

DAWAN Paris  
DAWAN Nantes  
DAWAN Lyon

11,rue Antoine Bourdelle, 75015 PARIS  
32, Bd Vincent Gâche, 5e étage - 44200 NANTES  
Bt Banque Rhône Alpes, 2ème étage - 235 cours Lafayette 69006 LYON



# Certification DevOps Foundation

Plus d'info sur <http://www.dawan.fr> ou **0810.001.917**

Formateur: Matthieu LAMAMRA

# Présentation de la formation

- Définir le mouvement DevOps
- Comprendre les principes DevOps
- Acquérir les pratiques DevOps
- Maîtriser les outils DevOps
- Préparer la certification DevOps Foundation

# Introduction : ITSM

- ITSM : Information Technology Service Management – Gestion des services informatiques
- Un service IT : Equipe – Equipements – Projets (approche technique)
- Dans l'ITSM : approche « service » - description des processus / flux métier et interfaces
- Processus : ensemble de tâches corrélées utilisant des entrées pour produire des sorties
- Dans l'ITSM : approche « affaires » - gestion de la relation client / relation fournisseur
- Dans l'ITSM : les « 4P » - Personnes – Processus – Produit – Partenaire
- Implémentation d'un cadre pour définir les services : ITIL (1989), COBIT (1994), MOF(Microsoft 2008)....

# Introduction : ITSM

- CMDB : les 4 dimensions de l'IT



# ITIL : historique

- Courant années 1980 : remise en cause par le gouvernement Britannique de ses services IT => missionne la création d'un cadre de bonnes pratiques pour les rendre financièrement viable : « Information Technology Infrastructure Library »
- 1989 - ITIL V1 : catalogue de 30 volumes de descriptions orientées besoins clients et facilitation des affaires
- 2000 – ITIL V2 : vulgarisation de la V1 en regroupant les pratiques en 9 catégories. Microsoft base son propre cadre MOF sur ITIL
- 2009 – ITIL v3 : « Refresh Project » - orientation sur la notion de service. Description de 26 processus en 5 volumes.
- 2011- mise à jour v3
- 2019 – ITIL v4 : accent mis sur la collaboration et rapprochement avec Agile, DevOps et Lean

