

4 ING INFO

4 ING INFO

# DEVOPS



# Plan

- ▶ Introduction
  - ▶ Définition
  - ▶ Méthodologie d'implémentation
  - ▶ Les pratiques DevOps
  - ▶ Les outils
  - ▶ Les Avantages
  - ▶ Perspectives
  - ▶ Conclusion
  - ▶ Bibliographie



**business  
nécessite  
rapidité**

NATIONAL BESTSELLER

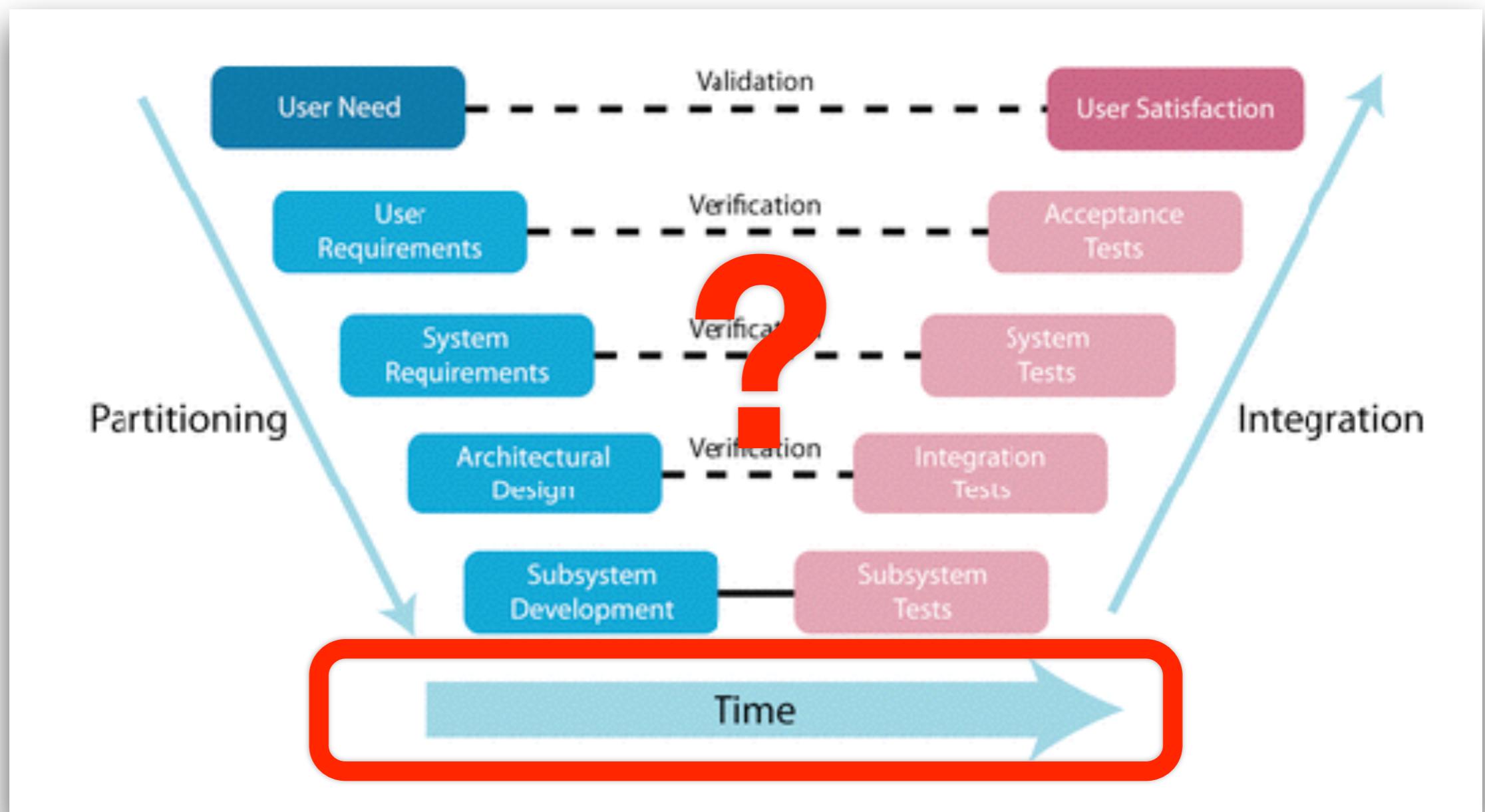
with a new introduction

**it's not the  
BIG that eat  
the SMALL  
...it's the FAST  
that eat  
the SLOW**

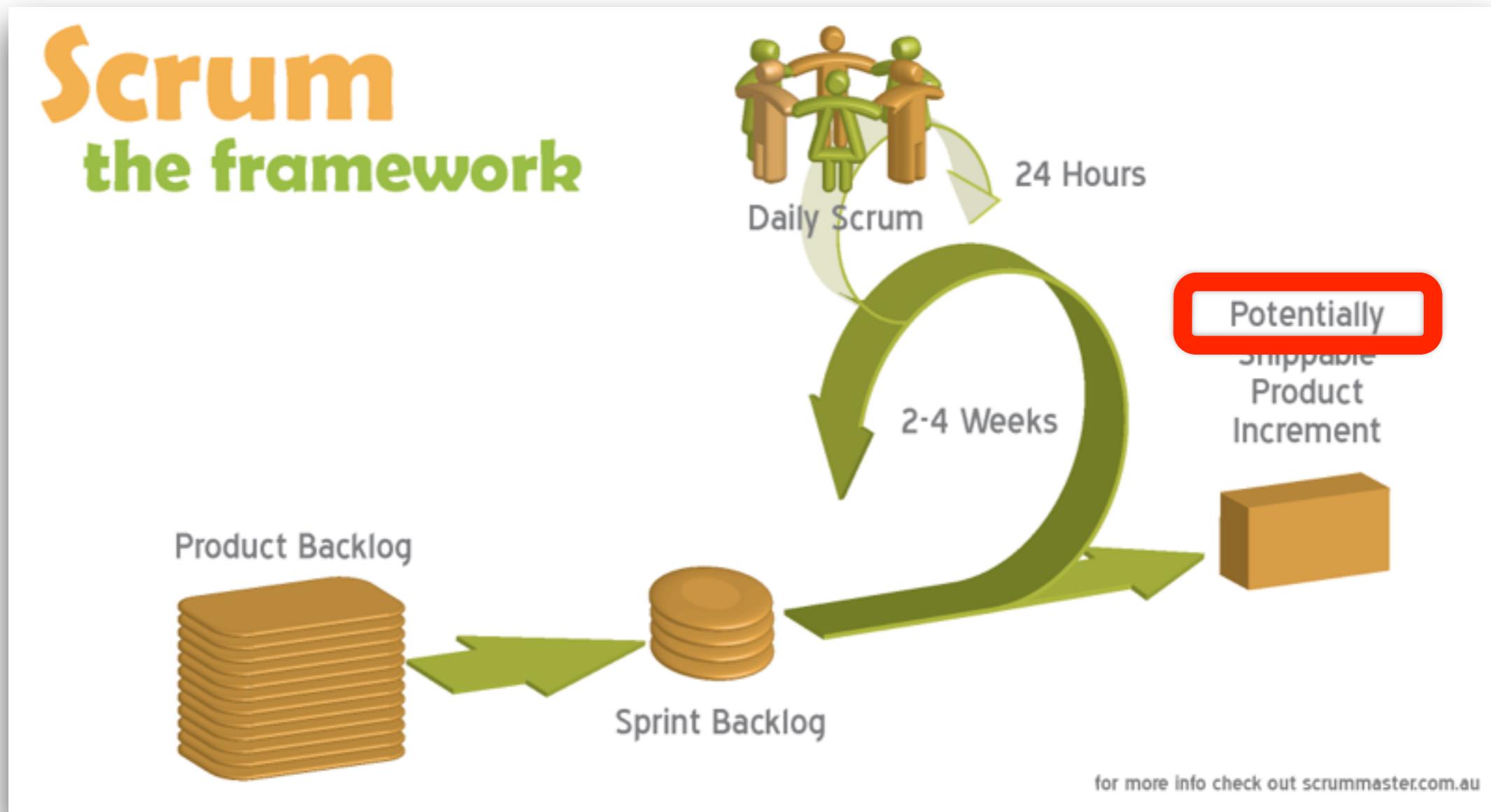
How to Use Speed as a Competitive  
→ → Tool in Business → →

