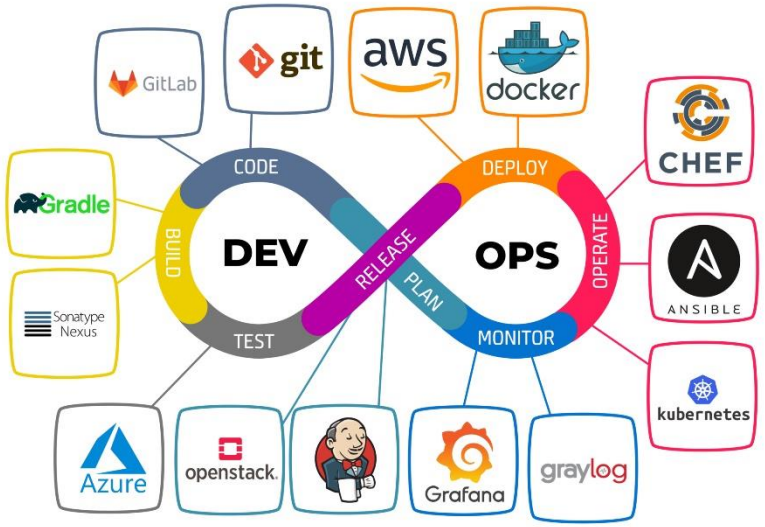
INITIATION AU DEVOPS

CODE :B3COM0105 **Durée** : 20H



Ing. BOGNI-DANCHI T.

OBJECTIFS PEDAGOGIQUES

Prérequis : ***Connaissances de base en administration système et développement logiciel***

Ce cours a pour objectif d'initier les étudiants aux concepts et pratiques de DevOps. Les étudiants apprendront à combiner le développement logiciel et l'administration système pour améliorer la collaboration entre les équipes et automatiser les processus de livraison continue. Le cours couvre les outils, les processus et les méthodologies utilisés dans un environnement DevOps.

À l'issue de ce cours, l'étudiant doit être capable de :

- Comprendre les principes fondamentaux de DevOps et leur rôle dans le développement logiciel.
- Utiliser des outils de gestion de versions, d'intégration continue et de déploiement continu.
- Automatiser les tests et les déploiements à l'aide d'outils comme Jenkins, Docker, et Kubernetes.
- Mettre en place des pipelines CI/CD (Continuous Integration/Continuous Delivery) pour optimiser les processus de développement.

PLAN

SECTION1: Introduction à DevOps

- Historique
- Définition et Contexte
- Principes Fondamentaux
- Cycle de vie du DEVOPS
- Les Outils du Devops
- TP : Préparation de l'environnement
  - ✓ Installation de WSL
  - ✓ Installation de la VM et du système

SECTION2: Le versioning

- Principes de Git
- Workflow de versionning
- Bonnes pratiques
- TP (3h)
  - ✓ Installation de GIT
  - ✓ Création de dépôts
  - ✓ Branches, merge, pull requests
  - ✓ Exercices sur GitHub/GitLab
  - ✓ Gestion collaborative de code source

PLAN

SECTION3: Conteneurisation

- Principes des conteneurs
- Docker : architecture et concepts
- Comparaison machines virtuelles vs conteneurs
- Travaux Pratiques (3h)
  - ✓ Installation de Docker
  - ✓ Création de Dockerfile
  - ✓ Construction et gestion d'images
  - ✓ Docker Compose
  - ✓ Déploiement d'applications conteneurisées

SECTION4: Intégration continue

- Gestion des erreurs et notifications
- Concepts d'intégration continue
- Outils : Jenkins, GitLab CI
- Principes des pipelines CI/CD
- Travaux Pratiques (3h)
  - ✓ Installation de Jenkins
  - ✓ Configuration de Jenkins
  - ✓ Création de premier pipeline
  - ✓ Tests automatiques
  - ✓ Déploiement automatisé

