

.....?!

Qu'est-ce que DevOps ?

- DevOps est l'union des personnes, des processus et des produits pour permettre une livraison continue de valeur à nos utilisateurs finaux.
- L'objectif est de faire un cloisonnement entre les équipes de développement « Dev » et les opérations « Ops » avec des pratiques et des outils partagés et efficaces.
- Les pratiques DevOps essentielles incluent la planification agile, l'intégration continue (CI : continuous integration), la livraison continue (CD : continuous delivery).

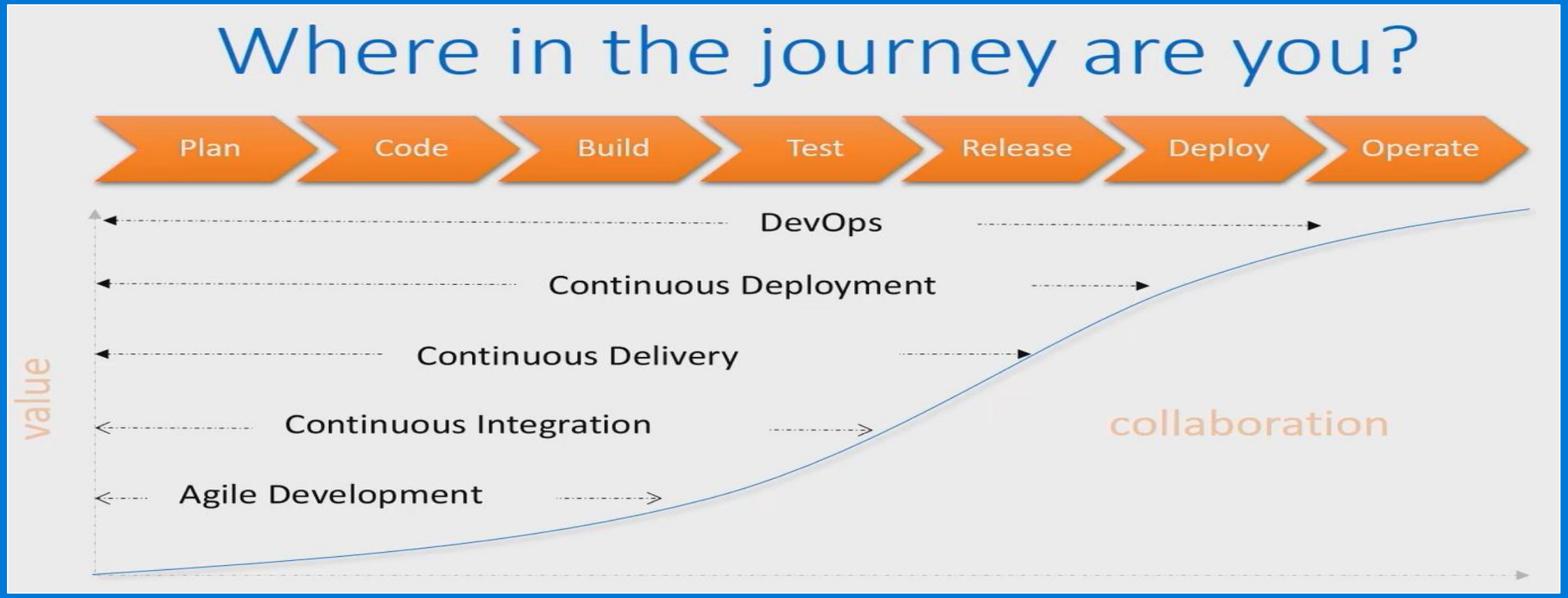


- DevOps vise à créer une culture et un environnement dans lesquels la conception, les tests et la diffusion de logiciels peuvent être réalisés rapidement, fréquemment et efficacement. DevOps n'est pas seulement une méthodologie, **c'est une véritable philosophie de travail.**

Qu'est-ce que DevOps ?

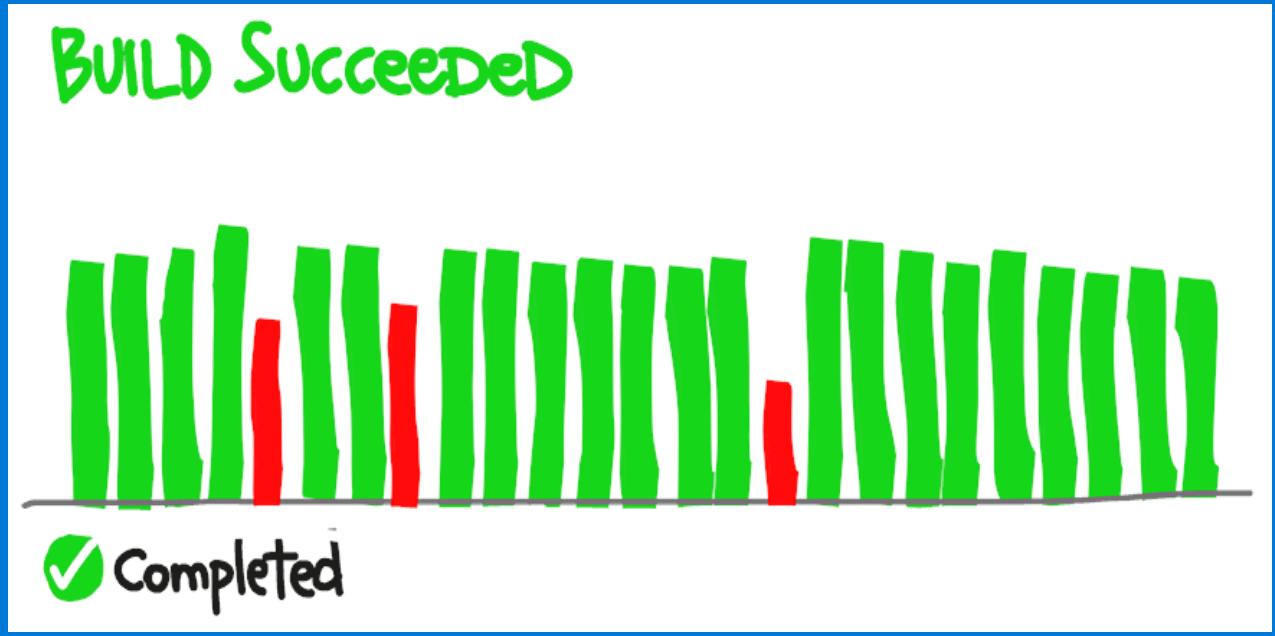
- DevOps est un moyen révolutionnaire de publier des logiciels rapidement et efficacement tout en maintenant un haut niveau de sécurité.
- Le contrôle des sources (contrôle de version) est un élément essentiel de DevOps.

