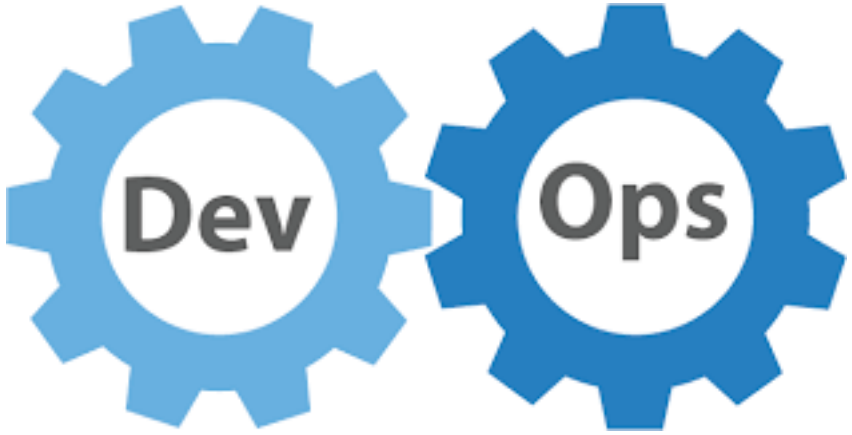


Ch.1- Le courant DEVOPS

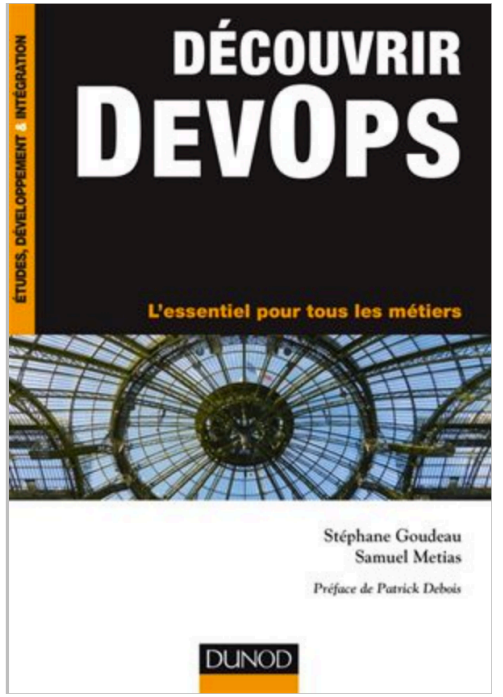
Licence pro DEVOPS

IUT Université Lyon1

V. Deslandres

Références

- http://devops.fr/
- http://www.timspirit.com/publications/livres/la-revolution-devops/



Mars 2016, pour débutants

Culture

Mesure

Automatisation

Partage

Enjeux de Devops

- « Entre le moment où *on veut livrer* et la *livraison*, il s'écoule un mois ! »
- « Mes Dév font la livraison en Prod, car la Prod ne sait pas le faire… »
- « Je trouve que les Dév font n'importe quoi en Prod »

Terme créé par un Français, **Patrick Debois,** en 2009

(première idée : 'Agile System Administration')

DEF « DevOps est la livraison agile de systèmes logiciels » avec une déclinaison **culturelle** et **technique** (Rob England)

Les 3 voies de DevOps

1. **Accélération du travail en cours (WIP)**
   - Une fonctionnalité développée *non déployée* : pas de valeur pour l'E.
   - **Priorisation** des fonctionnalités à forte VA, **notion de fini** (DoD)
2. **Maîtrise de l'environnement**
   - Cela signifie **expérimenter** et **répéter** les tests
   - **Tests Canaris** : vérifier que les environnements de Production sont toujours appropriés
   - **Tests A/B** : on bascule certains *users* cible sur de nouveaux environnements avec des fonctionnalités spécifiques, étude d'impact
   - *Chaos Engineering* : injecter des incidents volontairement, évaluer la résilience des systèmes.
   - Netflix déploie en production + de 1000 fois par jour

« Les problèmes intéressants se passent toujours en production »

Principe **d'Amélioration Continue** : « Si ça fait mal, faisons-le plus souvent »

Les 3 voies de DevOps

3. Création de **FeedBack rapides**, tests à toutes les étapes du processus
   - *Codes review* et Analyse statique de code : à chaque *pull request*
   - Tests unitaires automatisés : à chaque compilation
   - Déploiement automatique (avec des tests) : à chaque intégration (*Continuous Integration)*
   - Déploiement automatique en environnement de pré-production : *Continuous Delivery*
   - Voire en environnement de production : *Continuous Deployment*