PLAN

- SECTION5: Orchestration et gestion des configuration
  - Architecture Kubernetes
  - Orchestration de conteneurs
  - Concepts : pods, deployments, services
  - Notion sur Ansible
  - Travaux Pratiques (3h)
    - ✓ Installation d'un cluster Kubernetes
    - ✓ Installation et configuration de Ansible
    - ✓ Déploiement d'applications
    - ✓ Mise à l'échelle
    - ✓ Gestion des mises à jour
    - ✓ Configuration de services
    - ✓ Gestion des configurations

- SECTION6: PROJET FINAL (Par l'étudiant)

SECTION 1 : INTRODUCTION AU DEVOPS

- 1- HISTORIQUE

2003 : Le SRE

En 2003, Google a embauché Ben Treynor pour diriger des ingénieurs dans un environnement de production, distinct de l'environnement de développement. Les nouveaux titres de poste de ces ingénieurs sont nommés **SRE (Site Reliability Engineering)**.
Leur principale tâche était de maintenir un temps de disponibilité élevé pour les services proposés par Google, le tout en travaillant en étroite collaboration avec les développeurs pour garantir que les opérations ne s'arrêtent pas pour les clients, car tout temps d'arrêt rencontré par Google dans ses principaux produits, entraîné des pertes de revenus. Bien que le terme n'ait pas encore été inventé, les SRE ont été les **premiers véritables praticiens d'une philosophie plus connue maintenant sous le nom "DevOps"**.

2007-2008 :**un Belge frustré**
Le consultant, chef de projet et praticien agile belge Patrick Debois, a accepté une mission auprès d'un ministère belge pour accompagner la migration de leurs centres de données où il était en charge des tests. Ses fonctions l'ont obligé à chevaucher les activités et les relations entre les équipes de développement d'applications et les équipes d'exploitation (serveur, réseau et base de données). Les expériences et les frustrations rencontrées sur le manque de cohésion entre les méthodes d'application et les méthodes d'infrastructure ont semé le mécontentement de Debois. Son désir d'une meilleure voie le conduira bientôt à l'action.

SECTION 1 : INTRODUCTION AU DEVOPS

1- HISTORIQUE

Plus tard, en 2008, lors d'une conférence Agile tenue à Toronto, au Canada, un homme du nom d'Andrew Shafer a tenté d'organiser une séance de rencontre intitulée « Infrastructure Agile ». Lorsque Patrick s'est présenté à la session il était le seul présent, mais Andrew avait reçu tellement de commentaires négatifs, qu'il n'était pas venu lui-même à sa propre session. Cependant, Patrick qui était présent pour animer une conférence nommée « l'utilisation du Scrum* et des pratiques agiles dans un contexte opérationnel », était tellement excité que quelqu'un d'autre partage sa frustration qu'il est allé retrouver Andrew dans le couloir de la conférence et ont eu une longue discussion afin de réaliser qu'il devait y avoir d'autres personnes qui voulaient échanger de ce qui semblait être un problème si répandu et systémique.

Octobre 2009 – aujourd'hui : La popularisation du mot « DevOps »

Ne pouvant pas assister en personne à la conférence organisée en Juin 2009 par deux employés de Flickr (John Allspaw, vice-président directeur des opérations techniques et Paul Hammond directeur de l'ingénierie de la société Flickr), Debois a décidé de la regarder par flux vidéo. Inspiré, il décide d'organiser sa propre conférence nommée « DevOpsDays ». En, cherchant à annoncer la conférence sur Twitter, il utilise le hashtag « DevOps » comme un raccourci des mots « développement » et « opérations » qui devient rapidement le nom du mouvement qu'Allspaw et Hammond ont articulé pour la première fois lors de leur conférence.

SECTION 1 : INTRODUCTION AU DEVOPS

2 - DÉFINITION ET CONTEXTE

2.1 Les défis traditionnels du développement logiciel : L'émergence de DevOps

1. Les silos entre développement et opérations

Dans le modèle traditionnel de développement logiciel, les équipes de développement (Dev) et les équipes opérationnelles (Ops) fonctionnaient de manière totalement cloisonnée. Les développeurs créaient des applications sans réelle compréhension des contraintes de déploiement et de maintenance, tandis que les équipes opérationnelles recevaient des livrables sans connaître les logiques de conception initiales.