# ITIL v4 : principes



Focalisez-vous sur la valeur



Démarrez d'où vous êtes



Progressez de façon itérative en vous appuyant sur les retours



Collaborez et mettez l'accent sur la visibilité



Pensez et travaillez de manière holistique

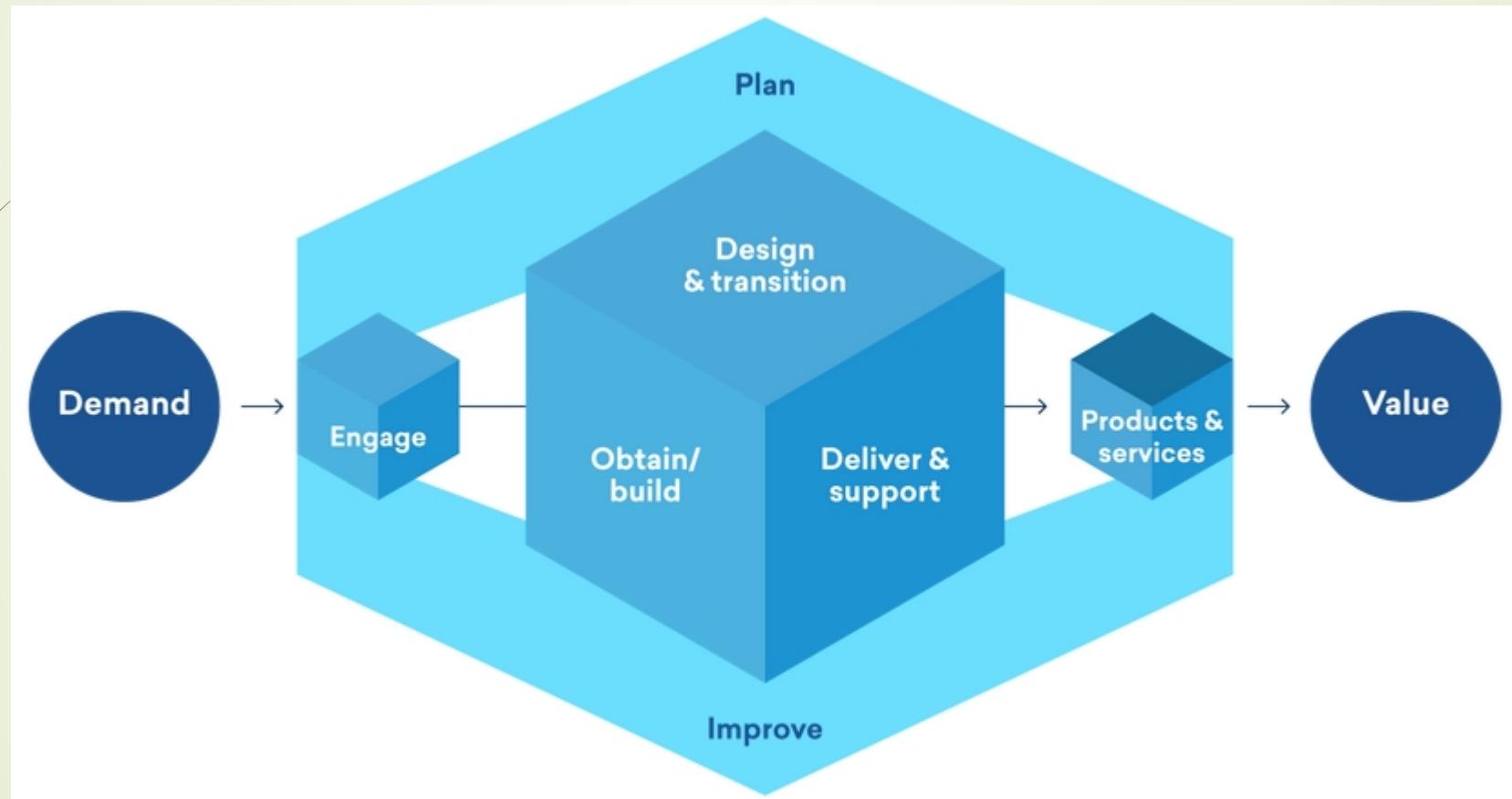


Privilégiez la simplicité et restez pratique



Optimisez et automatisez

# ITIL v4 : activités clés



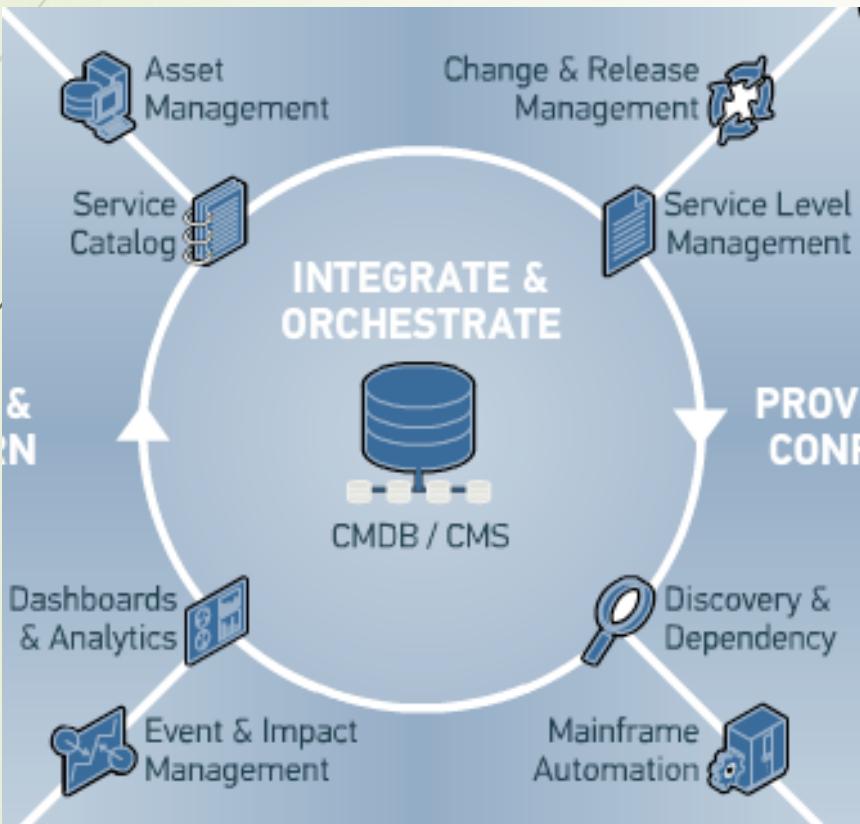


# ITIL : composants

- CMDB : le cœur d'ITIL
  - « Configuration Management Data Base » Gestion des configurations et des changements