Raccourcir le cycle de vie



L'Intégration Continue (Continuous Integration)

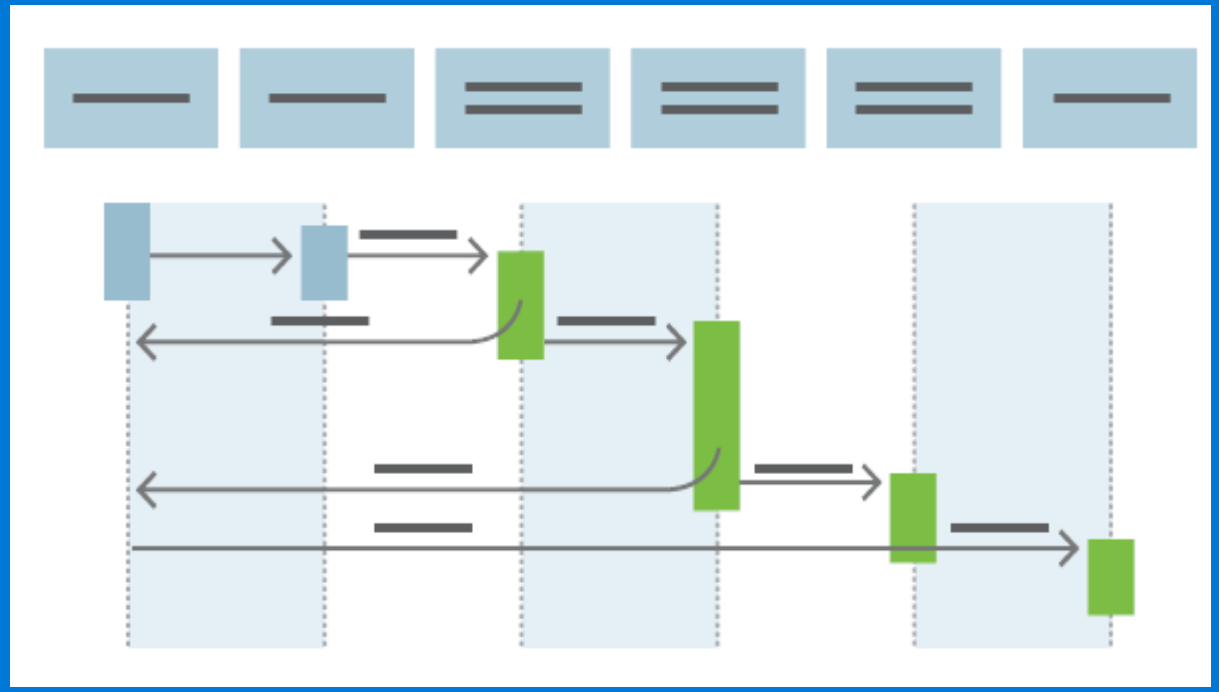
- Rappelez-vous, l'objectif de la CI est de réduire le cycle d'intégration de logiciel.
- Combien de temps vous faut-il pour déployer une modification d'une ligne de code ou d'une configuration ?



- CI entraîne la fusion et les tests continus du code, ce qui conduit à détecter les défauts à l'avance.
- CI offre moins de temps perdu à lutter contre les problèmes de fusion et une rétroaction rapide pour les équipes de développement.
- CI est le processus d'automatiser les tests et l'intégration du code à chaque commit du code vers un contrôleur de code source (TNR).
- CI encourage les développeurs à partager leur code et leurs tests unitaires en fusionnant leurs modifications dans contrôleur de code source partagé après chaque petite tâche terminée.

La Livraison Continue (Continuous Delivery)

- La livraison continue (CD) est le processus de développent, de test, de configuration et de déploiement d'une génération vers un environnement de production



- Plusieurs environnements de test ou de production créent un **pipeline de version** pour automatiser la création et le déploiement d'une nouvelle version.
- Les environnements successifs prennent en charge des activités progressivement plus longues d'intégration, de charge et de tests d'acceptation des utilisateurs.
- L'intégration continue démarre le processus du CD et le **pipeline** met en scène chaque environnement successif au suivant après la réussite des tests.
- L'objectif du CD est de maintenir la production à jour en réalisant le chemin le plus court depuis la disponibilité du nouveau code dans le contrôle de version jusqu'au déploiement.
- Le CD minimise le temps de déploiement et le temps d'atténuation ou le temps de correction des incidents de production.

Devops est un mouvement visant à simplifier et améliorer les relations entre les "devs" et les "ops" ...

Développement

(build = développeurs, testeurs, PO, SM)

Objectifs :

- plus de fonctionnalités
- réduire le TTM
- s'adapter au besoin utilisateur
- corriger les anomalies rapidement

Agilité

SCRUM

Intégration continue

Cycles courts

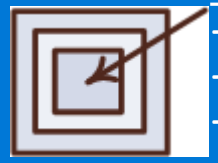
Tests auto

on veut **plus** de déploiements !

Objectifs opposés

Exploitant

(run = RM, ingénieur de production, admin système, DBA, admin réseau)



Objectifs :