Ces silos entraînaient plusieurs problèmes majeurs :

- Une incompréhension mutuelle des objectifs et des contraintes de chaque équipe
- Des transferts de responsabilités complexes et source de tensions
- Une perte significative de temps dans la communication et la résolution de problèmes

2. Lenteur des déploiements

Les processus de déploiement traditionnels étaient caractérisés par :

- Des cycles de livraison très longs, allant parfois de plusieurs mois à plus d'un an
- Des déploiements manuels, fastidieux et sources d'erreurs
- Une documentation souvent incomplète ou obsolète
- Une capacité limitée à réagir rapidement aux besoins du marché
- Ces lenteurs avaient des conséquences directes sur :
- La compétitivité des entreprises
- La satisfaction des utilisateurs
- La capacité d'innovation

SECTION 1 : INTRODUCTION AU DEVOPS

- 2 - DÉFINITION ET CONTEXTE

3. Manque de communication

Le manque de communication se manifestait par :

- Des objectifs divergents entre les équipes
- Une absence de vision partagée du produit
- Des malentendus sur les spécifications techniques
- Une difficulté à identifier et résoudre rapidement les problèmes

Cette communication déficiente générait :

- Des retards dans les projets
- Des développements non alignés avec les besoins réels
- Une démotivation des équipes

4.DevOps : Une solution holistique

DevOps est né précisément pour répondre à ces problématiques en proposant :

Principes fondamentaux

- Collaboration étroite entre développeurs et opérationnels
- Automatisation des processus de déploiement
- Intégration continue et déploiement continu (CI/CD)
- Culture de responsabilité partagée

SECTION 1 : INTRODUCTION AU DEVOPS

2 - DÉFINITION ET CONTEXTE

Bénéfices concrets

•Réduction des temps de déploiement
•Augmentation de la qualité logicielle
•Meilleure réactivité aux besoins métiers
•Amélioration de la satisfaction des équipes

2.2 Qu'est ce que l'approche DEVOPS

Le DevOps est une combinaison de pratiques, d'outils et de principes culturels visant à automatiser et intégrer les processus entre les équipes de **développement logiciel (Dev)** et d'**exploitation informatique (Ops)**. L'objectif principal est de raccourcir le cycle de développement des applications tout en garantissant une haute qualité des livrables.

3 - PRINCIPES FONDAMENTAUX

Les principes fondamentaux de DevOps :

1) Collaboration :

- Favorise une communication étroite entre les équipes de développement et d'exploitation pour réduire les silos organisationnels.
- La responsabilité est partagée du développement à la production

SECTION 1 : INTRODUCTION AU DEVOPS

3 - PRINCIPES FONDAMENTAUX

Éléments clés

- Communication transparente
- Respect mutuel
- Alignement des objectifs
- Élimination des barrières organisationnelles

2. Automatisation

•Automatise les processus répétitifs tels que les tests, l'intégration et les déploiements afin d'accélérer les livraisons et de réduire les erreurs humaines.

Domaines d'automatisation

- Intégration continue (CI)
- Déploiement continu (CD)
- Tests automatisés
- Provisionnement d'infrastructure
- Monitoring et gestion des configurations

3. Approche Lean et Agile

Principes

- Livraisons fréquentes et incrémentales
- Cycles de développement courts
- Feedback rapide
- Amélioration continue

SECTION 1 : INTRODUCTION AU DEVOPS

3 - PRINCIPES FONDAMENTAUX

Pratiques

- Méthodes agiles (Scrum, Kanban)
- Développement piloté par les tests (TDD)
- Gestion de projet flexible
- Adaptation permanente

4. Culture centrée sur la Qualité

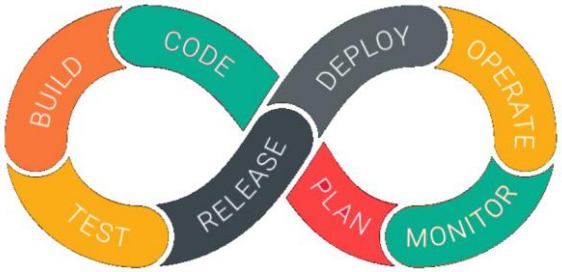
- **Tests automatisés** : Mise en œuvre de tests automatisés (unitaires, fonctionnels, de performance) tout au long du cycle de vie.
- **Détection précoce des erreurs** : Réduire les risques grâce à des tests fréquents et continus.
- **Infrastructure comme code (IaC)** : Configurer l'infrastructure de manière reproductible et versionnée (Ansible, Terraform).