  - Base de données de description des actifs SI ou Configuration Item (CI) : logiciels, matériel, documentation, ressources humaines
  - La CMDB décrit ces éléments et leurs interactions pour assurer le bon fonctionnement des services IT : **pratique SCM « Service configuration Management »**
  - Objectifs :
    - porter la visibilité des outils au delà du service IT
    - faciliter les décisions de changement
    - maximiser le Retour sur Investissement (ROI) des ressources : **pratique ITAM « IT Asset Management »**

# ITIL : composants

- CMDB : connections



DAWAN - Reproduction interdite

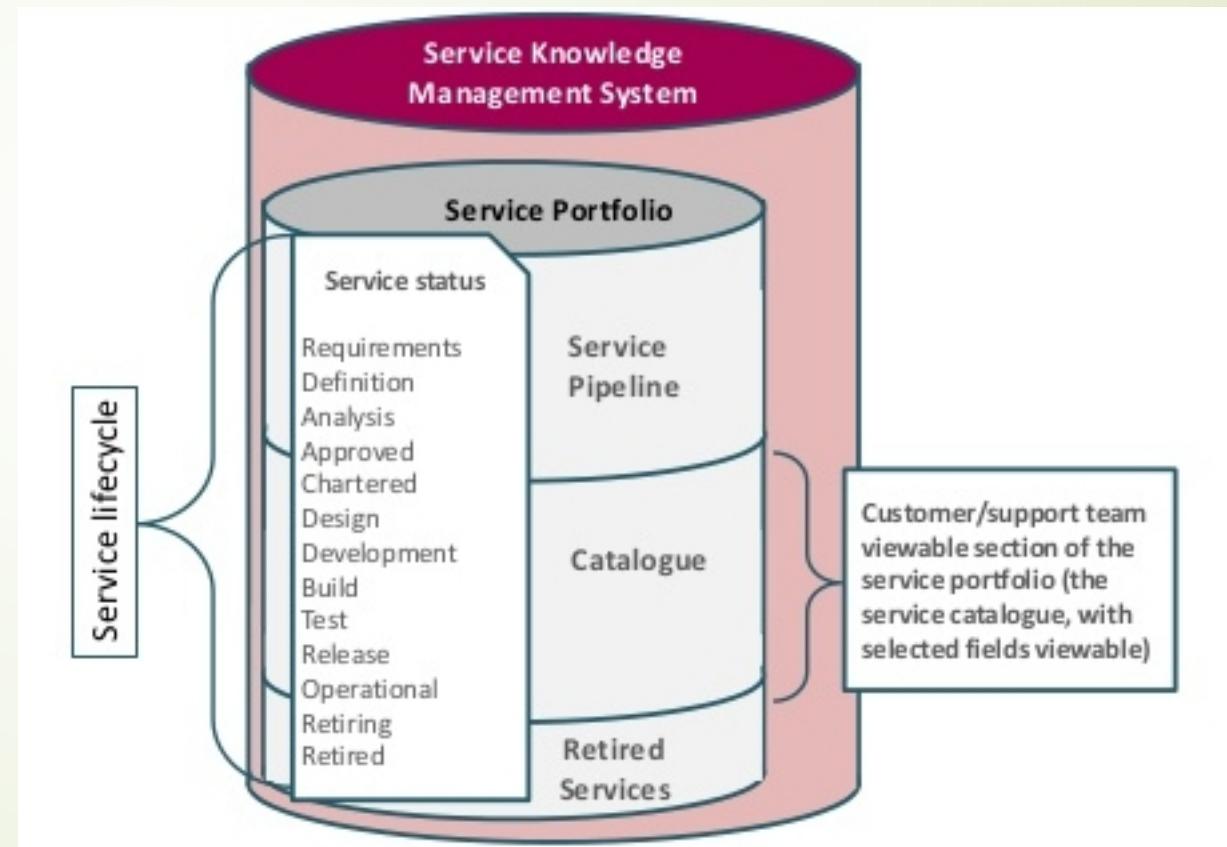
La CMDB communique avec :

