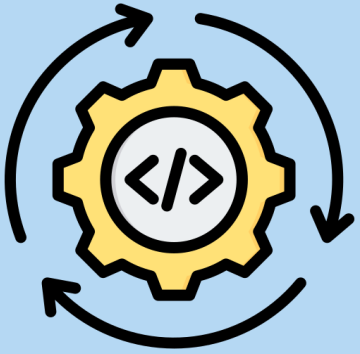


L'INTÉGRATION CONTINUE

INTRODUCTION

DÉFINITION

- **L'intégration continue** est un ensemble de pratiques utilisées en génie logiciel, consistant à vérifier, à chaque modification de code source que le résultat des modifications ne produit pas de régression dans l'application développée.
- Démarche qualité orientée industrialisation des développements

LES PHASES D'AUTOMATISATION

En pratique, on distingue 3 activités dans la livraison de logiciel automatisée :

- L'intégration continue
- La livraison continue
- Le déploiement continu

INTÉGRATION CONTINUE (CI)

Dans le cadre d’une intégration continue

- la phase de développement (mise à disposition et test du code) est entièrement automatisée
- chaque fois que vous poussez du code, les modifications sont validées et fusionnées dans la branche principale
- le processus d'intégration continue génère des artefacts de build qui contiennent le code ou les binaires résultants

INTÉGRATION CONTINUE : LES BONS INGRÉDIENTS

- Utiliser le contrôle de code source
- Valider tôt, valider souvent
- Compiler votre solution à chaque commit
- Automatiser les tests
- Écouter les retours
- Développer une culture DevOps

LIVRAISON CONTINUE

Dans le cadre d’une livraison continue

- On automatise la phase de livraison
- chaque fois qu’un nouvel artefact de build est disponible, l’artefact est automatiquement placé dans l’environnement souhaité.

LIVRAISON CONTINUE : LES BONS INGRÉDIENTS

- Build unique
- Séparer les préoccupations environnementales
- Stocker la configuration dans le contrôle de code source (pas forcément au même endroit que le code)
- Nettoyer vos environnements
- Maintenir le pipeline

DÉPLOIEMENT CONTINU (CD)

Dans le cadre d’un déploiement continu

- On automatise l’ensemble du processus, de la validation du code à la production
- L'étape entre les phases de développement et de livraison est automatique
- Les modifications de code sont donc transmises en direct une fois qu’elles reçoivent la validation et réussissent tous les tests.

DÉPLOIEMENT CONTINU : LES BONS INGRÉDIENTS

- Maitriser la livraison continue et l'exploiter
- Maitriser l'infrastructure
- Automatiser chaque déploiement
- Une bonne communication entre Dev et Ops
- Maitriser les modes de déploiement (Rolling upgrade, Blue-green , canary)

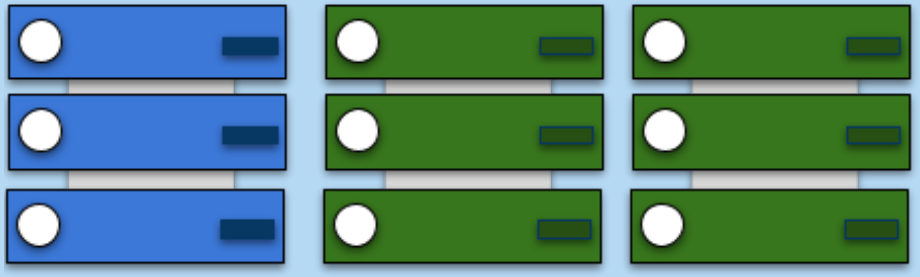
ROLLING UPGRADE

- La nouvelle version d’une application remplace progressivement l’ancienne
- Le déploiement réel se produit sur une période de temps