**JASON JENNINGS & LAURENCE HAUGHTON**

# Le cycle en V est long et risqué



# Le développement agile : Une série de « fausses » iterations



Le besoin de rapidité

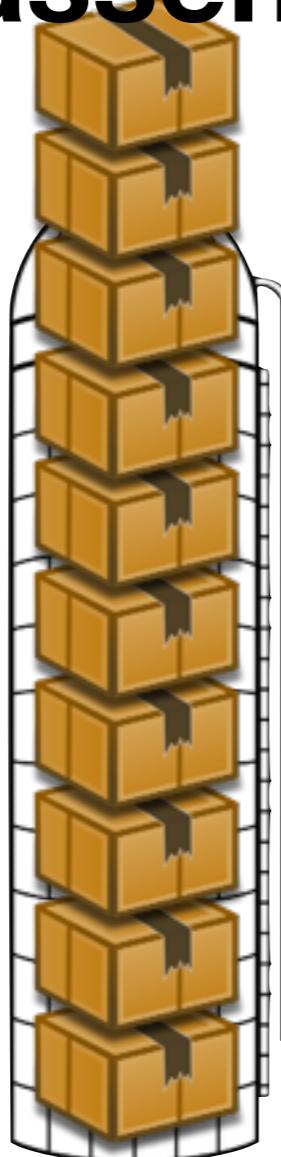
---

Le miracle DevOps : Les faits

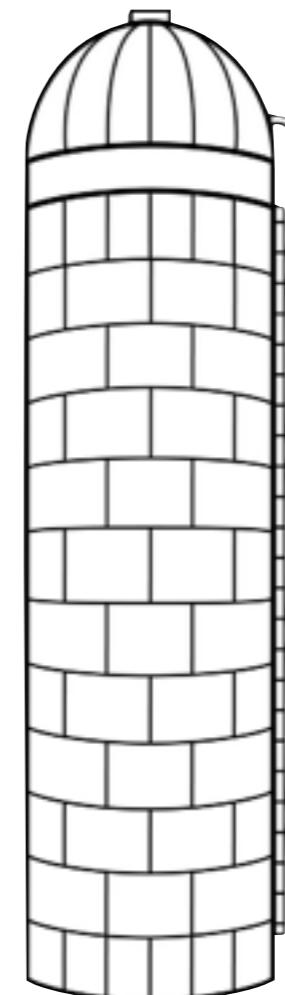
---

Mettre en place DevOps, pas à pas

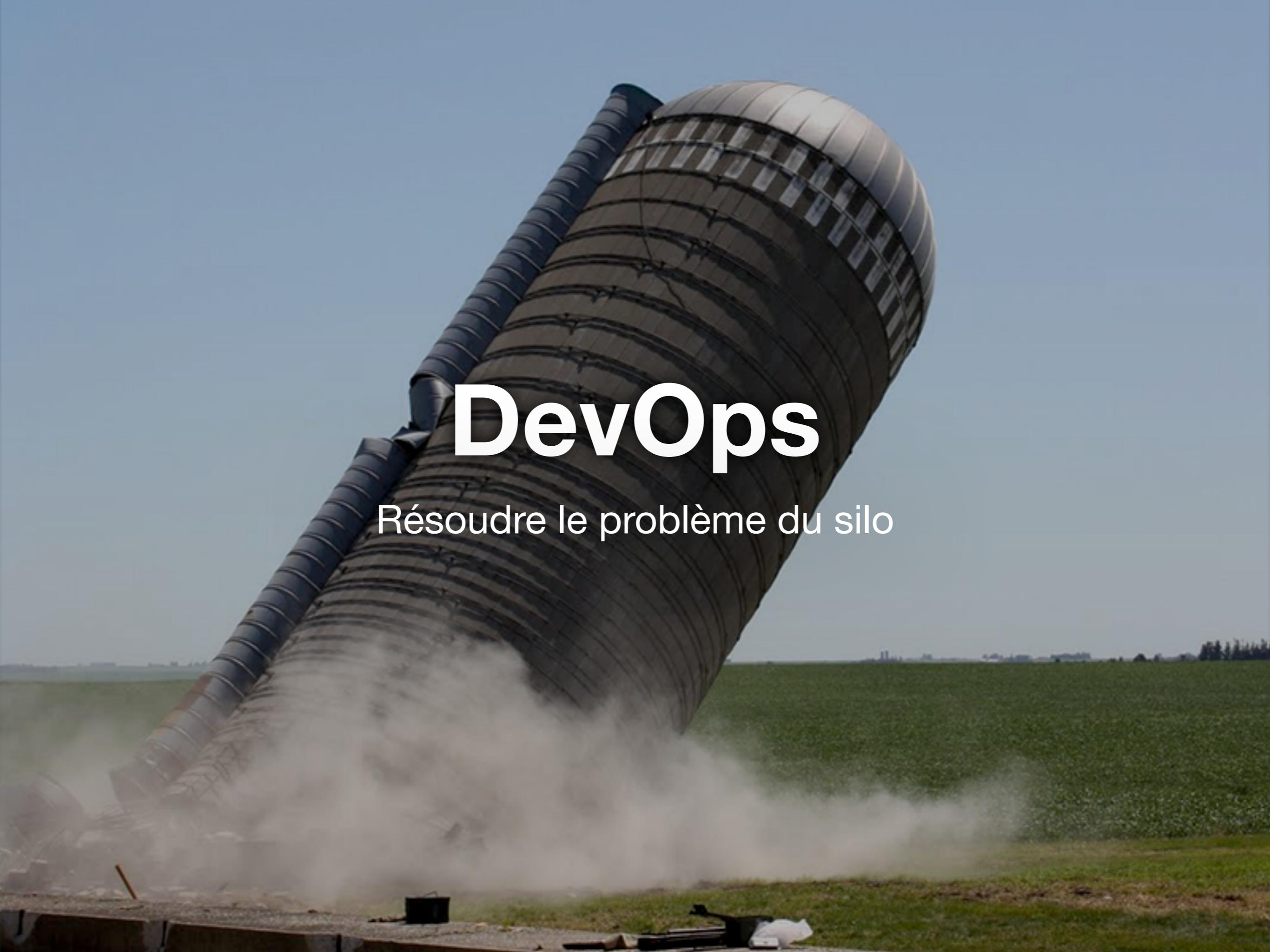
# Car les features s'entassent et ne passent pas en production



**DEVS**



**OPS**

A large, dark grey cylindrical silo stands in a field under a clear blue sky. A thick blue pipe runs along its side, from which a powerful stream of white dust is being sprayed onto the grassy ground below. The dust cloud is visible against the bright sky.

# DevOps

Résoudre le problème du silo

# Introduction

## ► Avant l'apparition de cette approche Devops:

Avant l'apparition de l'approche DevOps, le problème de communication entre les équipes de développement et les équipes opérationnelles était bien réel. Ce qui avait pour conséquences qu'aucunes des deux équipes ne pouvaient connaître les obstacles de l'autre. Ces derniers étant découverts lors de la mise à disposition du code de la part des équipes de développement, aux équipes d'exploitation. Suite aux nouvelles demandes où la vitesse et l'efficacité sont des critères devenus importants, les entreprises doivent être capables de moderniser leurs processus afin de ne plus perdre de temps.

# Définition

- ▶ Crée en 2009
- ▶ Résulte de la combinaison de 2 mots anglais: « **development** » (développement) et « **operations** » (exploitation)
- ▶ Il a pour objectif de créer une communication entre les développeurs et les opérationnels, le but sera de rapprocher les deux univers pour faire en sorte de créer un travail de collaboration, d'avoir des objectifs communs notamment dans les situations à risques avec un cycle de développement court. L'idée est de pouvoir **saisir les opportunités du marché et d'innover rapidement.**