- les outils de monitoring
- gestion de niveaux de services (priorités)
- le catalogue des services qu'elle décrit
- les agents d'automatisation de process (tâches périodiques)
- ...

# ITIL : composants

- Service Portfolio

- Service Portfolio : catalogue de gestion du cycle de vie des services
- Accès contrôlé et centralisé aux services
- Suivi de l'état des services
- Relié au système de gestion de la connaissance



# ITIL : composants

- SDP : Service Design Package
  - Ensemble des définitions et propriétés d'un service
  - Agrège des données stratégiques, structurelles (Développement, maintenance) et évènementielles (changements et amélioration continue)



# Vers l'agilité

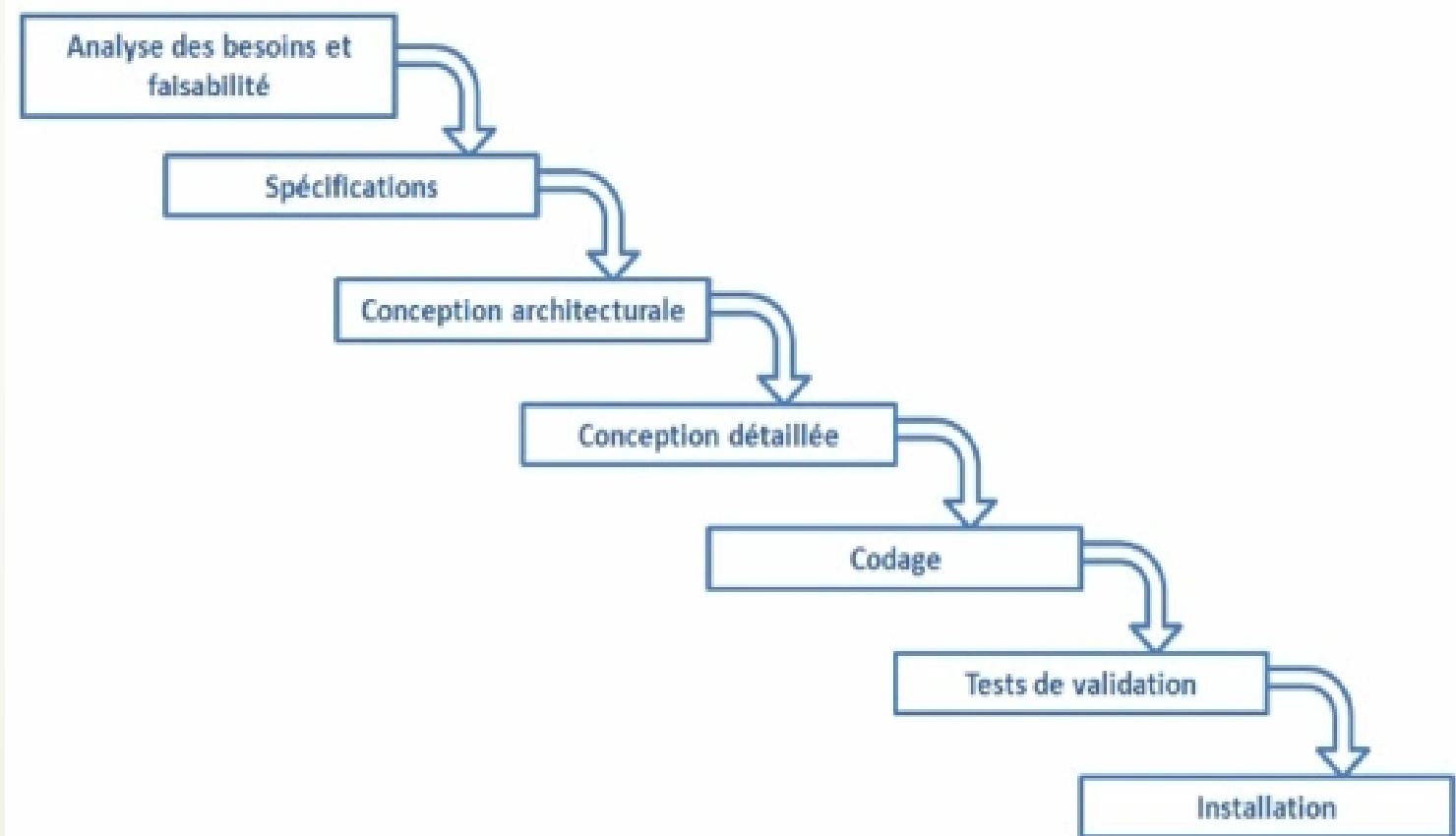
- Cycles prédictifs : Le cycle en cascade