DEV & OPS : des besoins contradictoires

Culture du produit

Nouvelles fonctionnalités, amélioration innovation

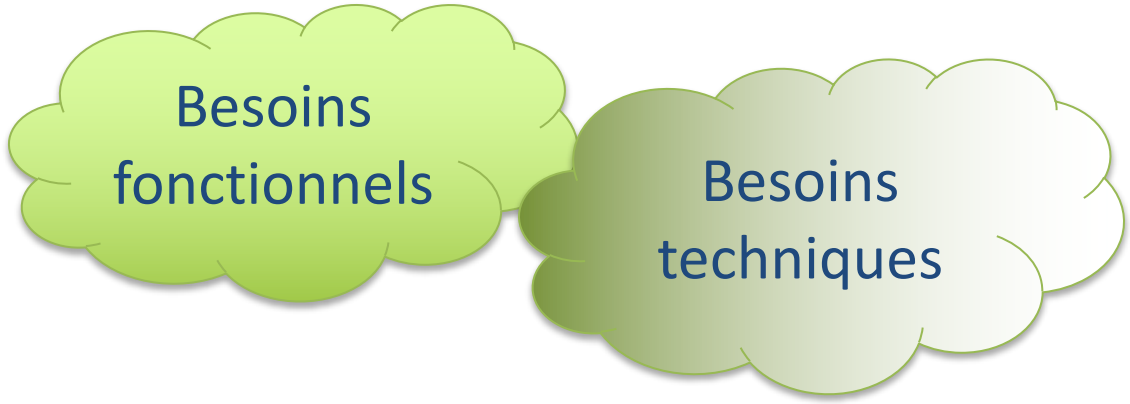
Stabilité, Rationalisation

Culture du service

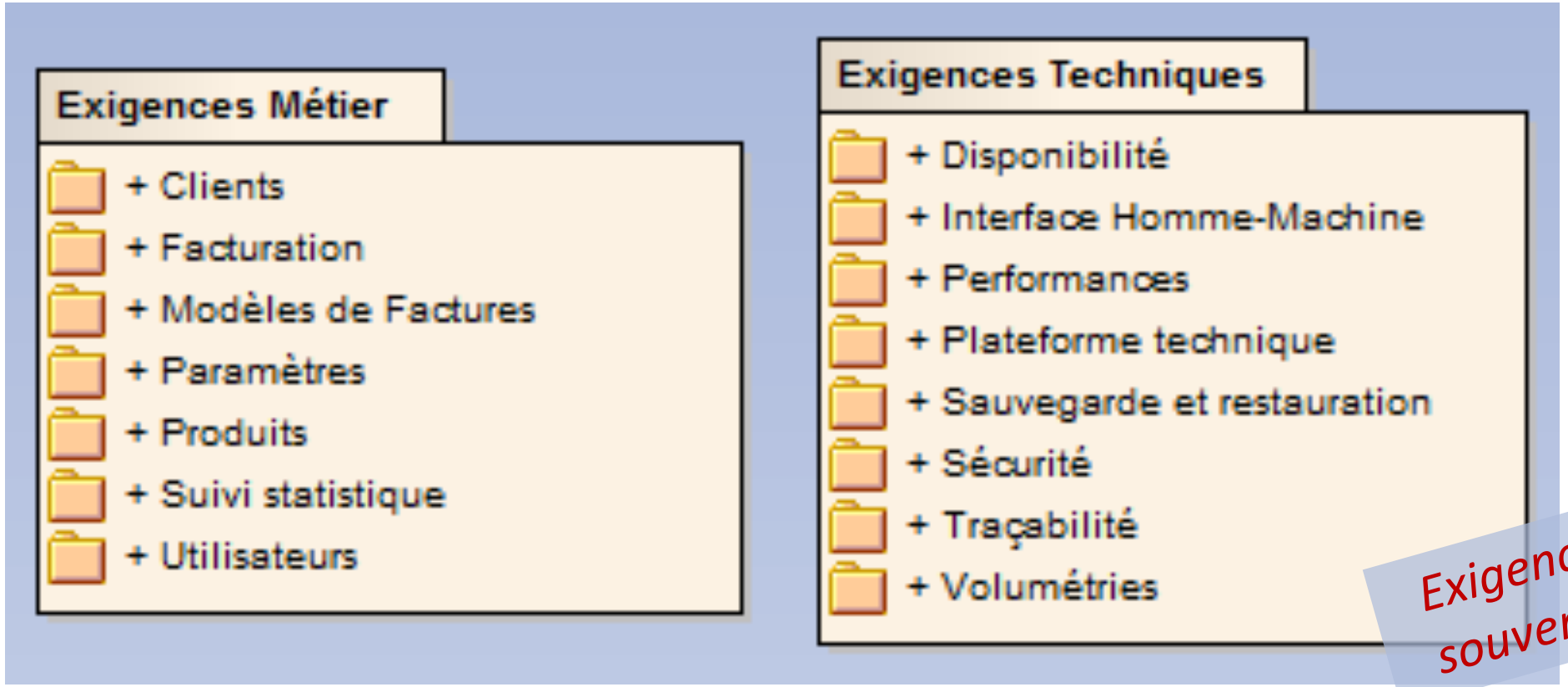
➔ Augmenter le nb des mises en production et les livrer rapidement

➔ Limiter le nb de mise en production

Côté Développeurs



- Considérer 2 besoins
  - Un ex. de cahier des charges :



Exigences techniques : souvent peu détaillées

Besoins fonctionnels

Les **besoins utilisateurs** sont le fruit de la collaboration de 3 types d'acteurs:

LES 3 AMIGOS

QA

(Quality Analyst)
Teste et garantit la qualité du logiciel

MOA/Business

Décrit le besoin
Valide le logiciel livré

Développeurs

Réalisent le logiciel

Besoins techniques

D’un autre côté, les **fonctionnalités techniques** sont souvent écrites, priorisées, développées, testées, validées… uniquement par les Développeurs !