Endless Possibilities: DevOps can create an infinite loop of release and feedback for all your code and deployment targets.

# Méthodologie d'implémentation

Obtenir le soutien de la direction et de ses équipes

Réaliser un audit pour avoir une photo du fonctionnement actuel

Définir des objectifs successifs facilement atteignables (étapes par étapes)

Mettre en place le multi-environnement des applications (développement, intégration, test, pré-production et production)

# Méthodologie d'implémentation

## 1. **Obtenir le soutien de la direction et de ses équipes**

la direction doit concevoir l'organisation la plus adaptée et soutenir les équipes dans la transition vers ce mode de fonctionnement.

## 2. **Réaliser un audit, comprendre l'existant, et l'organisation actuelle**

- ▶ Concevoir un processus DevOps, c'est comprendre l'existant par la réalisation d'un audit qui intégrera le fonctionnement actuel de l'organisation pour ensuite se projeter sur les objectifs à atteindre.
- ▶ Chaque processus DevOps a ses spécificités qui collent à l'histoire, à l'organisation de l'entreprise.
- ▶ Il n'y a pas un seul et unique processus DevOps, mais une multitude de processus DevOps qui répondent globalement à la même finalité.

# Méthodologie d'implémentation

### 3. Définir des objectifs successifs facilement atteignables

- ▶ Les évolutions de l'organisation et la mise en place de l'automatisation doivent être réalisées par petit pas pour ne pas perturber la livraison du sprint.
- ▶ On commençant par la mise en œuvre sur un nouveau, ou sur un petit projet, pour en assurer la viabilité et le rodage. Les **objectifs doivent être simples et clairs**.