## Failles :

- Activités en silos
- Mauvaise communication
- Documentation pléthorique

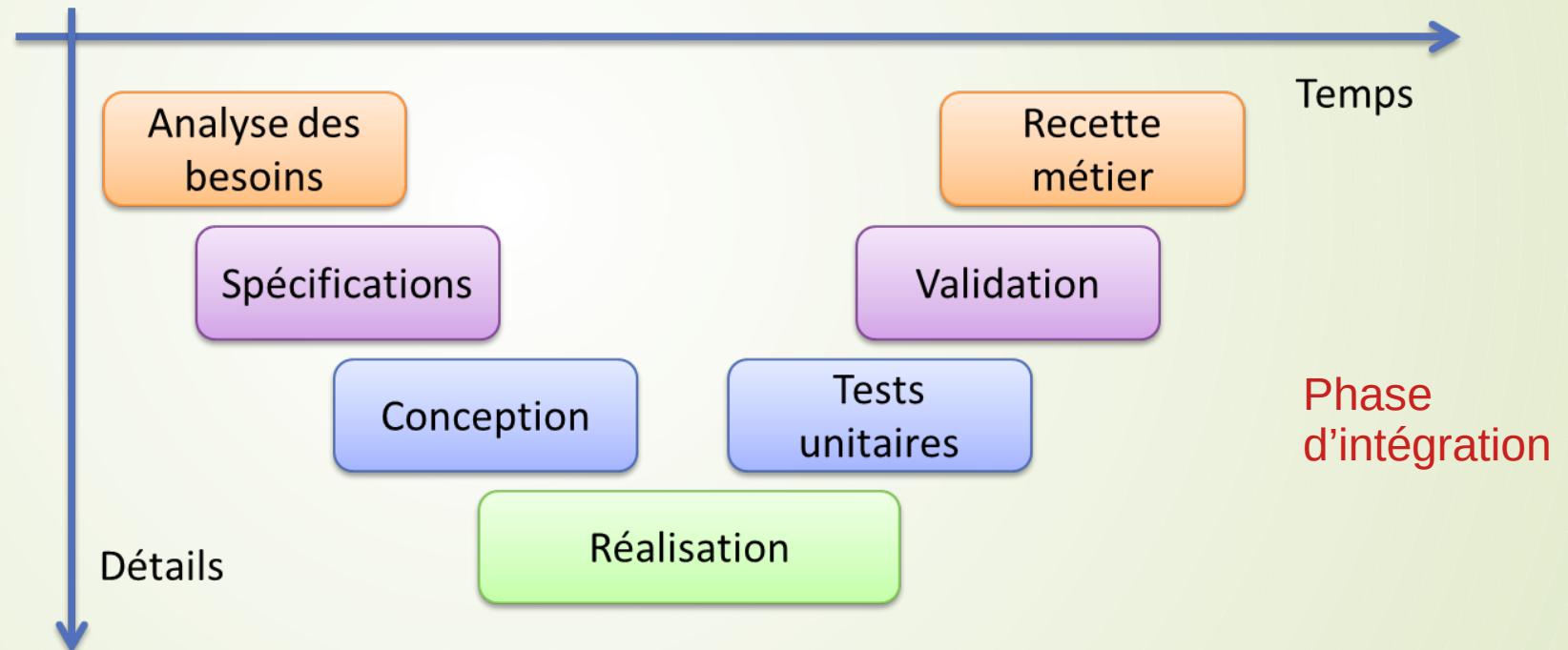
=> Effet tunnel

=> Levée tardive des facteurs de risques



# Vers l'agilité

- Cycles prédictifs Le cycle en V





# Cycles prédictifs

- Budget et plan, **tout est défini**
- La spécification, en amont, doit pouvoir **tout prévoir** et décrire de façon précise et exhaustive
- **Isolement** des équipes et transmission de l'information **unidirectionnel** :
  - À chaque passage de section les erreurs/imprécisions/surcoûts se multiplient



# Cycles prédictifs

- Tunnel : recette métier tout à la fin, opacité de l'avancement
- Périmètre / délais contractuels : la qualité est la variable d'ajustement

# Agilité : Historique

- Années 30, 40 : réflexion sur le cycle de développement **itératif**
- 1976 : Méthode **EVO** « Evolutionary Value Delivery » => **cycles courts**
- 1977 : Jay Galbraith : « **organisational design** » défini comme « un processus de décision visant à aboutir à une cohérence entre les objectifs originels et structurant de l'organisation (la stratégie) et les modes d'organisation du travail et de coordination des unités et du personnel »
- 1986 : développement de **Scrum** : flexibilité, multidisciplinarité, collaboration
- 1991 : développement de **RAD** « Rapid Application Development » : cycle de développement **itératif, incrémental et adaptatif**

# Agilité : Historique

- 2000 : Réunion d'unification (17 fondateurs) des différentes méthodes dans le but de trouver un socle commun de valeurs et de bonnes pratiques.
- Résultats : écriture du **Manifeste pour le développement logiciel Agile** :  
**<http://agilemanifesto.org/iso/fr manifesto.html>**
- et création de l'**Agile Alliance** (association chargée de la promotion de l'agilité et du soutien aux équipes) :  
**<http://www.agilealliance.org/>**

# Manifeste Agile

- Les 4 Valeurs