- performance
- uptime
- sécurité
- Satisfaction client

Stabilité

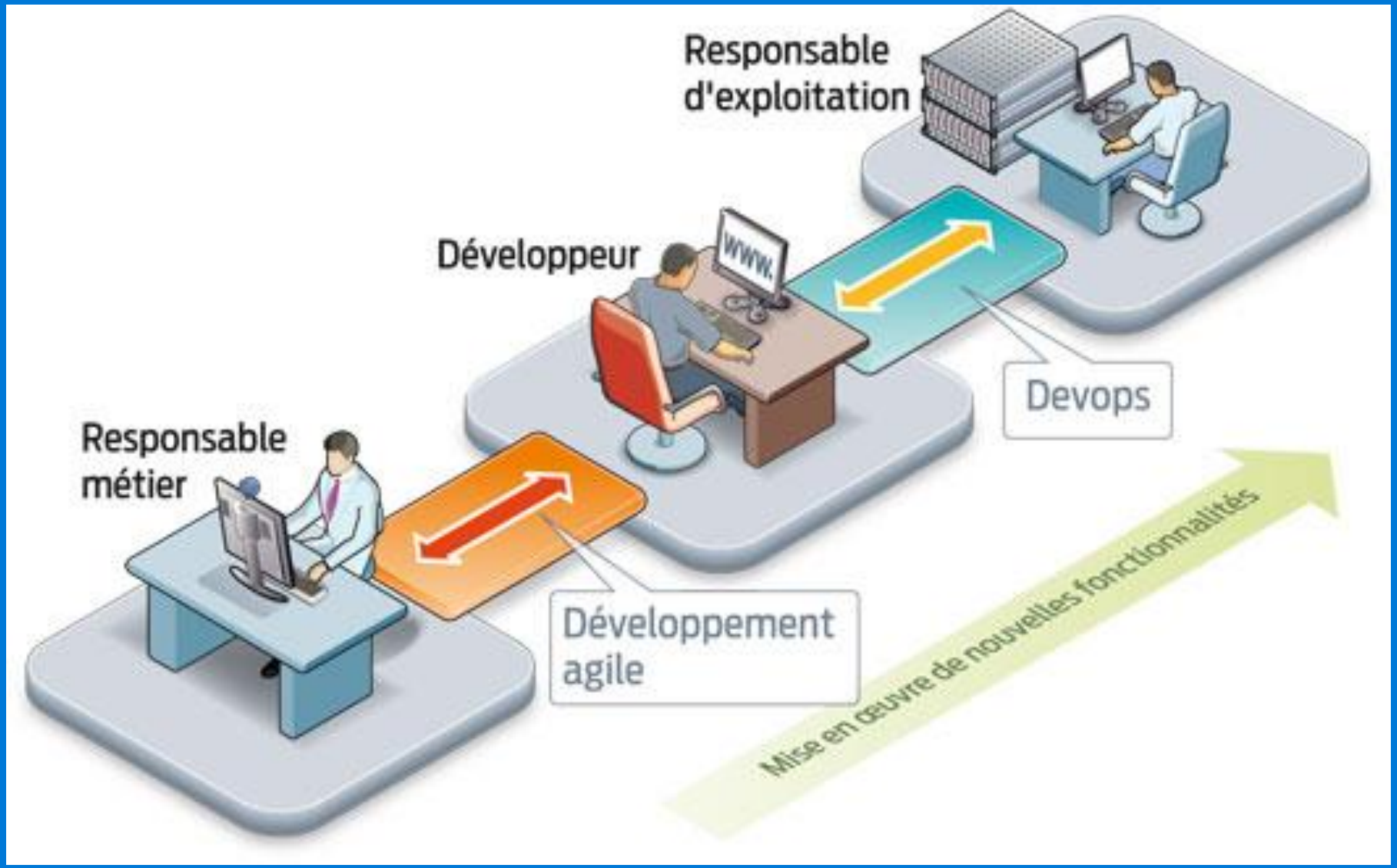
CHANGE = RISK

CAB

SD

on veut **moins** de déploiements !

… en mettant en œuvre un processus agile sur l'ensemble de la chaine du développement à la mise en production



DevOps : La définition

Développer
- l'intégration continue
- automatisation des tests
- livraison continue

=> **déploiement continu**

Diminution du temps de cycle, réduction des coûts de livraisons, réduction des erreurs, diminution du stress des équipe, processus de livraison fiable et rapide

AUTOMATISATION

CULTURE

Elaborer et partager une **culture commune entre le Build et le Run**
Basée sur la notion « d' équipe intégrée », l'échange, l'autonomie, l'excellence technique, l'amélioration continue, les outils agiles

Favoriser le mode projet, l'agilité, la créativité et la mise en avant des compétences de chaque membres des équipes

PARTAGE

Partager enjeux, contraintes, problèmes, infos, outils, vision d'entreprise, métier de l'autre, tableau de bord …

Etre tous orientés vers une même cible, cible partagée entre business , développement et exploitation. **Développer le « tous ensemble »**

MESURE

Définir et mettre en œuvre des **indicateurs de performances communs** : processus techniques, métiers, humain, Logs utiles et efficaces, mesurer les delta

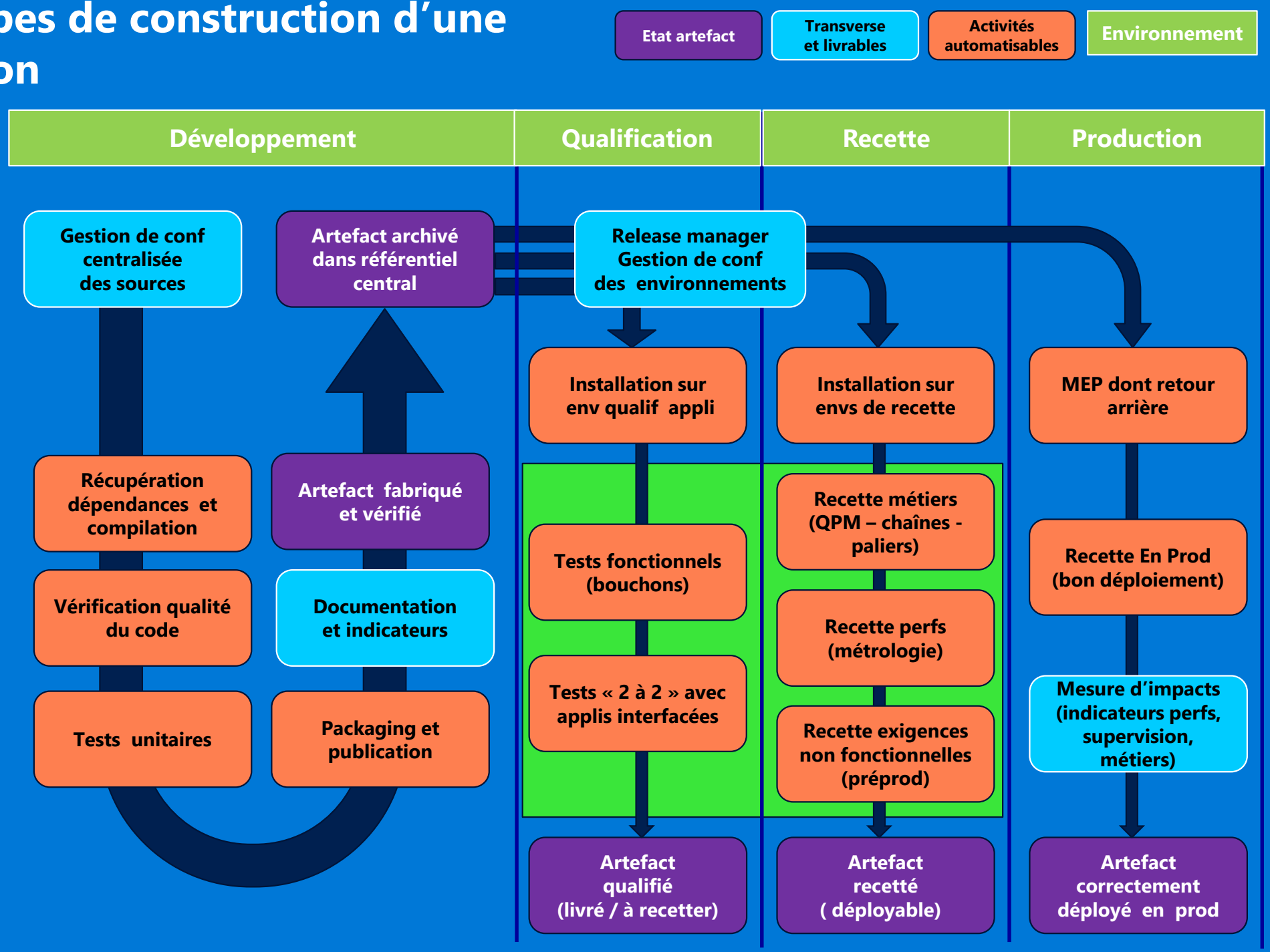
Détecter les incidents pro activement, anticiper pour mieux respecter coûts, délais, périmètre