### 4. Définir les outils nécessaires

- ▶ L'efficacité des outils est très importantes. Ils ne se valent pas tous et ne répondent pas toujours aux mêmes objectifs. C'est pourquoi il est important de mettre en place un environnement logiciel efficace, on parle d'usine logicielle Devops.
- ▶ Il faut prendre le temps de les sélectionner, les évaluer, et bien comprendre leurs fonctionnements pour utiliser toutes leurs puissances.
- ▶ Il faut également s'intéresser à leurs intégrations dans l'entreprise et comment ils vont s'interfacer.

# Méthodologie d'implémentation

## → **Un accompagnement Devops**

- ▶ Le moyen le plus efficace (sans passer par des tâtonnements coûteux en temps et financièrement) est de passer par un **accompagnement Devops** et faire appel à un expert DevOps qui va vous indiquer les bonnes pratiques, vous guider dans les bons choix/décisions, et mettre en place une approche devops efficace et adapté à votre infrastructure.



# Méthodologie d'implémentation

- ❖ **Les autres éléments à prendre en compte pour mettre en place une approche Devops :**
  - ▶ Configurer la supervision de tous les environnements de la même manière en y incluant la supervision métier : extraire de l'application des indicateurs clés de son bon fonctionnement
  - ▶ Utiliser un gestionnaire de version.
  - ▶ Adopter une numérotation de version claire (traçabilité et identification rapide)
  - ▶ Sécuriser les accès aux environnements et n'utiliser que des comptes nominatifs
  - ▶ Utiliser des outils d'automatisation fiable et facile à maintenir
- ▶ Mettre en place une centralisation de logs

Faire travailler en  
équipe  
les Devs en les Ops vers des  
objectifs communs

# **Créer un esprit d'équipe entre les devs et les ops**

- 1. Eduquer les devs sur les contraintes opérationnelles
- 2. Inciter les ops à déléguer du pouvoir aux devs
- 3. Mélanger les équipes de devs et d'ops

# Le background « classique » d'un développeur



- A appris Java en école/a l'université
- Utilise Windows (Jeux !)
- A installé Linux car il est curieux
- A réalisé plusieurs sites pour son école ou ses amis
- Déploie avec FileZilla

# Etape 0 : Transformer le bébé dev en dev respectable



- Environnement Unix
- Contrôle de version (Git)
- Workflow
- Méthodologie agile
- Tests unitaires et fonctionnels

# Etape 1 : Lui donner un serveur pour s'amuser



- Donner une instance micro à tout le monde
- Libre de faire ce qu'on veut, quand on veut dessus
- ... tout en restant légal :)

# Etape 2 : Lui faire installer un serveur ... au moins deux fois



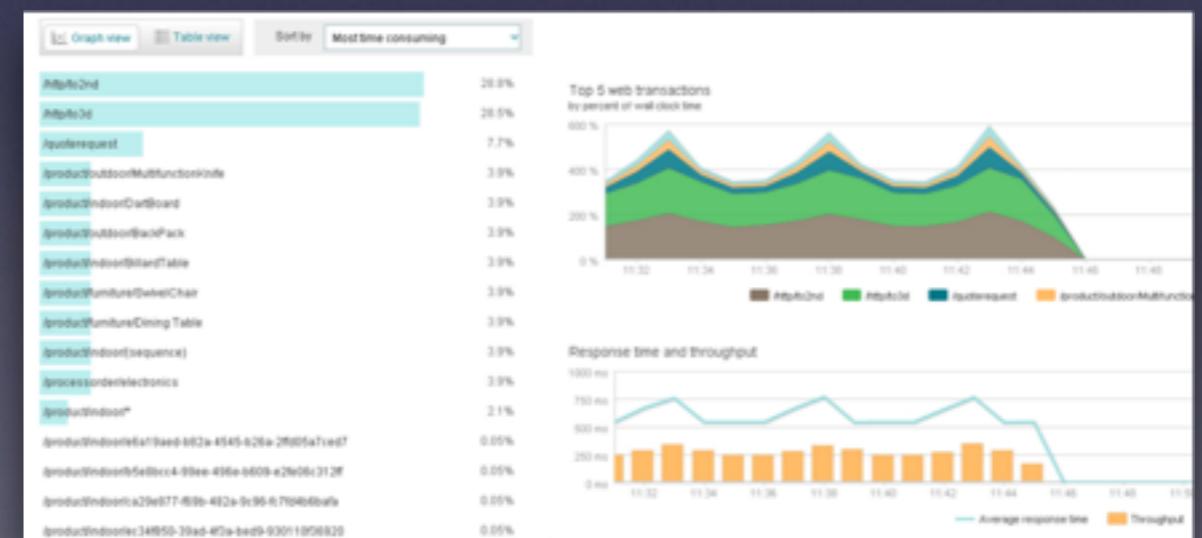
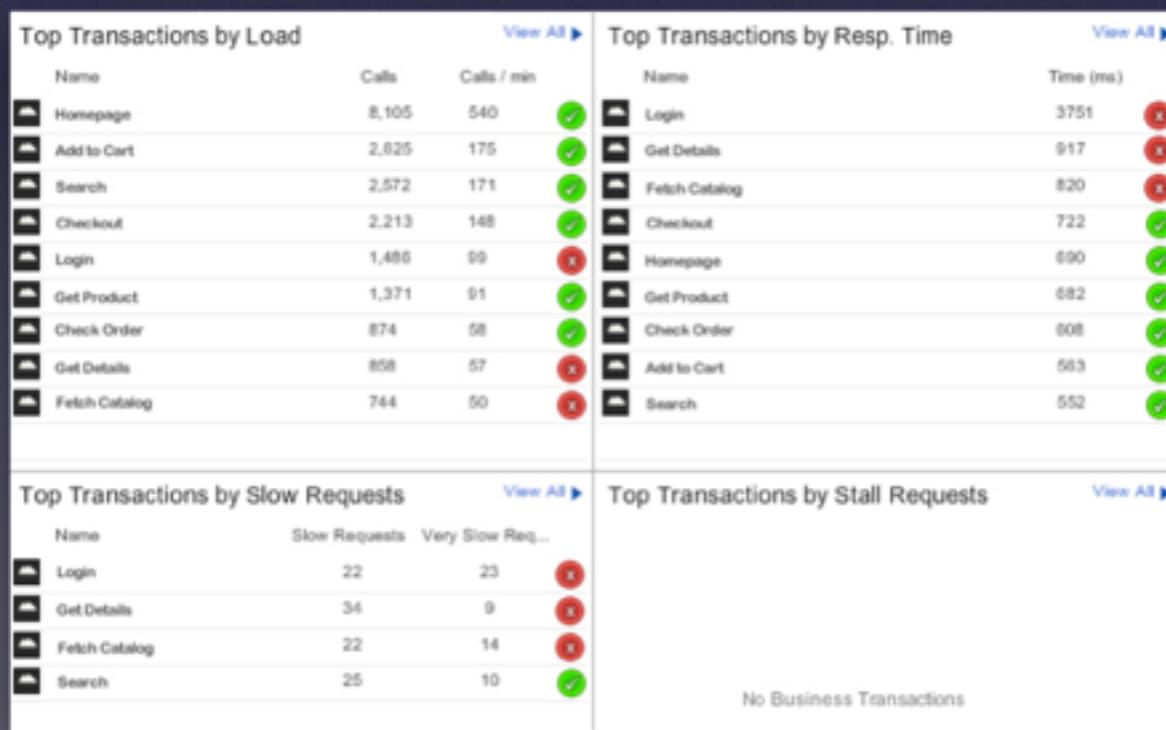
- Beaucoup de devs juniors n'ont jamais installé de serveurs
- Deux fois : pour comprendre l'intérêt du provisioning automatique