Nous découvrons comment mieux développer des logiciels par la pratique et en aidant les autres à le faire.

Ces expériences nous ont amenés à valoriser :

- **Les individus et leurs interactions** plus que les processus et les outils
- **Des logiciels opérationnels** plus qu'une documentation exhaustive
- **La collaboration avec les clients** plus que la négociation contractuelle
- **L'adaptation au changement** plus que le suivi d'un plan

Nous reconnaissons la valeur des seconds éléments, mais privilégiions les premiers.





# Les 12 Principes

#1 : Notre plus haute priorité est de satisfaire le client en livrant rapidement et régulièrement des fonctionnalités à grande valeur ajoutée.



# Les 12 Principes

- #2 : Accueillez positivement les changements de besoins, même tard dans le projet. Les processus Agiles exploitent le changement pour donner un avantage compétitif au client.



# Les 12 Principes

#3 : Livrez fréquemment un logiciel opérationnel avec des cycles de quelques semaines à quelques mois et une préférence pour les plus courts.



# Les 12 Principes

- **#4** : Les utilisateurs ou leurs représentants et les développeurs doivent travailler ensemble quotidiennement tout au long du projet.
- **#5** : Réalisez les projets avec des personnes motivées. Fournissez-leur l'environnement et le soutien dont ils ont besoin et faites-leur confiance pour atteindre les objectifs fixés.
- **#6** : La méthode la plus simple et la plus efficace pour transmettre de l'information à l'équipe de développement et à l'intérieur de celle-ci est le dialogue en face à face.



# Les 12 Principes

#7 : Un logiciel opérationnel est la principale mesure d'avancement.



# Les 12 Principes

- #8 : Les processus Agiles encouragent un rythme de développement soutenable. Ensemble, les commanditaires, les développeurs et les utilisateurs devraient être capables de maintenir indéfiniment un rythme constant.
- #9 : Une attention continue à l'excellence technique et à une bonne conception renforce l'Agilité.



