Depuis maintenant quelques années (il est difficile de donner une date précise), les DSI s’appuient sur la mouvance agile afin de mener à bien leurs projets. Aujourd’hui, les patterns agiles arrivent à maturité et offrent un éventail de méthodologies adaptables à tous les contextes. Les méthodes agiles garantissent la satisfaction du client et non la conformité aux termes d’un contrat de développement. Elles sont centrées sur la satisfaction de besoin du client et non sur les termes contractuels du projet. Nous n’allons pas aborder en profondeur le concept de l’agilité, ceci n’est pas le propos de ce mémoire, mais nous allons tout de même faire un petit rappel des idées fortes de cette méthodologie. Il faut des **cycles courts**, quelques semaines tout au plus, et découper le projet en **petites tâches** puis les **hiérarchiser** **en fonction du besoin**. Cela permet d’éviter le superflu et de se concentrer au début de chaque cycle sur ce qui a de la valeur pour l’utilisateur final. Le **feedback** permanent devient la règle d’or, avec des validations à chaque étape et des techniques ludiques d’évaluation de l’utilité des fonctions. L’agilité offre une meilleure visibilité et permet d’éviter les dérives observées lorsque les développeurs sont isolés. Le **changement** est autorisé voir encouragé, même tardivement,  car c’est un avantage décisif pour le client. Cela permet de ne pas se priver des bonnes idées en cours de route et surtout d’éliminer les mauvaises idées lancées au début du projet. Les méthodes agiles favorisent la **co-construction**, en intégrant l’annonceur lui-même dans le travail quotidien et en responsabilisant la totalité de l’équipe de développement, créant ainsi un véritable **esprit collaboratif** et l’ensemble du projet en gagne en qualité.

Cependant l’agilité, lorsqu’elle est exclusivement cantonnée au développement, se trouve néanmoins freinée par les tâches d’exploitation. Le mouvement DevOps a pour objectif d’étendre les pratiques agilistes à la livraison et au déploiement du projet.

DevOps

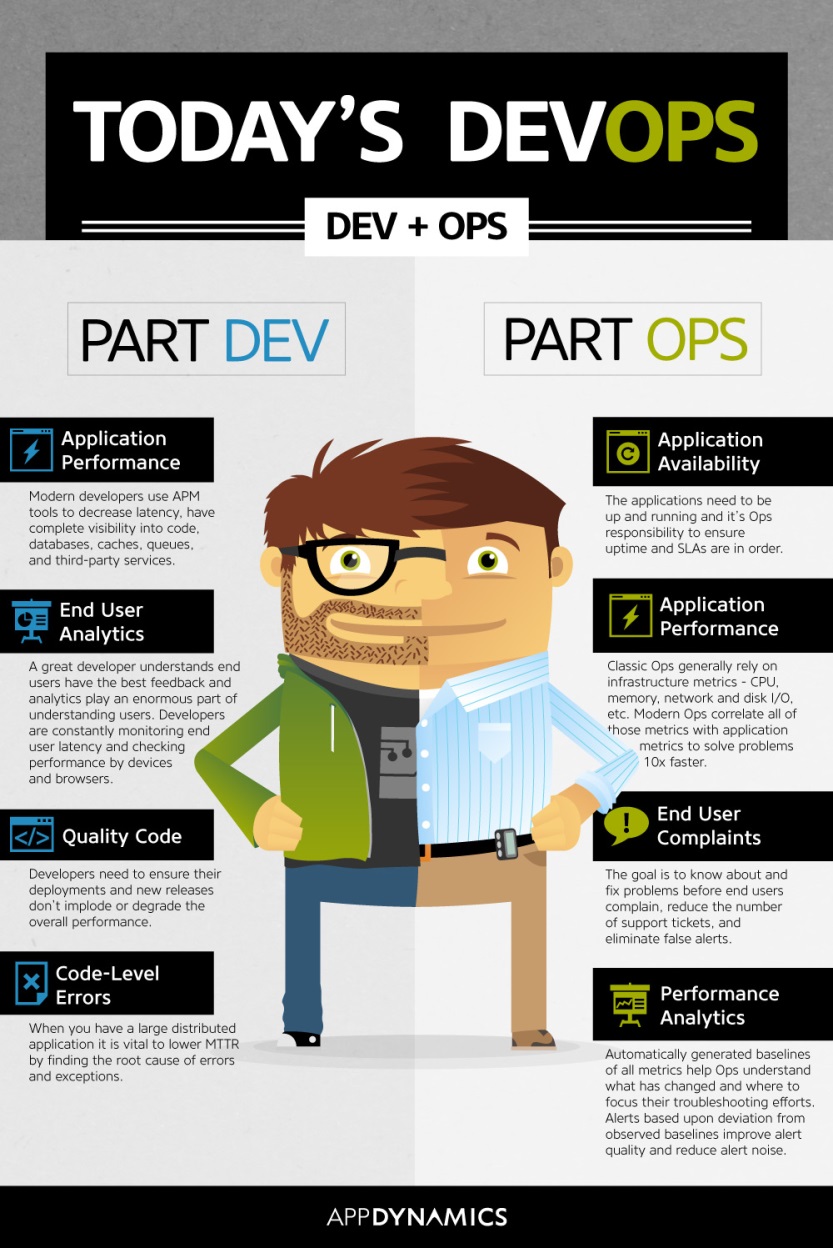
Le mouvement DevOps, contraction de « Development » et « Operational » tente de répondre à la problématique, certainement aussi ancienne que les DSI, qu’est la frontière qui sépare les développeurs (ceux qui écrivent le code source) et l’exploitation (ceux qui déploient et exploitent).

Cette problématique est remise au goût du jour notamment grâce à deux groupes d’acteurs qui sont les agilistes, qui ont amélioré

# DevOps mais encore?

Au vue de cette courte introduction, la première question que l’on ait amené à se poser est ; Pourquoi séparer Dev et Ops ?

## “The wall of confusion”



# Culture

## Features Team

“You build it, you run it”

## “Ops-ready”

## Anticiper l’activité Ops

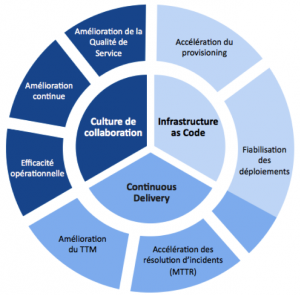
# Infrastructure as Code (Ops)

## Automatisation des environnements

(Puppet, Chef, …)

## Cloud computing

# Continuous delivery (livraison continue) (Dev)



## Continuous Integration (Intégration continue)

## Continuous Delivery (Livraison continue)

## Continuous Deployment (Déploiement continue)

## Continuous Improvement

Etude de l’art

# Bluemix

# Axa

Plateforme d’Intégration Continue (PIC)