# Etape 3 : Le faire déployer ... et comprendre comment ça marche

- Script shell ✘
- Capistrano ✓
- Fabric ✓
- Ansible ✓

# Etape 4 : Faire du monitoring une étape évidente

- Le monitoring est devenu facile (NewRelic, AppDynamic, ELK)
- Facilite de debug (StackTraces, Requêtes SQL, Page performances, HealthCheck)



# Etape 5 : Responsabiliser les devs sur l'intégration continue

- Jenkins : Très flexible et complet
- Travis-CI : Très facile à mettre en place
- CodeShip : Déploiement continu
- GitLab-CI : Intégration avec GitLab



# Etape 6 : Normaliser les environnements avec Vagrant



VAGRANT

- Limite les effets de bord (32/64 bits, Configuration, OS, etc.)
- Différents providers (VirtualBox, Openstack, EC2, Docker, etc.)
- vagrant package : share boxes between devs

# Etape 7 : utiliser Vagrant avec un provisioning automatique



- Apprendre puppet et chef est une étape douloureuse pour un dev ... mais moins que le provisioning manuel
- Pour créer le script:
  - Un Ops expérimenté crée un template
  - Un Dev expérimenté challenge et simplifie

# Etape 8 : Utiliser le cloud !



**openstack™**  
CLOUD SOFTWARE