Avantages à adopter l'approche DevOps

- ✓ Amélioration de la qualité du code, des produits et des services (réduction des anomalies, taux de réussite des changements plus important, etc.).
- ✓ Efficacité accrue (par exemple, optimisation du temps consacré aux activités qui créent de la valeur ajoutée).
- ✓ Amélioration du délai de mise en place sur le marché (Time To Market)
- ✓ Meilleur alignement entre l'informatique et les métiers.
- ✓ Des versions de plus petite taille fournies très rapidement et très fréquemment.
- ✓ Amélioration de la productivité, satisfaction du client, satisfaction du personnel.
- ✓ Moins de risques et moins de retours arrière.
- ✓ Réduction des coûts à long terme.

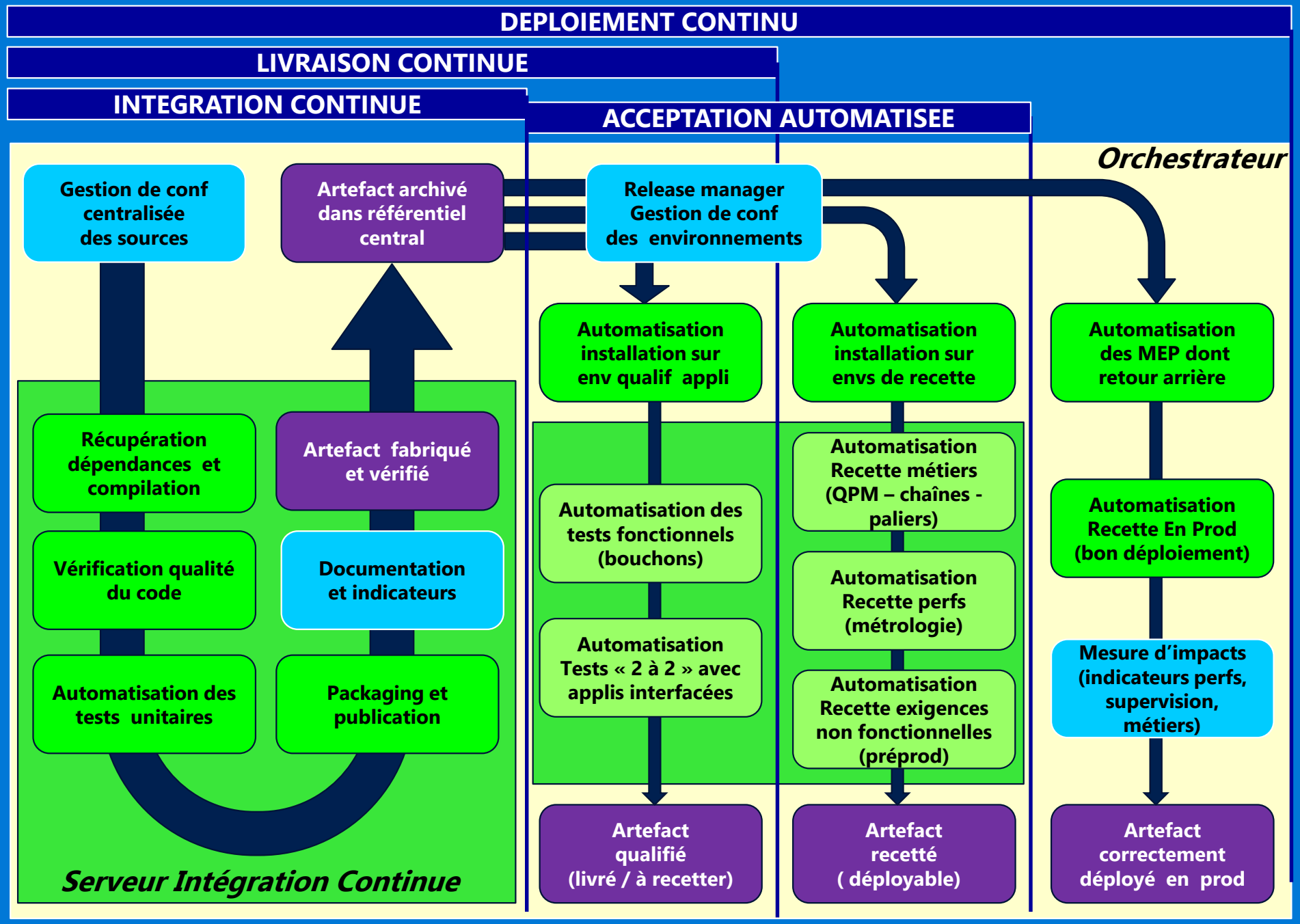
Automatisation

Les étapes de construction d'une évolution



Vers l'Automatisation

Etat artefact | Transverse et livrables | Activités automatisées | Mutualisables et contextuelles



Azure DevOps Services

Azure DevOps

- Azure DevOps fournit des services de développement pour aider les équipes à planifier le travail, à collaborer au développement de code et à créer et déployer des applications.
- Azure DevOps fournit des fonctionnalités intégrées auxquelles vous pouvez accéder via votre navigateur Web (azure.microsoft.com) ou votre client IDE (VS).

❖ **Azure Repos :** Pour le partage du code source.

❖ **Azure Boards :** Outils de la planification.

❖ **Azure Pipelines :** Workflow de build et déploiement.