5. Surveillance et Feedback Continu

- **Supervision en temps réel** : Surveillance des performances des applications et des infrastructures (métriques, logs, alertes).
- **Boucles de rétroaction** :
  - Collecter des retours des utilisateurs, des développeurs et des systèmes en temps réel.
  - Améliorer les systèmes de manière continue en fonction des retours.
- **Outils de monitoring** : Prometheus, Grafana, ELK Stack, Datadog.

SECTION 1 : INTRODUCTION AU DEVOPS

- 4 - CYCLE DE VIE DU DEVOPS



Étape 1) Développement continu
Cette pratique couvre les phases de planification et de codage du cycle de vie DevOps. Des mécanismes de contrôle de version peuvent être impliqués.

Étape 2) Intégration continue
Cette pratique de génie logiciel développe des logiciels en intégrant fréquemment ses composants. Cela permet de garantir que les modifications apportées au code source n'interrompent pas la construction ou ne provoquent pas d'autres problèmes.

Étape 3) Tests continus
Cette étape du cycle de vie DevOps intègre des tests de code automatisés, préprogrammés et continus au fur et à mesure que le code de l'application est écrit ou mis à jour. De tels tests peuvent être écrits manuellement ou en conjonction avec les outils de CI.

Étape 4) Déploiement continu
Le processus de déploiement se déroule en continu dans cette phase du cycle de vie DevOps. Ceci est effectué de manière à ce que les modifications apportées au code n'affectent pas le fonctionnement d'un site Web à fort trafic.

Étape 5) Surveillance continue
Au cours de cette phase, les développeurs collectent des données, surveillent chaque fonction et détectent les erreurs telles qu'une mémoire faible ou une connexion au serveur interrompue. Par exemple, lorsque les utilisateurs se connectent, ils doivent accéder à leur compte, et s'ils ne le font pas, cela signifie qu'il y a un problème avec votre application.

SECTION 1 : INTRODUCTION AU DEVOPS

- 4 - CYCLE DE VIE DU DEVOPS

Étape 6) Rétroaction continue

La rétroaction continue est comme un rapport d'avancement. Dans cette étape DevOps, le logiciel envoie automatiquement des informations sur les performances et les problèmes rencontrés par l'utilisateur final. C'est aussi l'occasion pour les clients de partager leurs expériences et de donner leur avis.

Étape 7) Continue Operations

Il s'agit de la dernière phase, la plus courte et la plus directe du processus. DevOps. Cela implique également d'automatiser la sortie de l'application et toutes ces mises à jour qui vous aident à garder des cycles courts et donnent aux développeurs plus de temps pour se concentrer sur le développement.