- Cloud privé : Openstack
- Cloud publique : Amazon EC2



# Etape 9 : Rendre le backup facile



- Push to S3/Swift
- Ou des solutions comme Idera ServerBackup

# Etape 10 : Créer une plateforme de test de charge

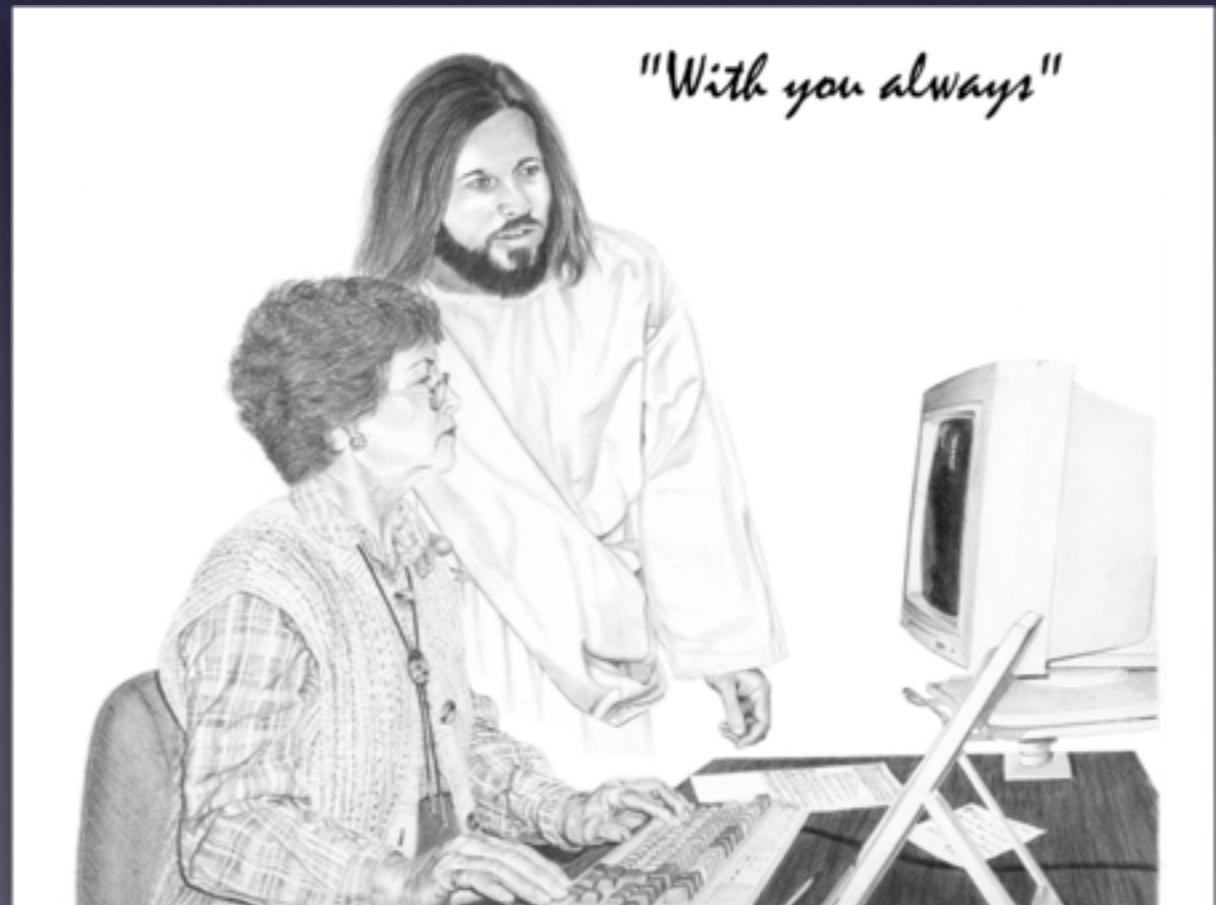


- Créer un serveur facile d'accès (VNC) avec une grande bande passante
- Les Ops expliquent aux Devs ce qu'est un bon test de charge
- Tester les performances régulièrement avec Devs+Ops

# Etape 11 : Inciter au « pair-devopsing »

- Les devs ont un problème avec Puppet/Chef ?
- Les tests de charge montrent un problème de performances ?
- Les Ops voient des erreurs dans les logs ?

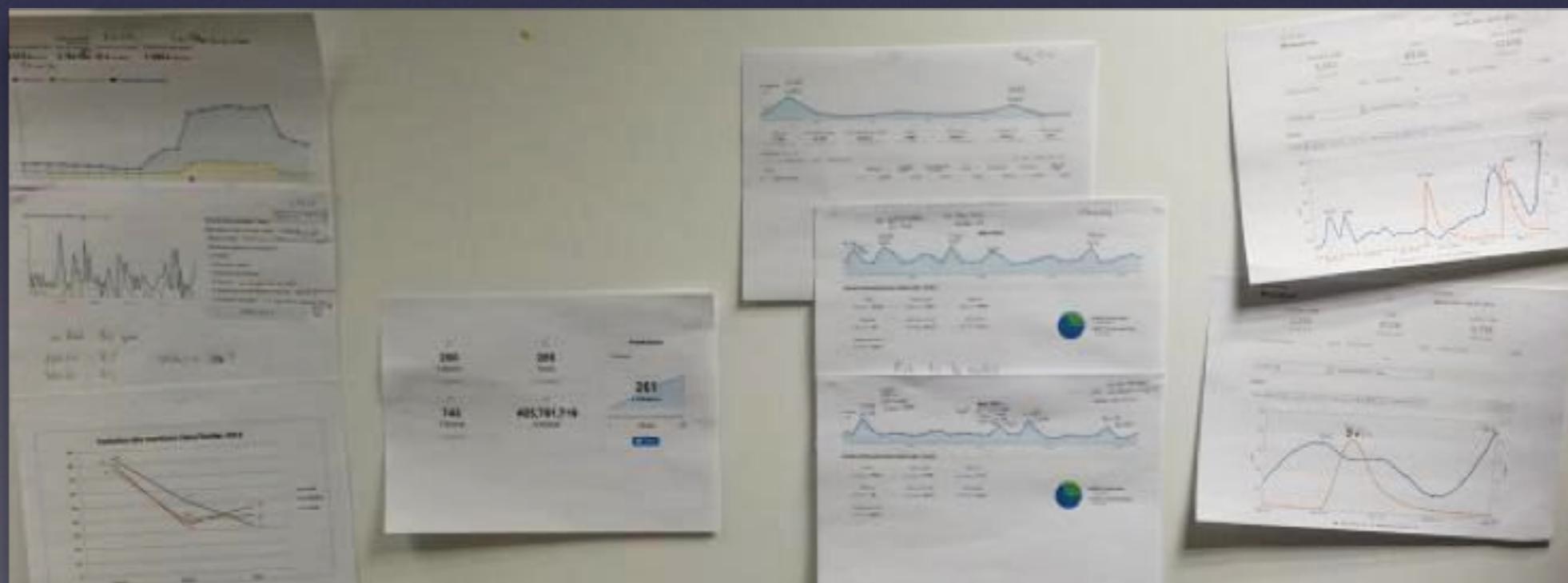
-> Pair devopsing



# Etape 12 : Utiliser la management visuel pour aligner les intérêts

Comment amener les devs à s'intéresser aux performances :

- L'inclure dans la définition du DONE
- Inclure des solutions de mesure de performances dans le provisioning
- Afficher des graphes de performances



WE TOOK THE HOSTAGES,  
SECURED THE BUILDING, AND  
CUT THE COMMUNICATION  
LINES LIKE YOU SAID.



EXCELLENT.