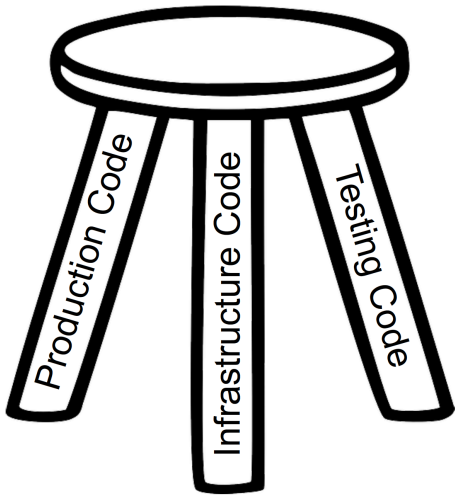
Ex. fonctionnalités techniques

- Mise à jour automatique du DNS
- Empêcher la duplication des noms de déploiement
- Échec rapide si opération non autorisée
- Mise à jour automatique des variables d'environnement
- Retour à la version précédente (rollback) manuelle
- Auto-rollback en cas d'échec d'une nouvelle livraison
- Améliorer les performances de copie de BD
- Etc.

Source : Abigail Bangser, Agile Tour 2015

Les avantages DEVOPS : coté Production

- Simplifier le déploiement d'application dans leurs **diverses versions** et sur **différents** environnements
  - Toute amélioration conduit à une nouvelle version
  - On livre plus fréquemment
- **Automatiser** un maximum de tâches :
  - déploiements d'infrastructure
  - optimisation des environnements
- Voir **ce qui est déployé** et **où**, pouvoir répéter les opérations
  - Déployer si possible sans interrompre les services



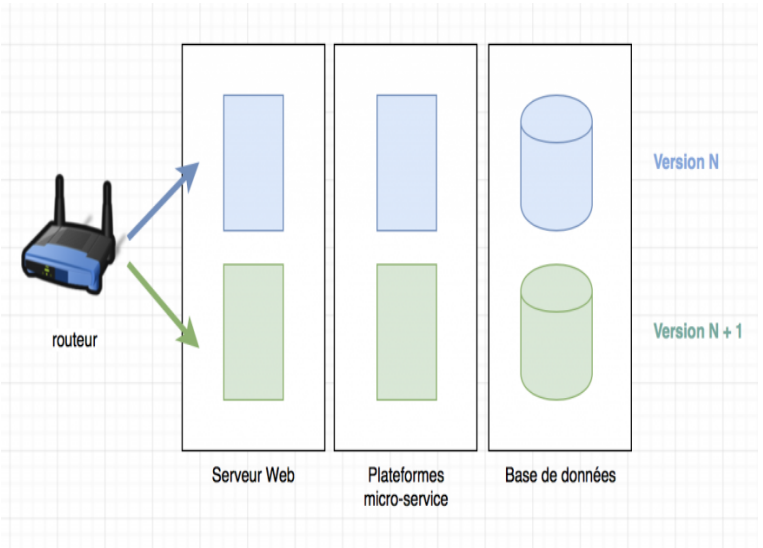


Illustration avec *le Blue / Green Deployment Pattern* :

- Tout tourne avec une **version n** ;
- On teste la **version n+1** sur une plateforme **dupliquée** : si ça marche, on bascule les users sur cette machine ;
- Si jamais il y a un pb quelques mois après, on peut revenir à la plateforme antérieure, c'est immédiat.

Idées fausses DevOps

- « Les principes DevOps ne s'appliquent **qu'aux *nouveaux* projets** »
  - Non, partout, même pour les mainframes.
  - Il suffit que le système soit architecturé pour être testé et déployé.

- « DevOps est un profil, un **métier** » : non, c'est une **culture d'entreprise**
  - Il n'y a pas de *métier DevOps* cumulant les compétences d'un développeur et celles d'un admin
    - un Dev restera un Dev (qui sait réaliser des applica-tions), un Ops restera un Ops (maîtrise le système sur lequel elles s'exécutent). Mais chacun s'enrichit du point de vue de l'autre
  - Si introduire la **sécurité** devient essentiel, on ne va pas ajouter la sécurité à DevOps pour obtenir *DevSecOps*… C'est **l'ensemble de la chaîne de production** qui doit être alignée sur les valeurs de l'entreprise.

Idées fausses DevOps (2)

- « **DevOps = automatisation** » : non, si on prend les processus existants pour les automatiser sans les améliorer, on risque de passer à côté de l'esprit DevOps.
  - Ex.: réservation de créneaux de mise en production par tickets automatiques JIRA ; OK… mais il y a-t-il création de valeurs ?
  - Automatisation = supprimer les tâches répétitives, fastidieuses, pour se concentrer sur les moyens de s'améliorer.

- « **DevOps = CI/CD** » : toutes les boites ne sont pas les GAFAM
  - Livrer des versions en production n'est pas une fin en soi. Le risque est d'empiler des releases, qui généreront du bruit sans améliorer réellement l'expérience des utilisateurs finaux.