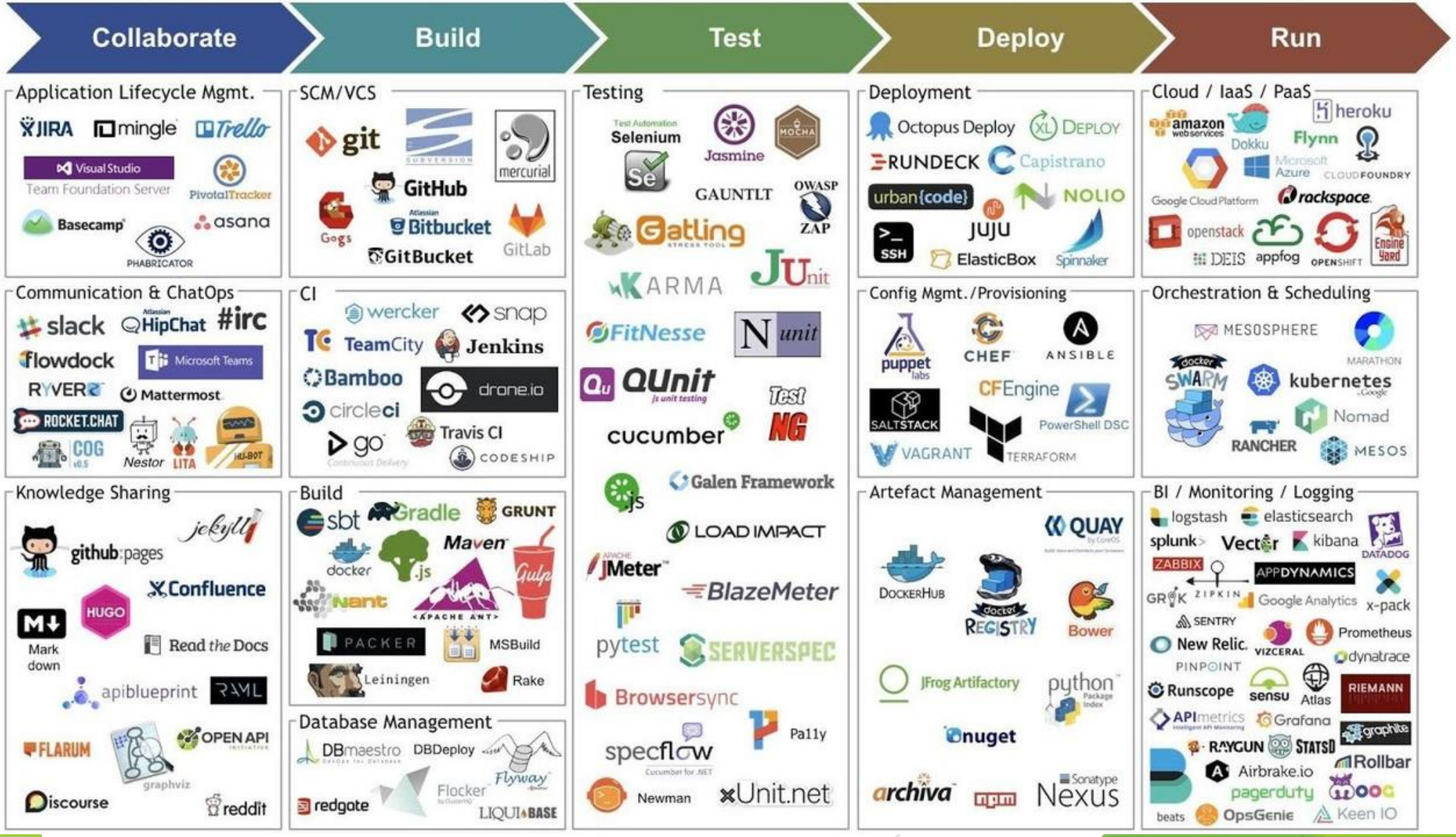
Avantages du cycle de vie DevOps

Voici quelques avantages essentiels du cycle de vie DevOps :

- Le cycle de vie DevOps est une approche utile qui guide les développeurs et les professionnels des opérations informatiques tout au long du processus complexe de création d'applications.
- Une meilleure efficacité conduit en effet à un retour sur investissement plus élevé.
- Largement utilisé par les grandes et petites équipes, il peut l'utiliser pour les aider à organiser, aligner et suivre les phases du cycle de vie.
- La surveillance, les tests et les versions automatiques aident les développeurs à détecter et à corriger les bogues à un stade précoce.
- Plusieurs méthodes automatisées de collecte de commentaires permettent aux développeurs DevOps d'en apprendre davantage sur leurs produits et d'améliorer la qualité de leur code.