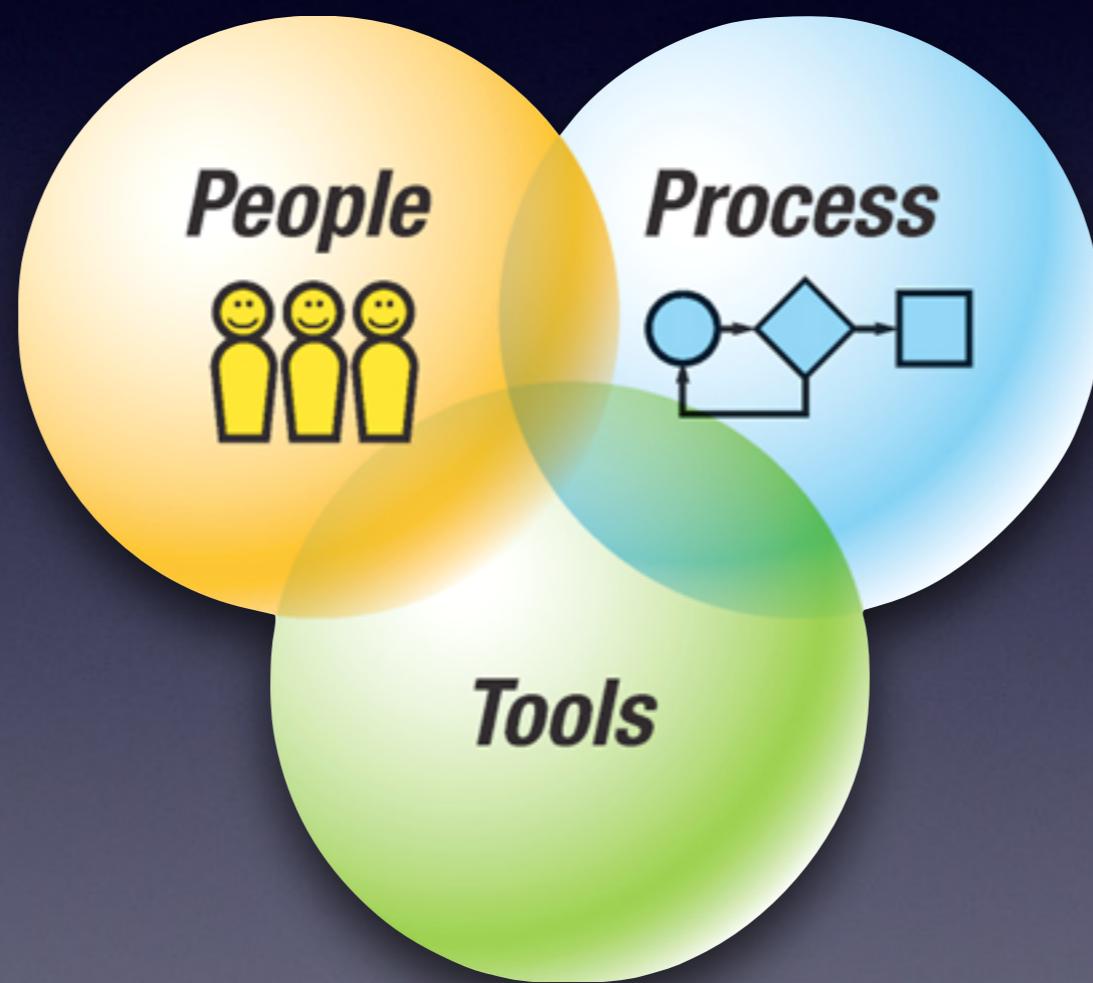
BUT THEN THIS GUY CLIMBED UP  
THE VENTILATION DUCTS AND WALKED  
ACROSS BROKEN GLASS, KILLING  
ANYONE WE SENT TO STOP HIM.



NO, HE IGNORED THEM.  
HE JUST RECONNECTED  
THE CABLES WE CUT,  
MUTTERING SOMETHING  
ABOUT "UPTIME".



# La trinité DevOps



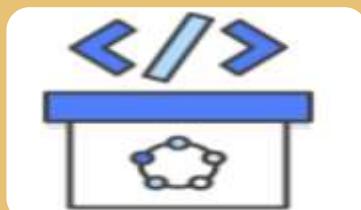
# Les Pratiques DevOps

## Liste des bonnes pratiques DevOps :



### Intégration continue

- Trouver et corriger plus rapidement les bogues.
- Améliorer la qualité des logiciels.
- Réduire le temps nécessaire pour valider et publier de nouvelles mises à jour de logiciels.



### Livraison continue

- Une bonne livraison continue permet aux développeurs de toujours disposer d'un artefact prêt au déploiement ayant suivi un processus de test normalisé.



### Microservices

C'est une approche de conception qui consiste à diviser une application en un ensemble de petits services. Chaque service est exécuté par son propre processus et communique avec les autres services par le biais d'une interface bien définie et à l'aide d'un mécanisme léger.

# Les Pratiques DevOps



## Infrastructure en tant que code

- L'infrastructure en tant que code est une pratique qui implique la mise en service et la gestion de l'infrastructure à l'aide de code et de techniques de développement de logiciels, notamment le contrôle des versions et l'intégration continue.