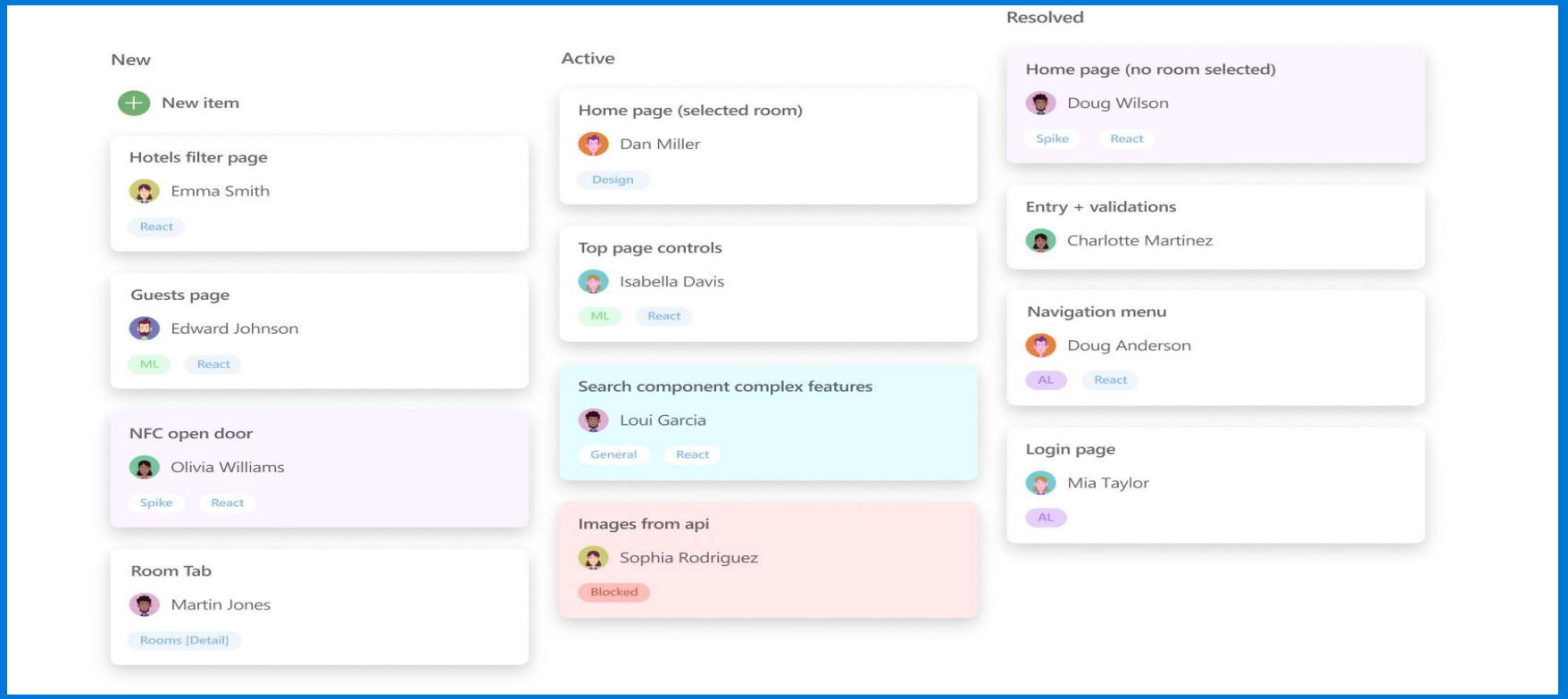
❖ **Azure Test Plans :** Automatisation et exécution des tests.

❖ **Azure Artifacts :** Création de package pour déploiement.

Azure DevOps

Azure Boards : Outils de planification Agile.

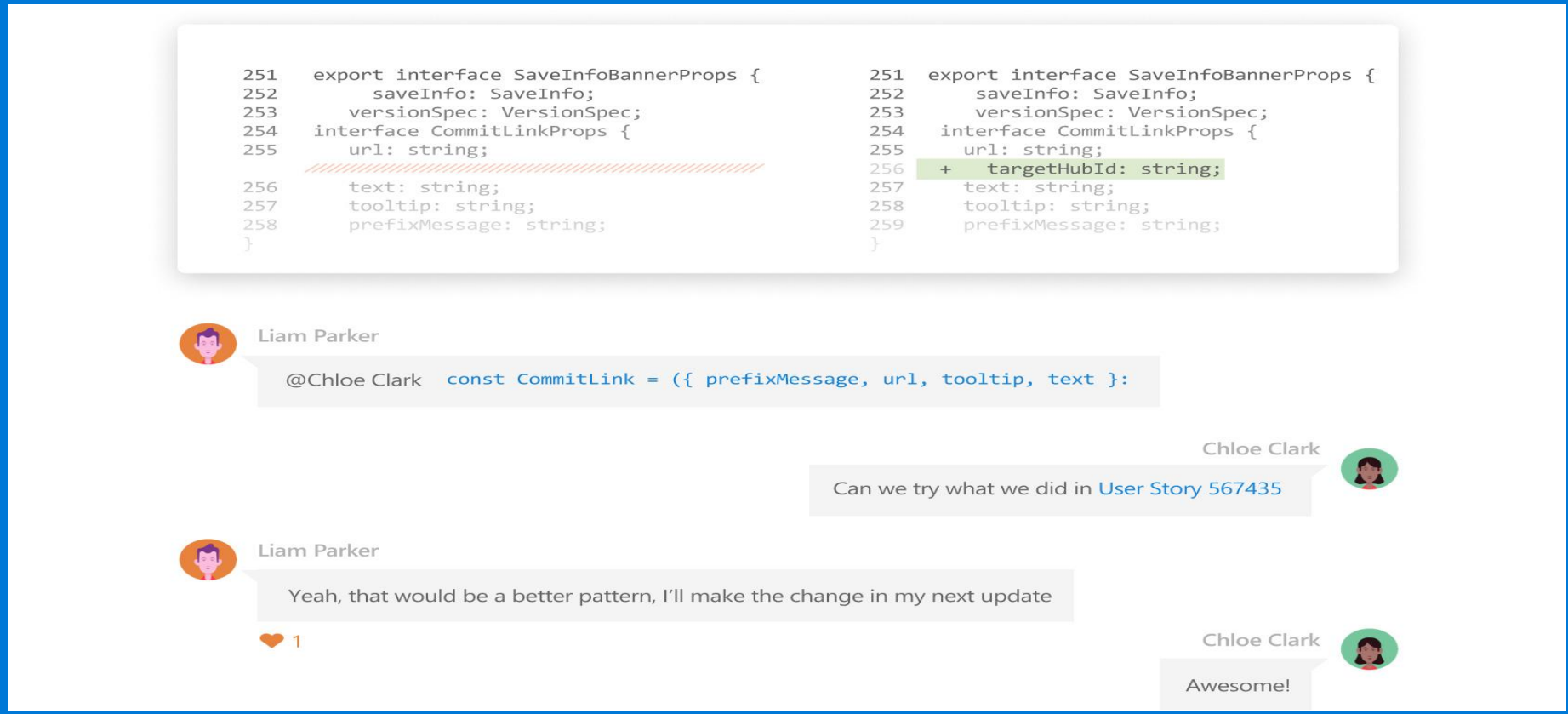
Suivez le travail effectué avec des tableaux Kanban configurables, des backlogs interactifs et des outils de planification puissants.
Grâce à une traçabilité et à des rapports inégalés, Boards est la solution idéale pour toutes vos idées, petites ou grandes.



Azure DevOps

Azure Repors : Dépôts privés gratuits et illimités.