Etape initiale

Etape 1

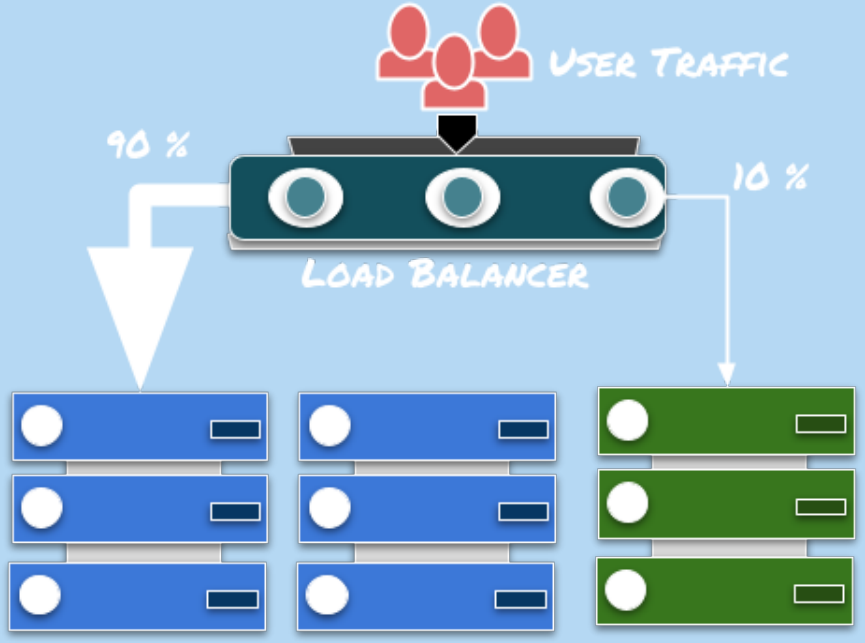
Etape 2



Etape finale

CANARY RELEASE

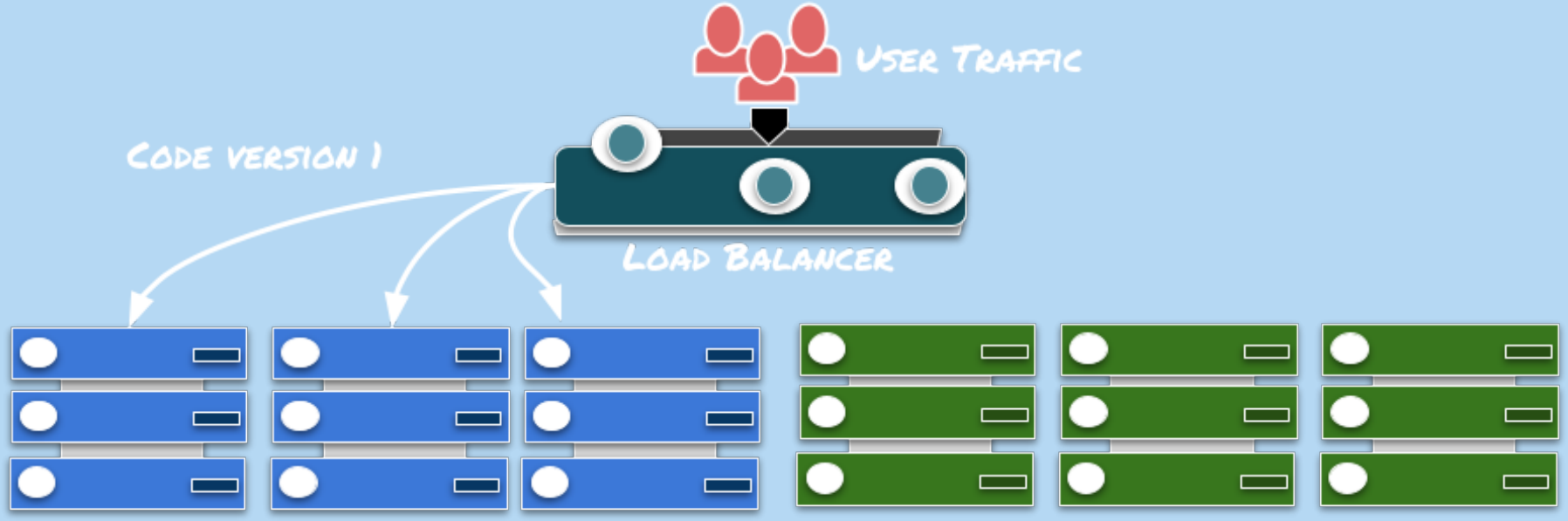
- **L'idée :** Rendre disponible une nouvelle version du logiciel à un petit sous-ensemble des utilisateurs (10%)
- Tester en conditions réelles et avoir un retour sur la nouvelle version
- Plusieurs nouvelles versions d’une fonctionnalité testées en parallèle
- On peut vouloir sélectionner certains "clients" pour tester une nouvelle version