# Les 12 Principes

#10 : La simplicité - c'est-à-dire l'art de **minimiser la quantité de travail inutile** - est essentielle.



# Les 12 Principes

- #11 : Les meilleures architectures, spécifications et conceptions émergent d'équipes auto-organisées.
- #12 : À intervalles réguliers, l'équipe réfléchit aux moyens de devenir plus efficace, puis règle et modifie son comportement en conséquence.



# Attention !

- Appliquer un framework agile ne rend pas votre organisation agile
- Vous pouvez être agile sans utiliser de « méthodes agiles »

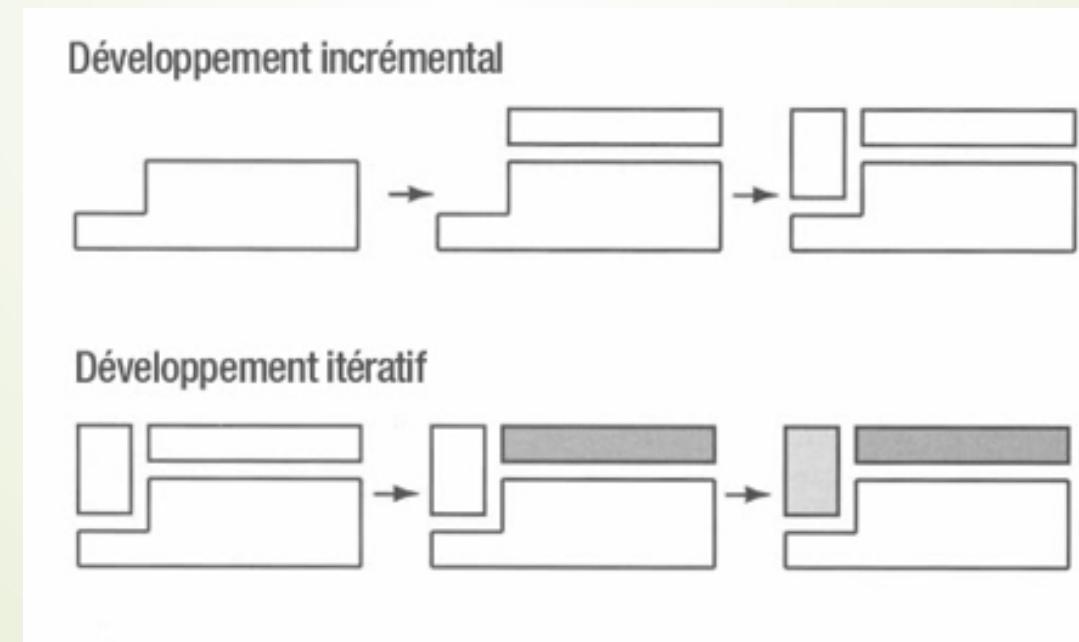


# En résumé

- Livrer en continu
- Des logiciels fonctionnels
- En s'adaptant au changement

# Méthodes Agiles

- Une méthode Agile est une **approche itérative, incrémentale et adaptative** qui est menée dans un **esprit collaboratif**, avec le minimum de **formalisme**.
- Elle génère un produit de **haute qualité** tout en prenant en compte l'**évolution des besoins du client**.



# Méthodes Agiles

- Itérative ET Incrémentale...