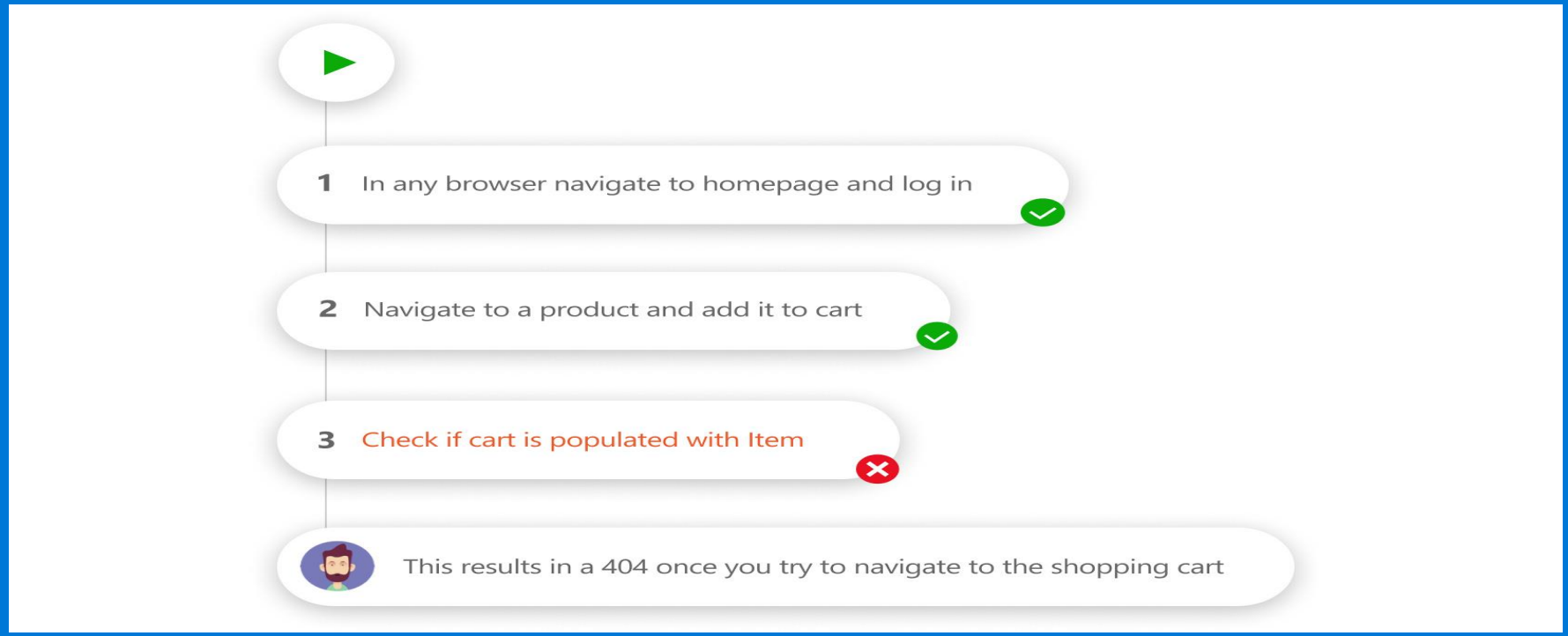
Bénéficiez d'un hébergement Git flexible et puissant avec des révisions de code efficaces et des dépôts gratuits illimités pour toutes vos idées, d'un projet individuel au plus grand dépôt du monde



Azure DevOps

Azure Test Plan : Tests exploratoires et manuels.

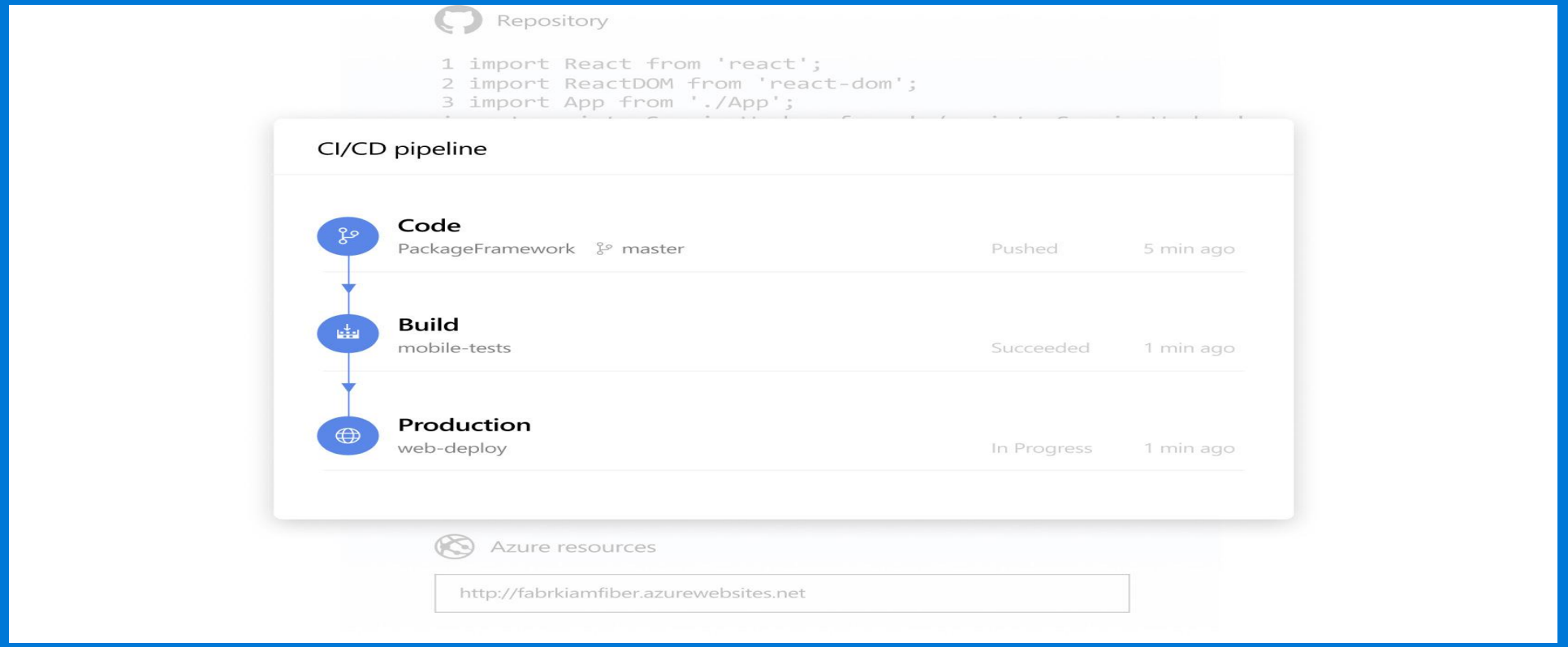
Testez souvent et publiez en toute confiance.
Améliorez la qualité globale de votre code avec des outils de tests manuels et exploratoires pour vos applications.



Azure DevOps

Azure Pipeline : CI/CD pour n'importe quelle plateforme.