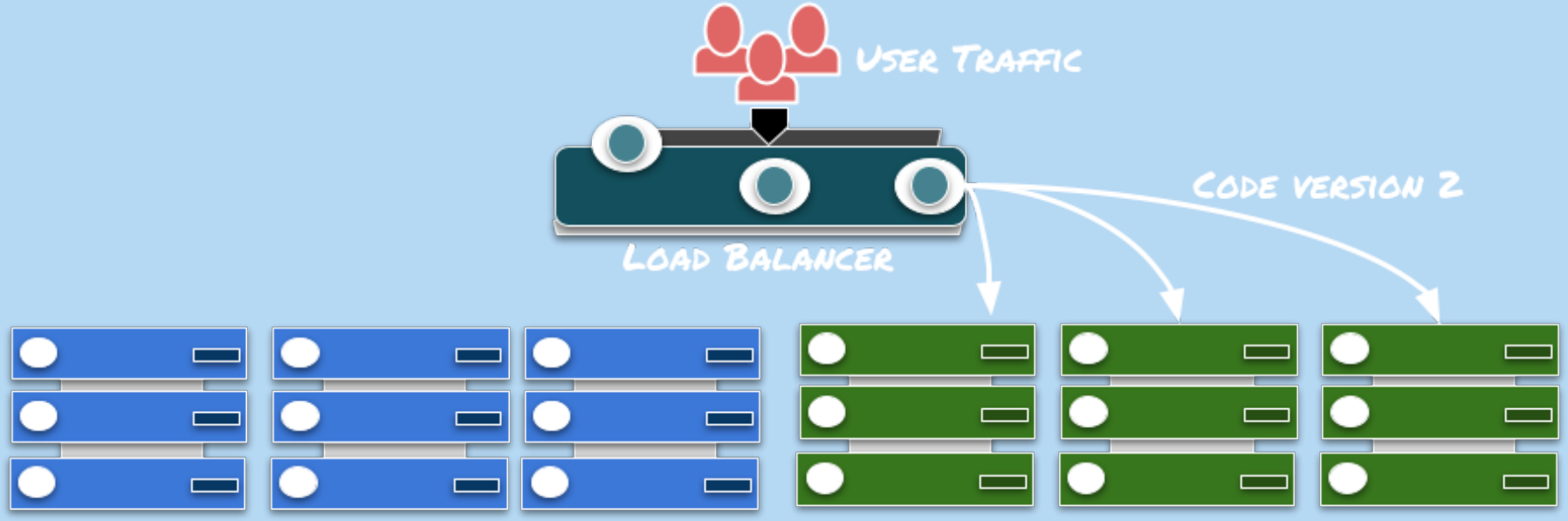
BLUE-GREEN DEPLOYMENT

- **L'idée :** Avoir deux environnements de production actifs en même temps (Bleu et Vert), mais un seul est utilisé
- Déploiement et derniers tests du nouveau logiciel sur l’environnement non utilisé
- Changement de version active en changeant le routage
  - Changement de version très rapide
  - Retour en arrière facile

BLUE-GREEN DEPLOYMENT : PHASE DE TEST



BLUE-GREEN DEPLOYMENT : PHASE DE TEST VALIDÉE



LIVRAISON CONTINUE OU DÉPLOIEMENT CONTINU ?