Idées fausses DevOps (3)

- « On **bascule intégralement** sur la stack DevOps partagée par les équipes qui l'ont déjà implémentée »
  - Penser déploiement, tests, basculer en Agilité, mettre en place un nouvel environnement de supervision, instaurer une stratégie de Contrôle de Versions, etc.
  - Il est préférable d'y aller **progressivement** : on ne peut **pas tout faire** en même temps. Définir des priorités et des objectifs, faire un POC. Le champ opérationnel de DevOps est très large.
  - Ne pas vouloir apporter une **réponse technique** à un problème d'organisation, de culture.
- « DevOps n'apporte **pas de valeur ajoutée au Client** / Utilisateurs »
  - L'objectif de DevOps est de produire **plus vite et mieux** : « Un client content rapportera plus de business »

Technologies DEVOPS (1/2)

- **Application Performance Management**
  - ***Stackify Retrace*** : la meilleure APM
    - permet de voir les pbs
    - à partir de 10€/mois
  - Et aussi : **NewRelic**, **AppDynamics** (free)

- **Gestion de configuration**
  - Les outils open source de gestion de configuration **Puppet** et **Chef** sont les technologies DevOps les plus souvent utilisées (avec un taux d'adoption de 28%)
  - **Ansible** (21%) : technologie simple à mettre en place, préférée des développeurs
  - Mais aussi : **Terraform, SaltStack, Vagabond**

Technologies DEVOPS (2/2)

- **Plateformes d'entreprise**
  - Microsoft **Azure DevOps** – partenariat avec Zend, Docker
  - **Amazon s3**
  - **OpenStack**

- **DOCKER : très utilisée en DEVOPS**
  - Donne plus de pouvoir aux dév et moins aux fournisseurs de plateforme du *cloud*
  - Utilisée avec Kubernetes de Google
  - Problèmes de sécurité ? (automne 2019)