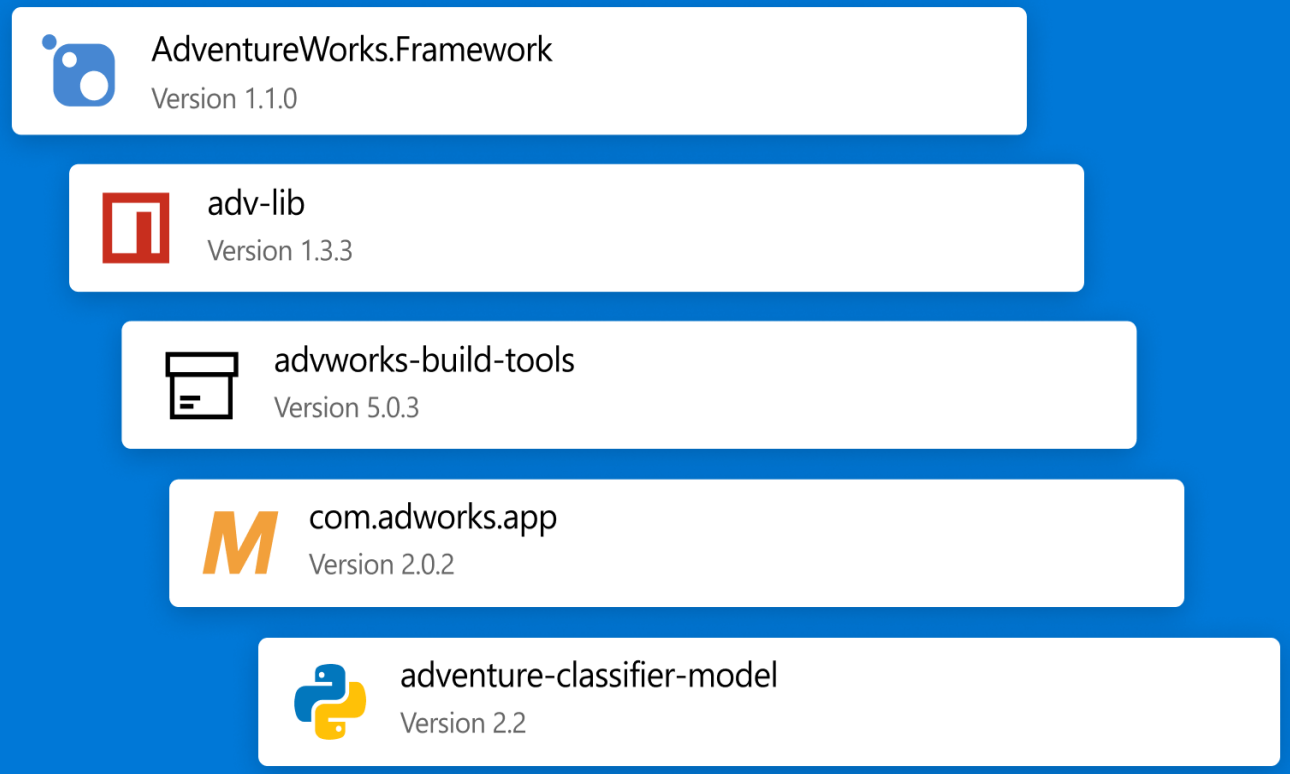
Générez, testez et déployez dans n'importe quel langage, sur le cloud de votre choix ou localement. Exécutez en parallèle sur Linux, macOS et Windows et déployez des conteneurs sur des hôtes.



Azure DevOps

Azure Artifacts : Dépôt de package universel.

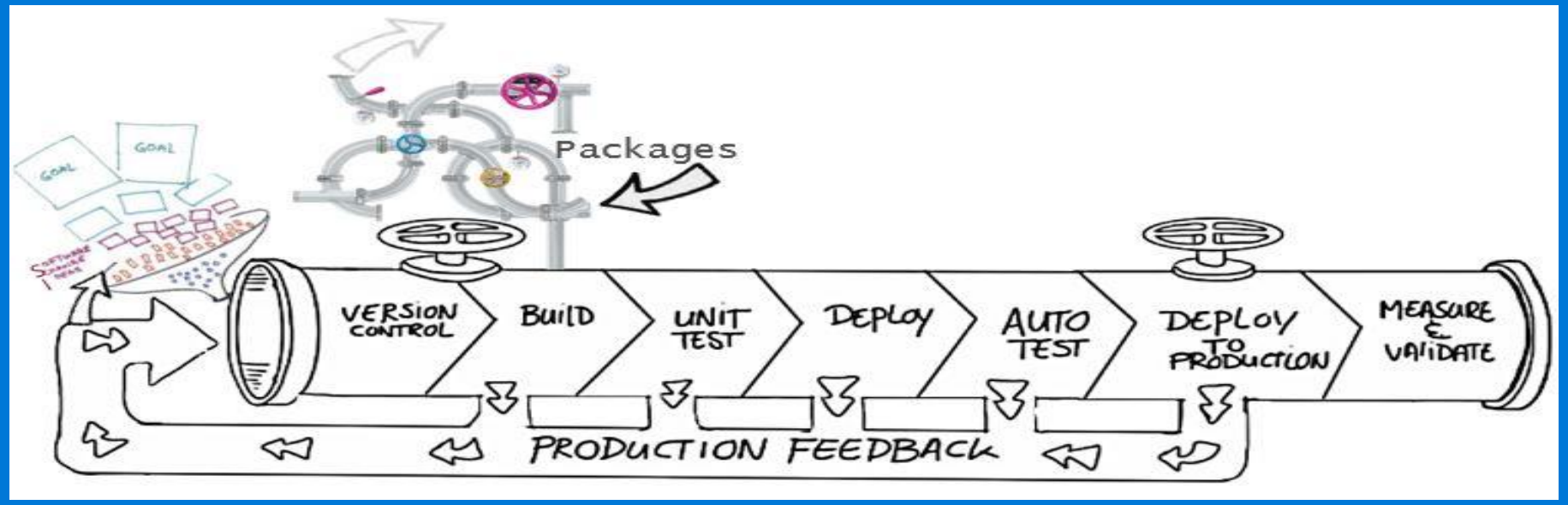
Partagez des packages Maven, npm, NuGet et Python provenant de sources publiques et privées avec toute votre équipe. Intégrez le partage de packages dans vos pipelines CI/CD d'une manière simple et scalable.



Contrôleur de code source (Source Contrôle)

Contrôleur de code source

- Source contrôle est la pratique **du suivi et de la gestion des modifications du code**
- Les sources contrôles fournissent un **historique** de développement du code et aident à résoudre les **conflits** lors de la fusion de contributions provenant de plusieurs sources
- Le contrôle des sources protège le code source des dégradations occasionnelles suite aux erreurs humaines.
- Les avantages incluent: la réutilisabilité, la traçabilité, portabilité, l'efficacité, la collaboration et l'apprentissage.