## Surveillance et journalisation

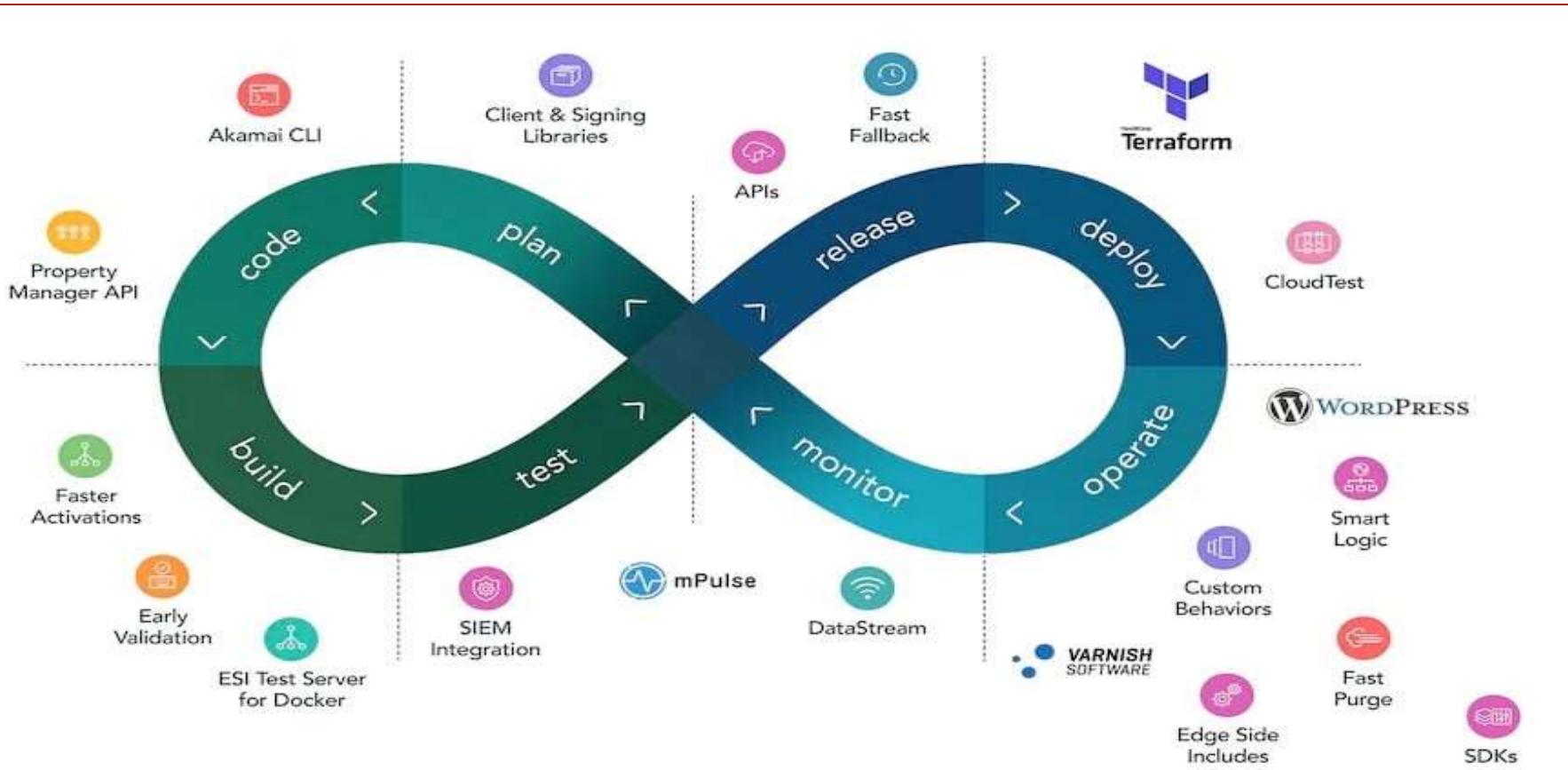
- La surveillance active est de plus en plus importante, car les services doivent aujourd'hui être disponibles 24 h/24 et 7 j/ 7, et la fréquence des mises à jour d'infrastructure augmente sans cesse.
- La création d'alertes et l'analyse en temps réel de ces données aident également les entreprises à surveiller leurs services de manière plus proactive.



## Communication et collaboration

- Le recours aux outils DevOps et l'automatisation du processus de livraison des logiciels établit la collaboration en rapprochant physiquement les flux de travail et les responsabilités des équipes de développement et d'opérations.

# Les outils



# Les avantages



## Rapidité

Avancer plus rapidement pour accélérer le rythme des innovations pour vos clients.

Améliorer votre capacité d'adaptation au marché et gagner en efficacité et en croissance.



## Livraison rapide

- Augmentez le rythme et la fréquence des publications de façon à innover et à optimiser vos produits plus rapidement.



## Sécurité

- Avancez rapidement tout en gardant le contrôle et en préservant la conformité

# Les avantages



## Fiabilité

Assurez la qualité des mises à jour d'applications et des changements d'infrastructure afin de livrer en toute confiance vos produits à un rythme accéléré tout en continuant de proposer une expérience positive aux utilisateurs finaux.



## Évolutivité

- Opérez et gérez vos processus d'infrastructure et de développement à grande échelle. L'automatisation et la cohérence vous aident à gérer les systèmes complexes ou changeants de manière efficace et moins risquée.



## Collaboration améliorée

Les équipes de développement et d'opérations collaborent étroitement, partagent de nombreuses responsabilités et combinent leurs flux de travail. Cela leur permet de limiter les pertes d'efficacité et de gagner du temps

# Perspectives DevOPS

## 1. Standardiser la production pour se recentrer sur l'humain

Les compagnies DevOps natives bénéficient de leur position de second movers. Le DevOps vise en fait soit à éviter, soit à résoudre les conflits classiques de la construction de logiciels. Elle automatise la production pour consacrer plus de ressources aux enjeux spécifiques et humains de l'entreprise

## 2. La décision collective, meilleure alliée de la transformation sur le terrain

La transformation digitale est un processus continu. Et malgré leurs efforts, la plupart des grands groupes historiques présentent un ou plusieurs des problèmes suivants :

- ▶ Des méthodes de production et de gestion “sur-mesure” pour chaque outil ;
- ▶ Et bien sûr, la réticence au changement.

Les instances d'auto-organisation doivent jouir d'un **pouvoir décisionnel**.



# Conclusion

- ▶ Pour conclure, *DevOps* est un enjeu de transformation vitale pour la compétitivité des entreprises. Cette transformation est bien d'ordre culturel et doit donc être soutenue par l'ensemble de l'organisation.
- ▶ Le DevOps prépare les grands groupes aux méthodes de travail du 21e siècle : automatisation de la production, aplatissement hiérarchique, souplesse culturelle, gestion par le collectif. Les grands groupes ne restent pas seuls face à ce défi de taille. Des experts de la transformation agile se tiennent prêts à les accompagner dans l'adaptation de leur organisation et de leur culture

# Bibliographie

- ▶ <https://www.cybersecura.com/post/le-besoin-de-devsecops-dans-les-entreprises>
- ▶ <https://blog.adimeo.com/comment-les-methodes-devops-peuvent-transformer-votre-relation-client-prestataire>
- ▶ <https://www.supinfo.com/articles/single/3997-mise-place-devops-entreprise>
- ▶ <https://blog.syloë.com/4-etapes-pour-mettre-en-place-une-approche-devops/>
- ▶ <https://aws.amazon.com/fr/devops/what-is-devops/>
- ▶ <https://www.soprasteria.fr/perspectives/details/repenser-l-organisation-des-grands-groupes-par-le-devops>