Enquête PUPPET sur DEVOPS

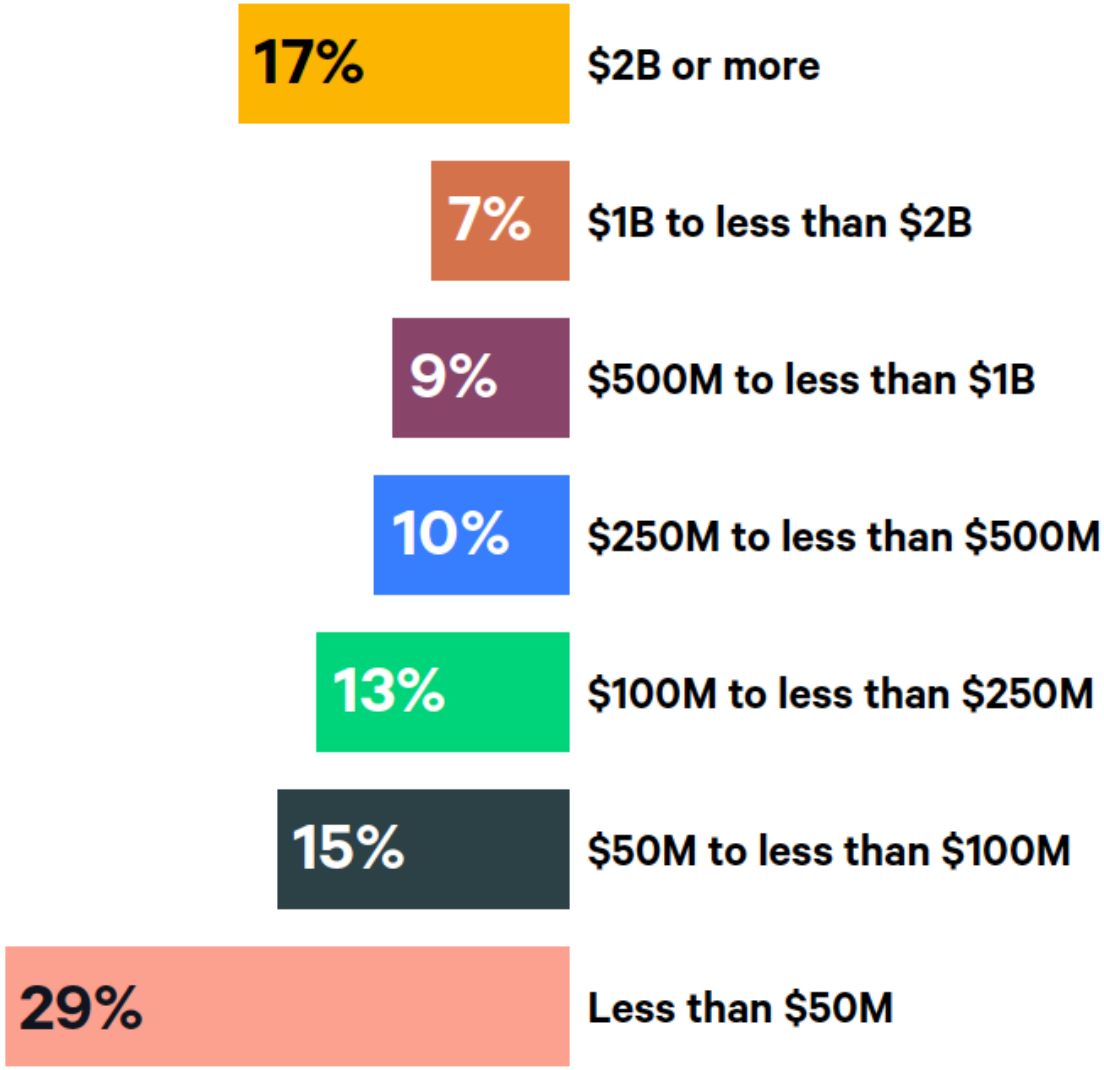
www.puppet.com

- Puppet Enterprise est une plateforme DEVOPS qui permet de distribuer, d'exécuter et de sécuriser automatiquement les infrastructures
- https://puppet.com/resources/report/2020-state-of-devops-report/

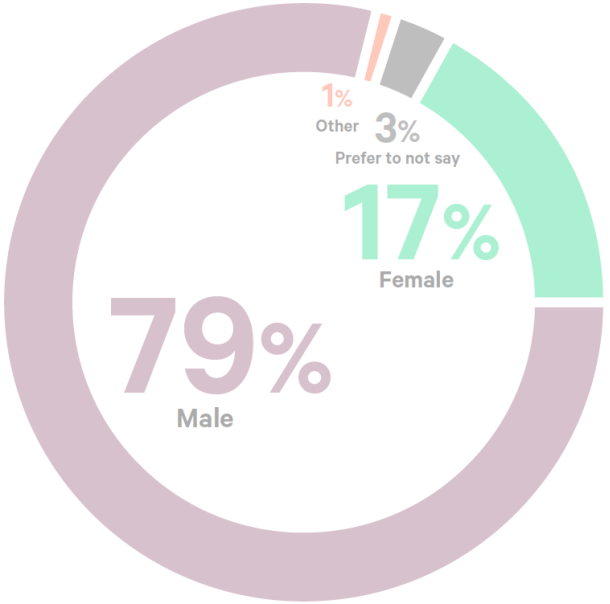
- Juin 2018, 3000 professionnels ont répondu
  - Enquête annuelle, depuis 5 ans, internationale
- Résultats = des **chiffres** + **infos qualitatives** sur :
  - how to integrate security into the software development cycle;
  - the value of experimentation;
  - evaluating the ROI of DevOps;
  - how employee engagement affects organizational success

https://puppet.com/resources/whitepaper/state-of-devops-report

CA des entreprises répondantes : **toutes les Entreprises** se sentent concernées par DEVOPS

