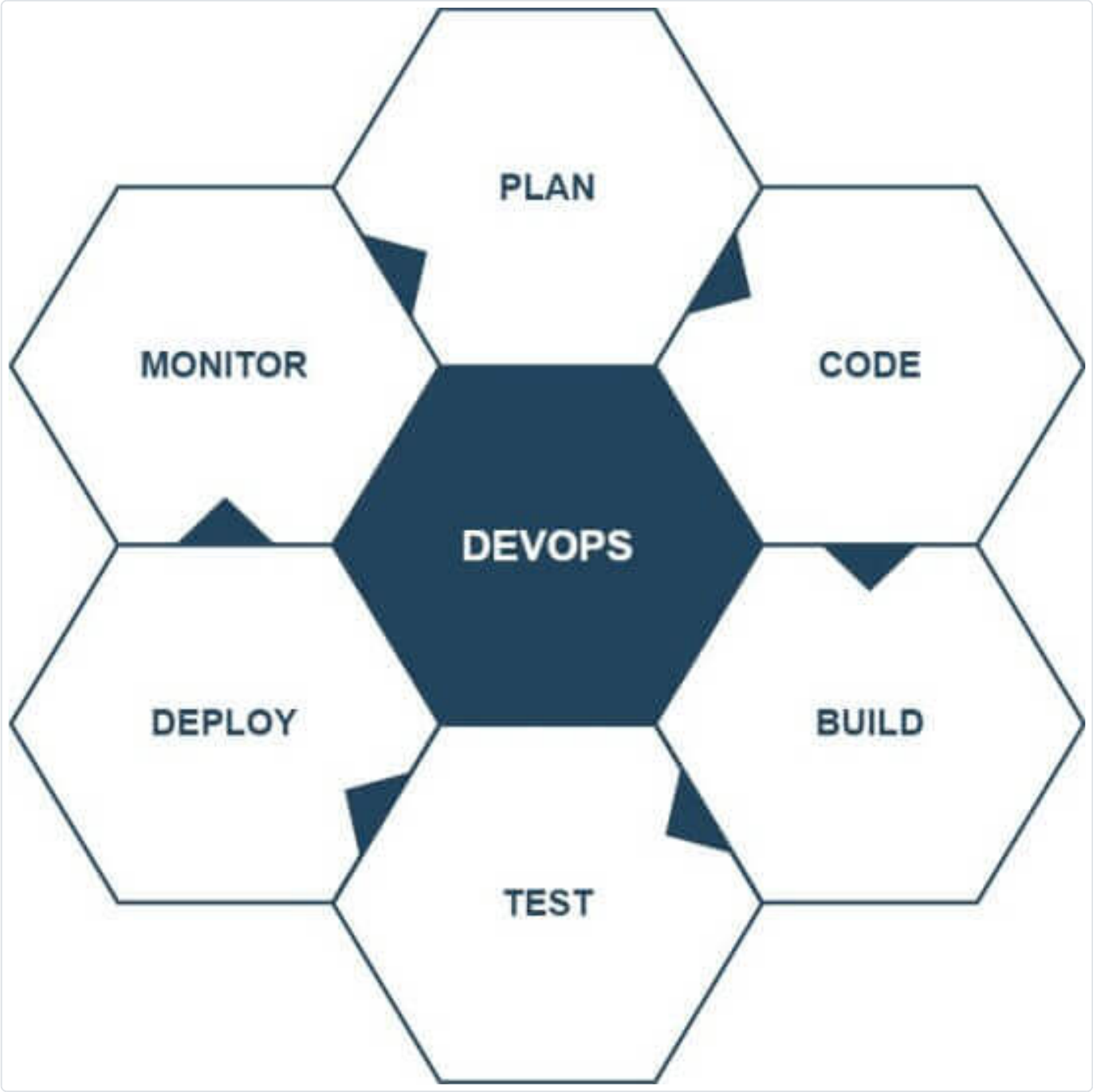
PIPELINE ET OUTILS DEVOPS

Pipeline Devops

Cette partie décompose **les différents composants et outils d'un exemple de pipeline*** DevOps. En regardant le cycle de vie d'une application qui se déroule tout au long du pipeline DevOps, nous pouvons le décomposer en différentes sections et concepts décrits sur le diagramme ci-dessous :



Pipeline Devops

Définition

***Pipeline** : voie de transmission d'informations de façon séquentielle.

Étapes et outils d'une pipeline Devops

Plan

Cette phase englobe tout ce qui se passe avant de commencer à écrire du code. Les besoins et exigences des parties prenantes et des clients sont recueillis et sont utilisés pour construire une feuille de route produite pour aiguiller le futur développement.