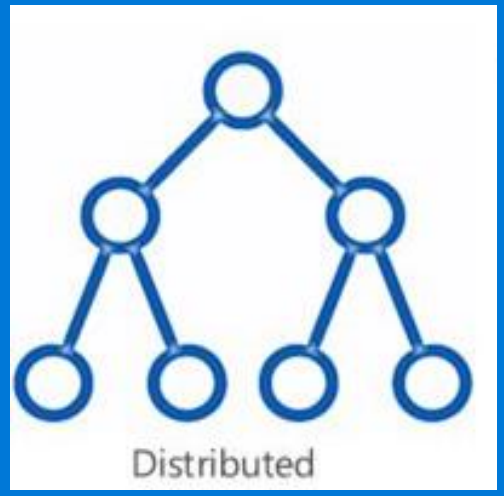
Contrôleur de code source

Bonnes pratiques pour le contrôle des sources :

- ✓ Faites de petits changements, avec des commit régulières.
- ✓ Ne pas commité des fichiers personnels.
- ✓ Mettre à jour souvent et juste avant de pousser pour éviter les conflits de fusion.
- ✓ Vérifiez votre changement de code avant de le pousser dans un référentiel, assurez-vous qu'il compile et que les tests réussissent.
- ✓ Portez une attention particulière à la validation des messages car ceux-ci vous indiqueront pourquoi un changement a été effectué.
- ✓ Lier les modifications de code aux éléments de travail (Work Item).

Contrôleur de code source

Contrôle de source distribuée :



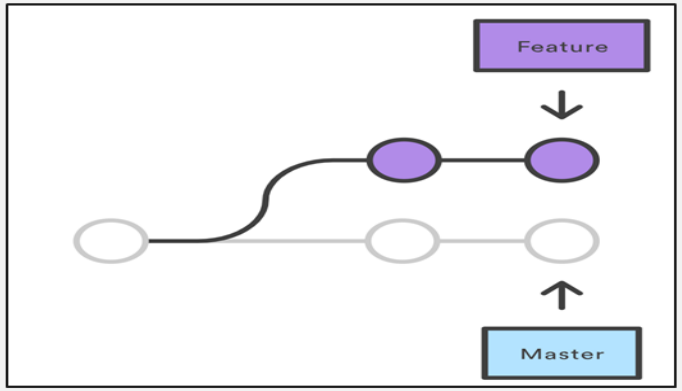
Forces	Idéal pour
• Prise en charge multiplateforme (CrossPlatform) • Un modèle de révision de code convivial open source via des demandes de tirage (Pull) • Historique • Demande de validation « Pull Request » • Gestion des branches.	• Petite modification de code. • Évoluer grâce à l'open source. • Des équipes réparties. • Des équipes travaillant sur toutes les plateformes.

- Chaque développeur clone une copie d'un référentiel et à l'historique complet du projet.
- Les systèmes de contrôle de source distribuée courants sont Git et Azure DevOps Serveur (TFS).

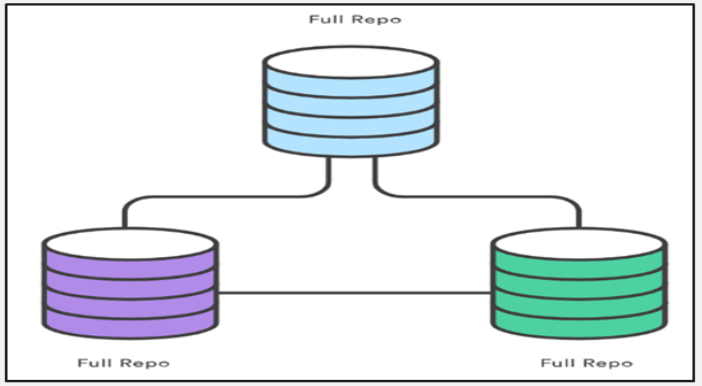
Contrôleur de code source

Why Git?

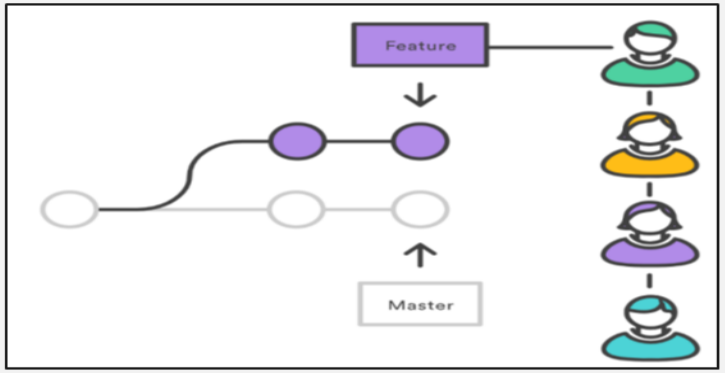
Feature branches



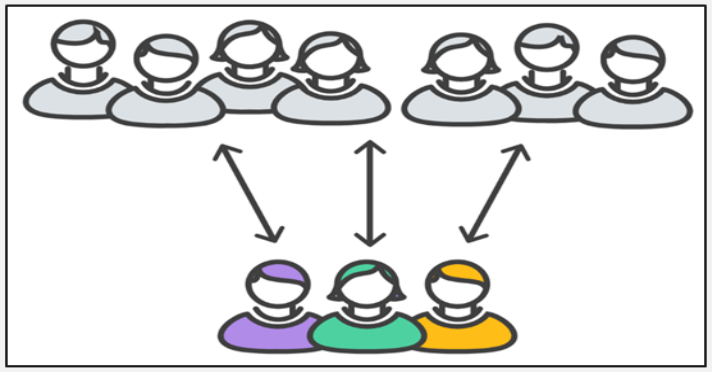
Distributed development



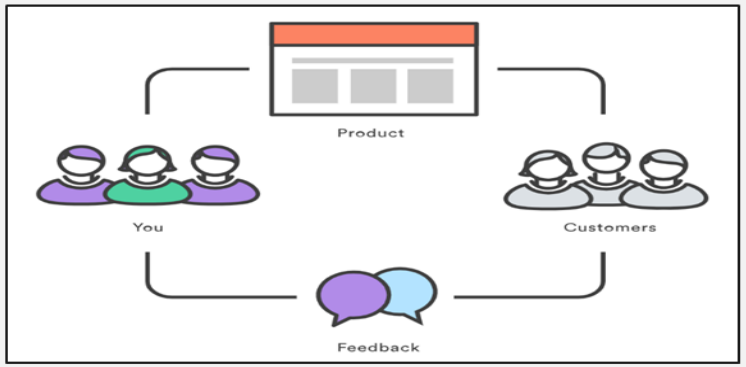
Pull requests



Community



Release cycles



Atelier Git