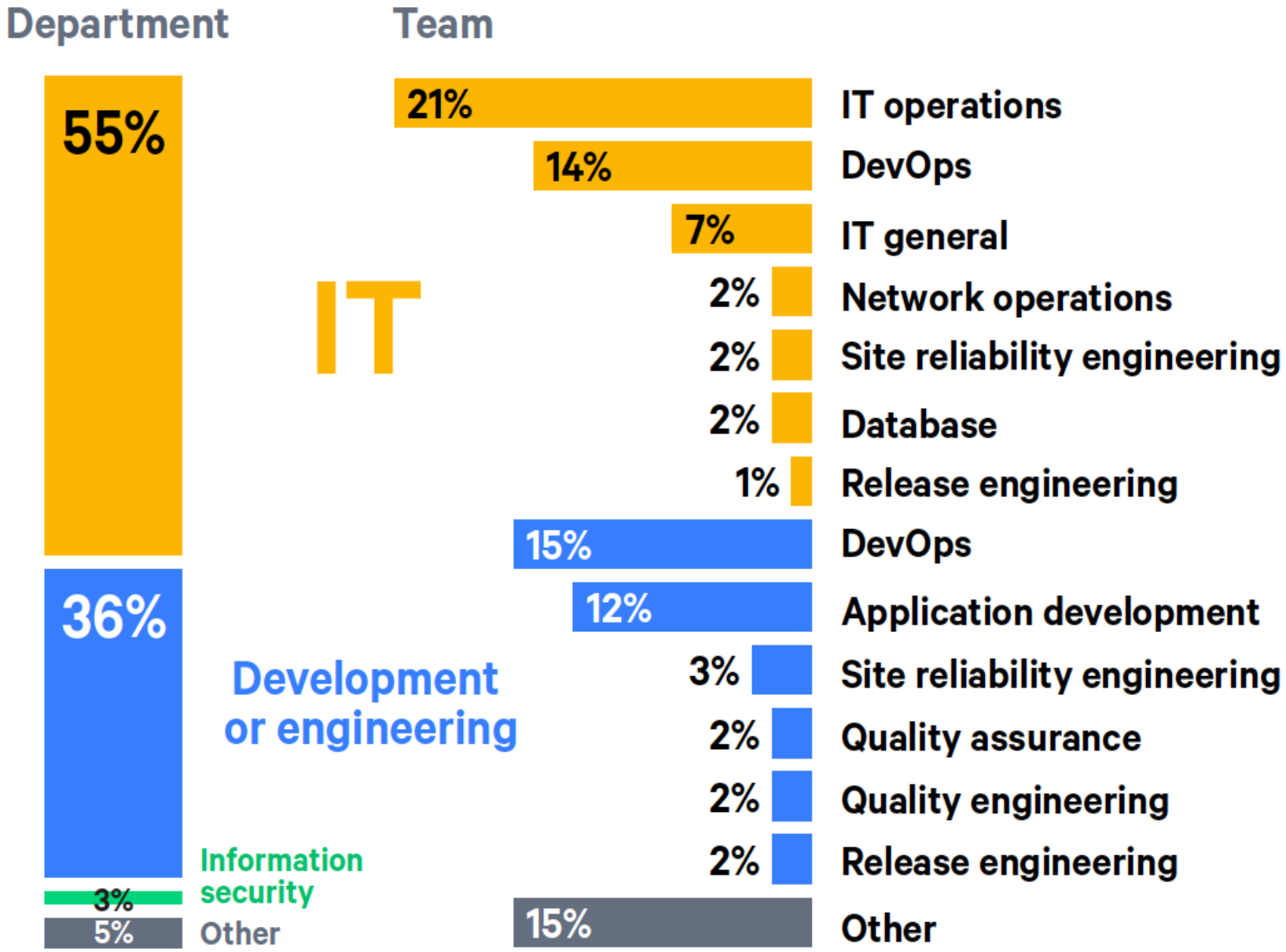


Forte augmentation du taux de répondants Féminin (6% en 2017)



Les services des répondants :

pas uniquement les DSI



Enquête PUPPET (2016) : quels OS ?

What server OSes have you widely deployed? Select all that apply.	**# of Responses**
Linux - Enterprise Linux variants (RHEL, Oracle, CentOS)	3,159
Windows 2012/2012R2	2,446
Windows 2008/2008R2	2,017
Linux - Debian/Ubuntu variants	2,016

# of OSes Selected	**# of Responses**	**% of Total**
0	25	1%
1	**1,035**	**22%**
2	**1,038**	**23%**
3	**883**	**19%**
4	**636**	**14%**
5	**420**	**9%**
6	234	5%
7	139	3%
8	86	2%

78%

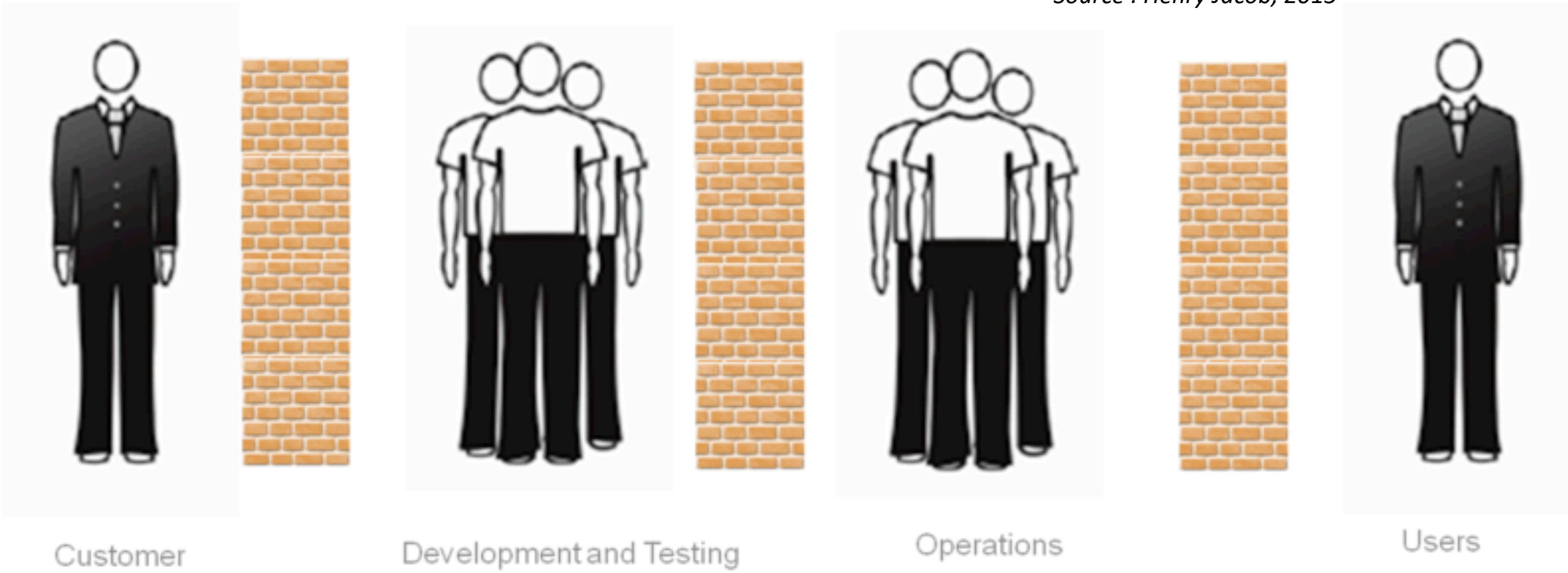
78% of respondents are widely deployed on 1-4 different operating systems.