- Pouvez-vous déployer sans l'approbation des parties prenantes ?
- Vos besoins en matière de système et de vérifications permettent-ils une automatisation de bout en bout ?
- Pouvez-vous exposer vos utilisateurs aux changements de production petit à petit
- Quelle niveau de maturité DevOps a mon équipe ?

DEVOPS

Mais c'est quoi le DevOps ?

LE DEVOPS

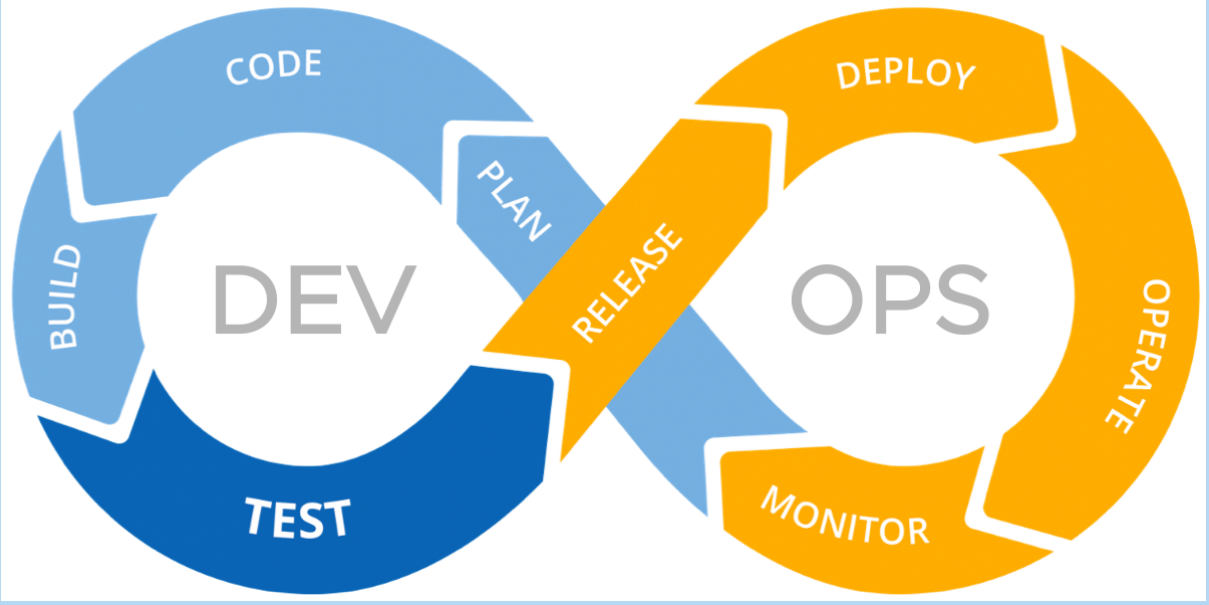
- Combinant développement -DEV- et opérations -OPS- :
  c'est l'union des personnes, des processus et des technologies destinés à fournir continuellement de la valeur aux clients.
- une culture
- une stratégie opérationnelle

UNE VISION INITIALE DIFFÉRENTE

- DEV: Modifications aux moindres coûts, le plus rapidement possible
- OPS: Stabilité du système, qualité, sécurité

LE BUT

- Améliorer la communication entre les développeurs et l'exploitation afin de réduire le temps de mise sur le marché d'un produit
- Proposer des bonnes pratiques destinées à répondre au besoin croissant d'industrialisation et de normalisation du système d'information
- Sécuriser et stabiliser les mises en production
- Limiter les interventions humaines : sources principales d'erreur



LES OUTILS NÉCESSAIRES

- Les outils de partage de code :
  - Les CVS : Git / SVN / Mercurial …
  - Les forges logicielle : Gitlab / Github / BitBucket
- Un IDE : Eclipse / NetBeans / IDEA ...
- Des outils de build : Jenkins / GitLab CI / teamCity ...
- Une plateforme d’analyse de la qualité de code : SonarQube / Code Climate...

L'idée est d'assembler tous ces composants dans une usine logicielle en mode DevOps