La feuille de route du produit peut être enregistrée et suivie à l'aide d'un **système de gestion des tickets et de suivi de projet** qui fournit une variété d'outils qui aident à suivre la progression, le planning, les problèmes et les tâches du projet tel que l'outil Jira.

De plus, c'est dans cette partie du projet où il faut également organiser les tâches et les plannings pour **aménager automatiquement le déploiement et la configuration de logiciels** grâce à des outils comme Ansible, (voir mon cours complet Ansible), Chef et Puppet etc … Et les infrastructures en faveur de l'IaC depuis des outils comme Terraform (voir mon cours complet Terraform).

Code

À ce stade, les développeurs écrivent le code du logiciel et le poussent dans un **référentiel de contrôle de source partagé** et lorsque le code est prêt, ils le fusionnent. Il existe plusieurs services d'hébergement de référentiel de code disponibles sur le marché (Github, Gitlab, etc …) ainsi qu'un système de contrôle de versions sous-jacent, le plus populaire reste git.

Build

C’est la première étape vers l'automatisation. Le développement du code est maintenant finalisé et l’équipe de développement utilise l’**intégration continue** depuis des outils comme Jenkins ou le Gitlab-CI, pour l'aider à compiler et à empaqueter automatiquement son code. Ce dernier sera alors prêt pour une future livraison à déployer dans les différents environnements, y compris celui de la production.

Test

Au cours de cette phase de Test, des tests continus des paquets créés précédemment sont exécutés automatiquement depuis le système d’intégration continu afin de réduire les risques de bogues et de failles.

Des tests, tels que des **tests système**, des **tests fonctionnels**, des **tests unitaires** (Ex: PHPUnit pour du code PHP, pytest pour du python etc ...) ou des **tests de vulnérabilités** (ex: SonarQube) peuvent être effectués sur la génération de la dernière phase. Si des problèmes sont détectés à cette phase, ces problèmes sont renvoyés à l’étape de code au développeur pour leurs résolutions afin d’assurer une qualité de code optimale.

Deploy

À ce stade, chaque changement de code a réussi la série de tests définis dans l’étape précédente. Intervient ensuite la phase de publication, qui est le moment où une version est prête à être déployée dans l'environnement de production.

À ce stade du pipeline et en fonction de la maturité du modèle DevOps dans l’entreprise, elle peut choisir d’effectuer soit du :

- **Déploiement continu** : déployer automatiquement en production le produit créé dans l'étape antérieure de build sans processus d'approbation manuel. C'est généralement grâce à ce processus que les organisations réussissent à déployer quotidiennement plusieurs versions de leurs produits.
- **Livraison continue** : ajouter un processus d'approbation manuel qui autorise uniquement certaines personnes au sein d'une organisation à déployer une version en production préalablement livrée dans un système de stockage dédié et centralisé (Ex : Nexus Repository Manager). Cela permet d'avoir un contrôle supplémentaire sur le produit avant son déploiement en production.

Monitor

La phase finale du cycle DevOps, consiste à **surveiller l'environnement de production** . Cette phase est gérée par un système de supervision capable de collecter et visualiser graphiquement les métriques de données, tout en fournissant des analyses sur le comportement, les performances et les erreurs des applications et de l'infrastructure. Cette phase peut être gérée depuis des outils comme Prometheus, la suite ELK, etc ...

Conclusion

Ces différents outils et phases, vous aideront à créer et automatiser votre pipeline DevOps afin de soulager vos équipes des tâches manuelles fastidieuses et répétitives. Ces outils peuvent également garantir que vos produits sont commercialisés de manière fiable et cohérente.