PUPPET : performances

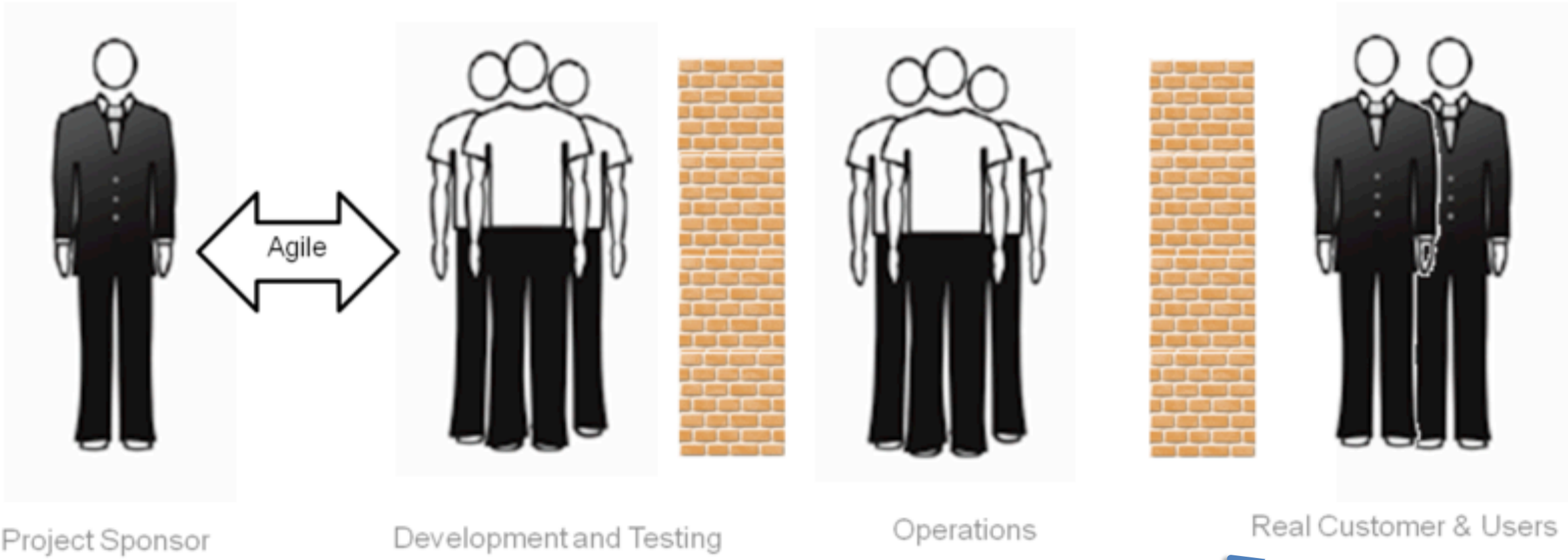
		Équipe performante	Equipe medium	Equipe moins performante
	Fréquence des déploiements	À la demande	Entre 1 par sem. et 1 par mois	Entre 1 par mois et 1 tous les 6 mois
	Temps nécessaire avant mise en production opérationnelle après des modifications *(lead time)*	Moins d'une heure	Entre une sem. et un mois	Entre 1 et 6 mois
STABILITE	**Temps nécessaire à la restauration d'une version propre après un incident (déni de service, coupure imprévue :** *recovery time*)	Moins d'une heure	Moins d'un jour	Moins d'un jour
	Part des modifications dues à un service dégradé (*change failure rate*)	0-15%	16-30%	31-45%