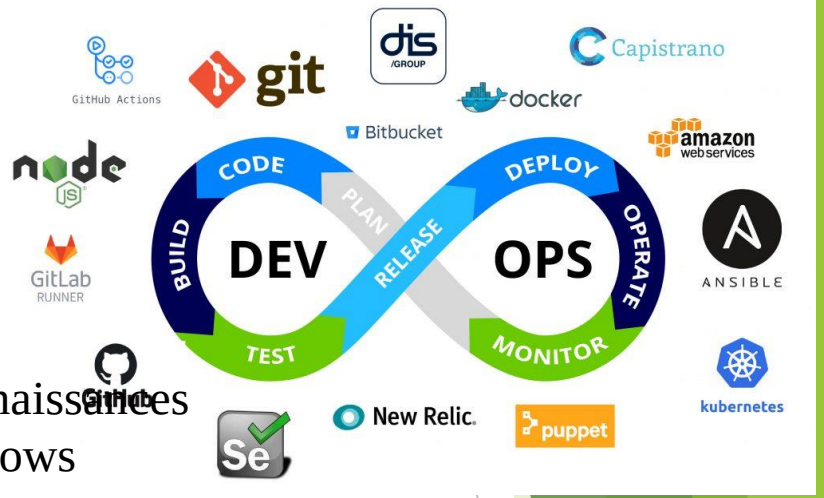
SECTION 1 : INTRODUCTION AU DEVOPS

- 5 – LES OUTILS DU DEVOPS



SECTION 1 : INTRODUCTION AU DEVOPS

5 – LES OUTILS DU DEVOPS



Les Outils DevOps par Cycle du DevOps

Planification

- **Jira** : Gestion de projet et suivi des tâches
- **Trello** : Tableaux collaboratifs et organisation
- **Confluence** : Documentation et partage de connaissances
- **Monday.com** : Planification et suivi des workflows

Développement

- **Git** : Contrôle de version et gestion du code source
- **GitHub** : Collaboration et hébergement de code
- **GitLab** : Intégration continue et gestion de dépôts
- **Bitbucket** : Gestion de code avec intégration Atlassian

Test

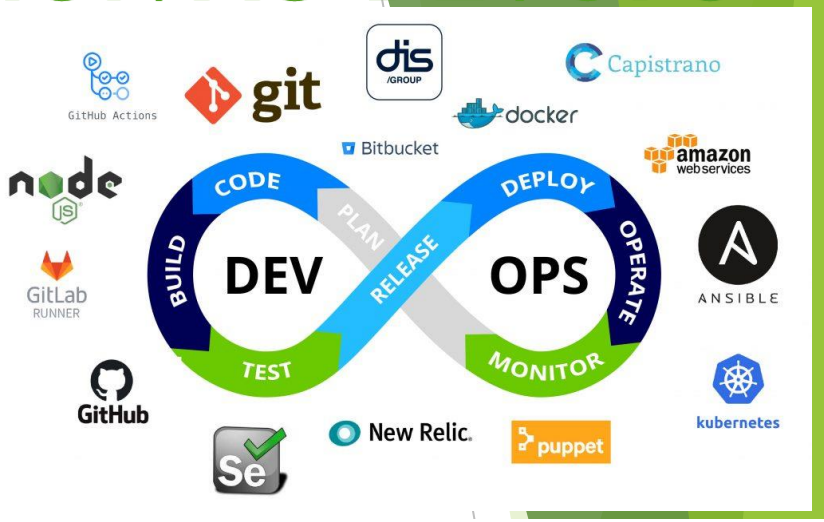
- Selenium : Tests automatisés d'applications web
- Jenkins : Intégration et déploiement continu
- SonarQube : Analyse qualité et sécurité du code
- JUnit : Tests unitaires pour applications Java

Intégration Continue

- Jenkins : Automatisation des builds et des tests
- Travis CI : Intégration et déploiement continu
- CircleCI : Pipeline de CI/CD cloud
- TeamCity : Serveur d'intégration continue

SECTION 1 : INTRODUCTION AU DEVOPS

5 – LES OUTILS DU DEVOPS



Déploiement

- Docker : Conteneurisation des applications
- Kubernetes : Orchestration de conteneurs
- Ansible : Automatisation du déploiement
- Terraform : Gestion de l'infrastructure as code