« Les murs de la confusion » du développement logiciel

Source : Henry Jacob, 2015

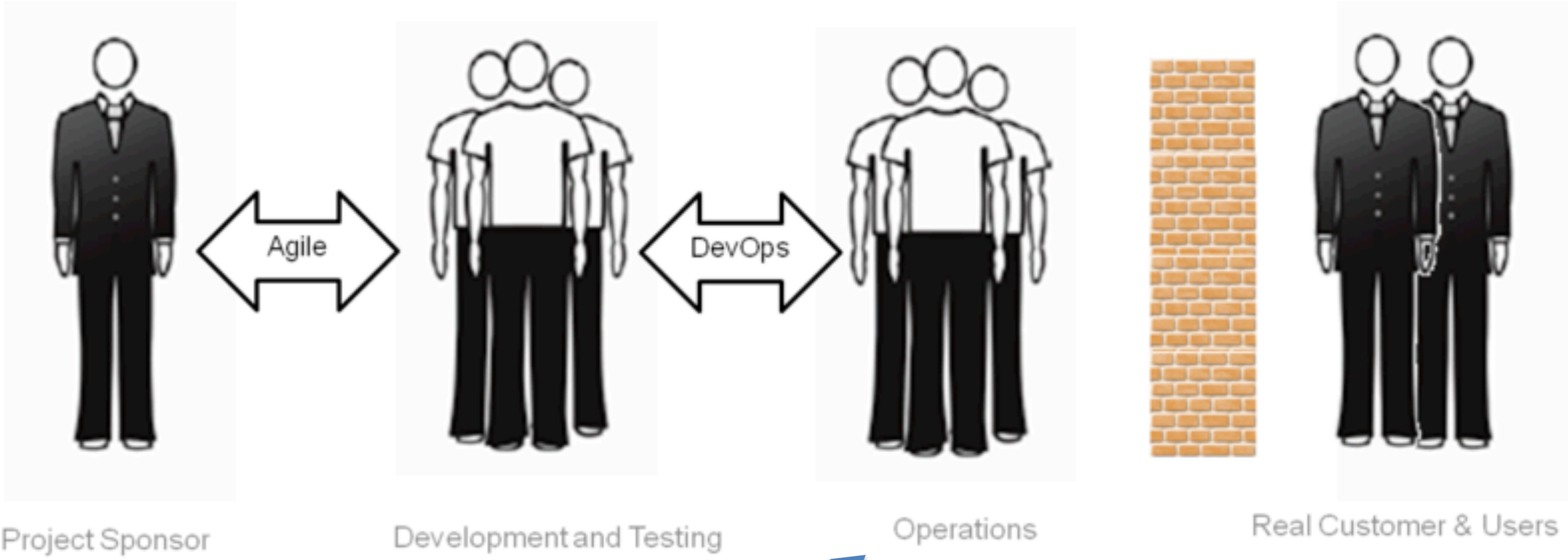


« Les murs de la confusion » (2)



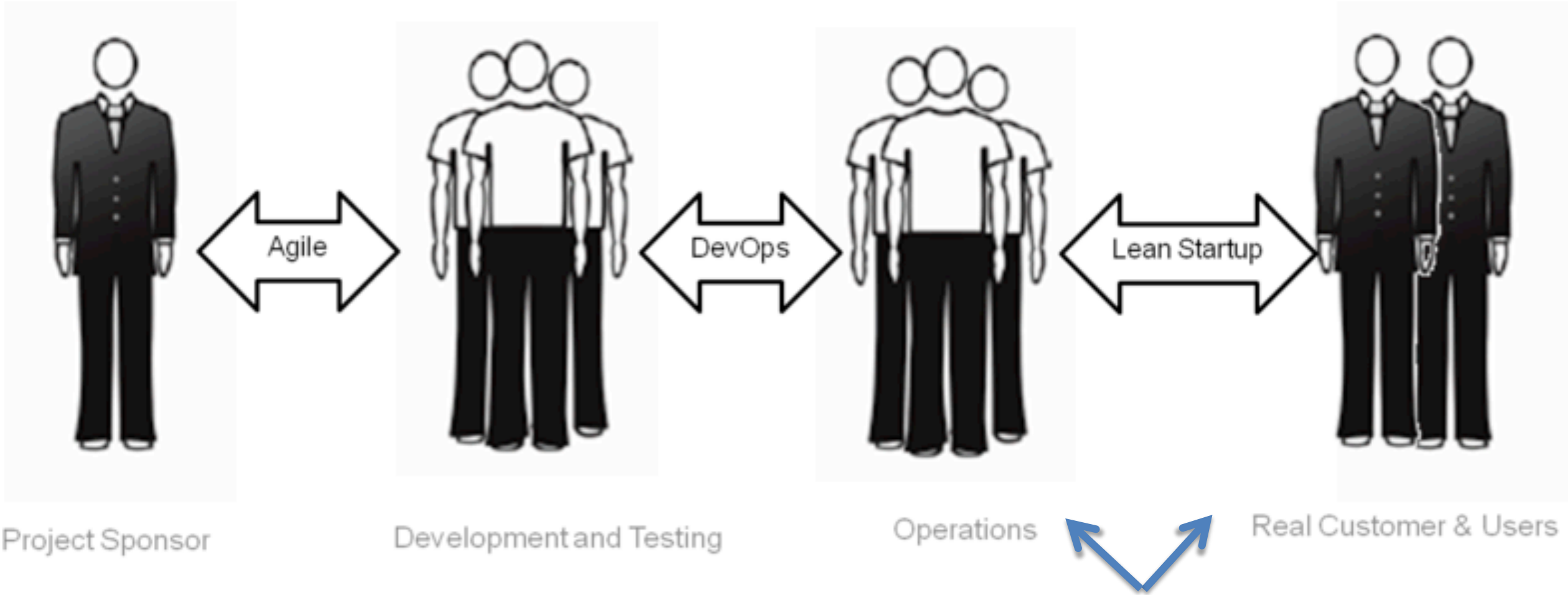
Méth AGILES : fournir des livrables sur des cycles courts
permettant d'avoir un feedback régulier du « représentant Client »

« Les murs de la confusion » (3)



DevOps : permet aux *dév* et à la *prod* de mieux travailler ensemble, en changeant les **mentalités**, en apportant des **techniques** et des **méthodes**.

« Les murs de la confusion » (4)



Le LeanStartUp et **la Livraison Continue** : permet de mieux répondre aux besoins des vrais utilisateurs, en les impliquant très tôt dans la boucle.

DEVOPS : JE RETIENS…

- La **philosophie**
- Les raisons des incompréhensions entre DEV et OPS
- Les attentes de la *prod,* les besoins techniques
- Le besoin **d'automatisation** pour le déploiement
- Le lien avec **l'agilité**, le *lean*
  - L'existence des 'murs de confusion'
- L'intérêt croissant des entreprises pour DEVOPS
- Ses **avantages**