Exploitation

- Prometheus : Monitoring et collecte de métriques
- Grafana : Visualisation des données de performance
- ELK Stack : Analyse de logs et monitoring
- New Relic : Observabilité des applications

Surveillance

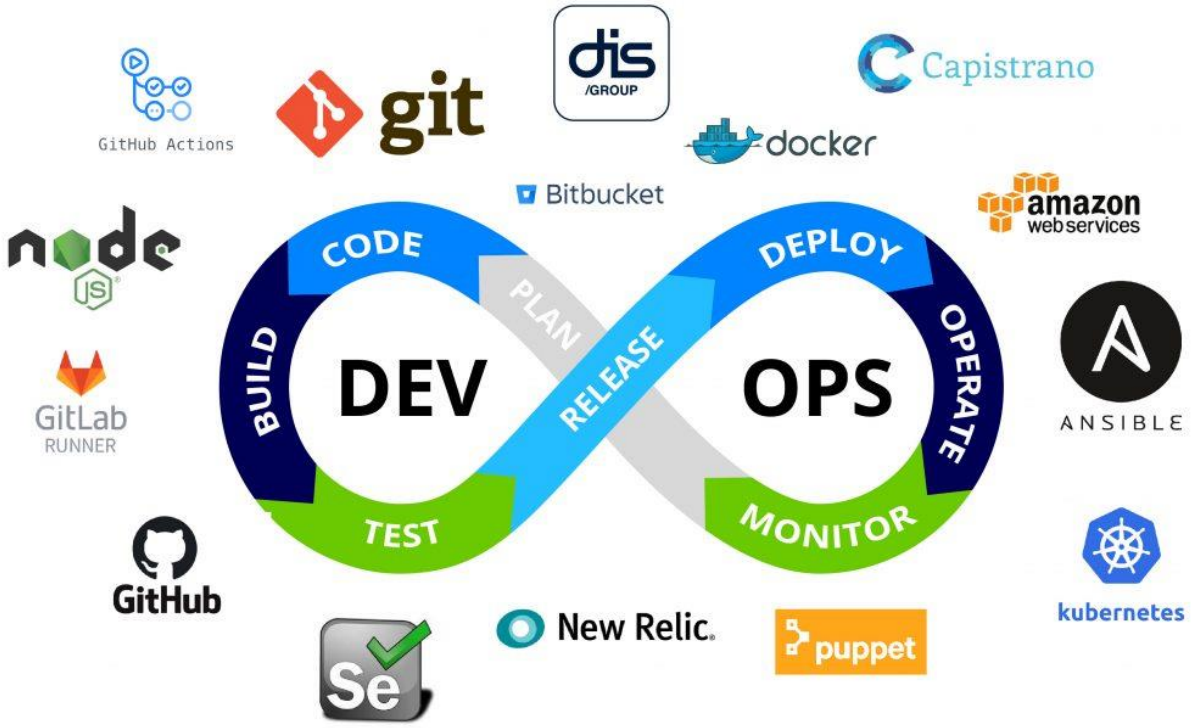
- Nagios : Supervision des systèmes et réseaux
- Datadog : Monitoring cloud et infrastructure
- Zabbix : Surveillance des performances IT
- AppDynamics : Monitoring applicatif

Étape	Outils courants
Planification	Jira, Trello, Azure DevOps, Asana, Confluence
Développement	Git, GitHub, GitLab, Bitbucket, Visual Studio Code
CI	Jenkins, GitHub Actions, GitLab CI/CD, CircleCI, Travis CI
CD	Ansible, Terraform, Kubernetes, Docker, Spinnaker
Surveillance	Prometheus, Grafana, ELK Stack, Datadog, New Relic
Feedback	Tableau, Google Analytics, Sentry, Splunk, SonarQube

SECTION 1 : INTRODUCTION AU DEVOPS

- 6 – TP

- **Préparation de l'environnement**
  - ✓ Installation de WSL
  - ✓ Installation de la VM et du système
  - ✓ Installation de Vscode



Fin

Par Thomas BOGNI-DANCHI