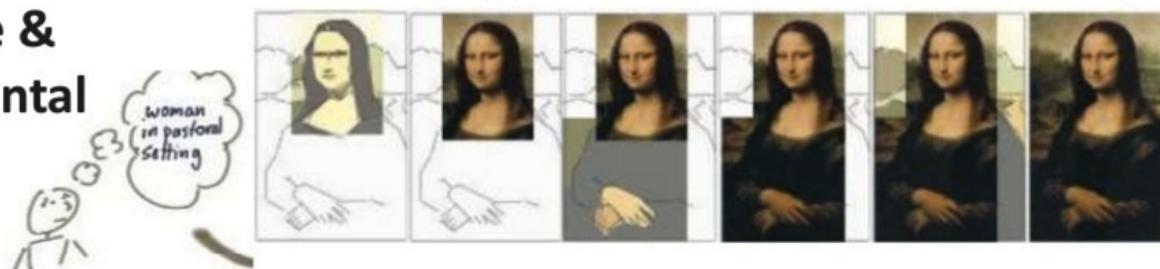
**Iterative**



**Incremental**

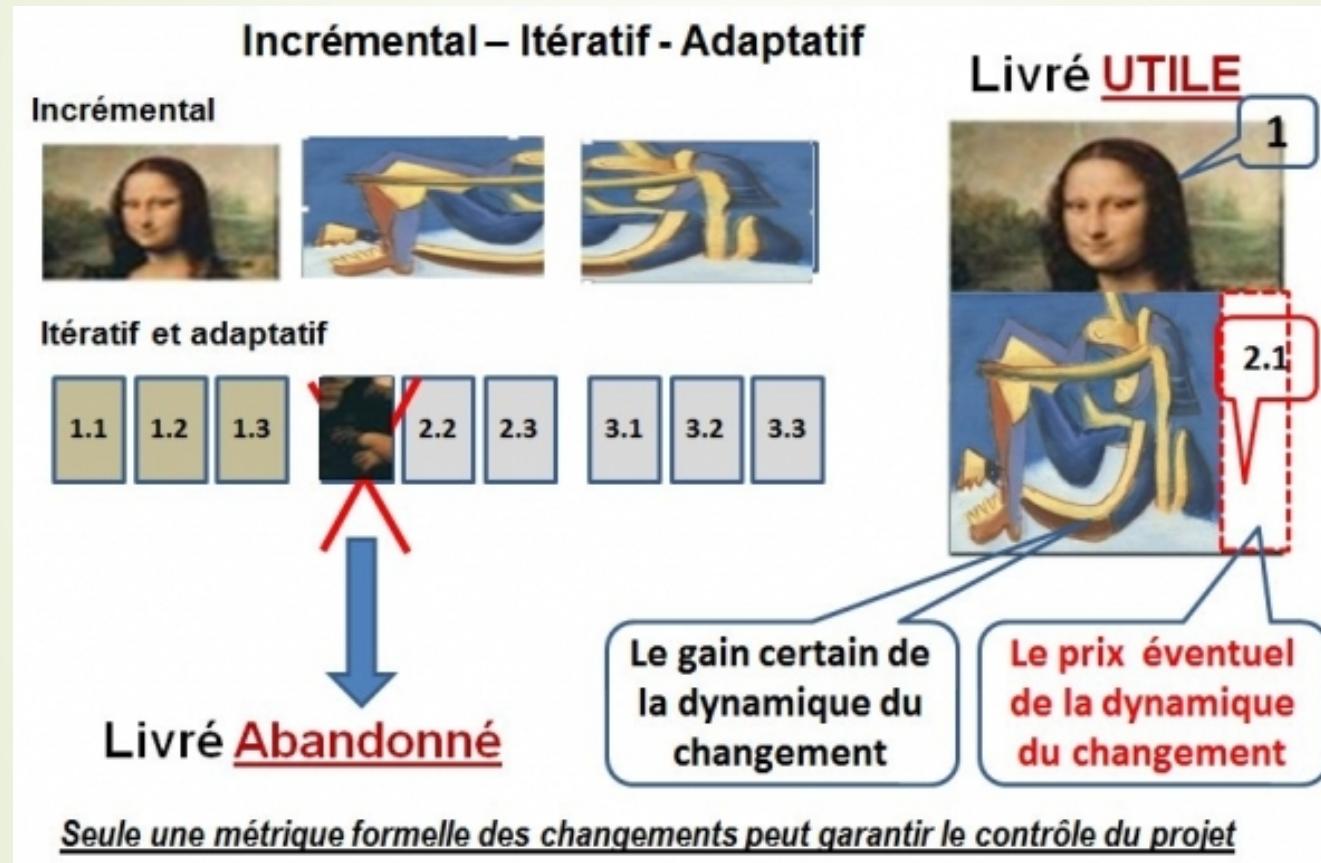


**Iterative &  
Incremental**



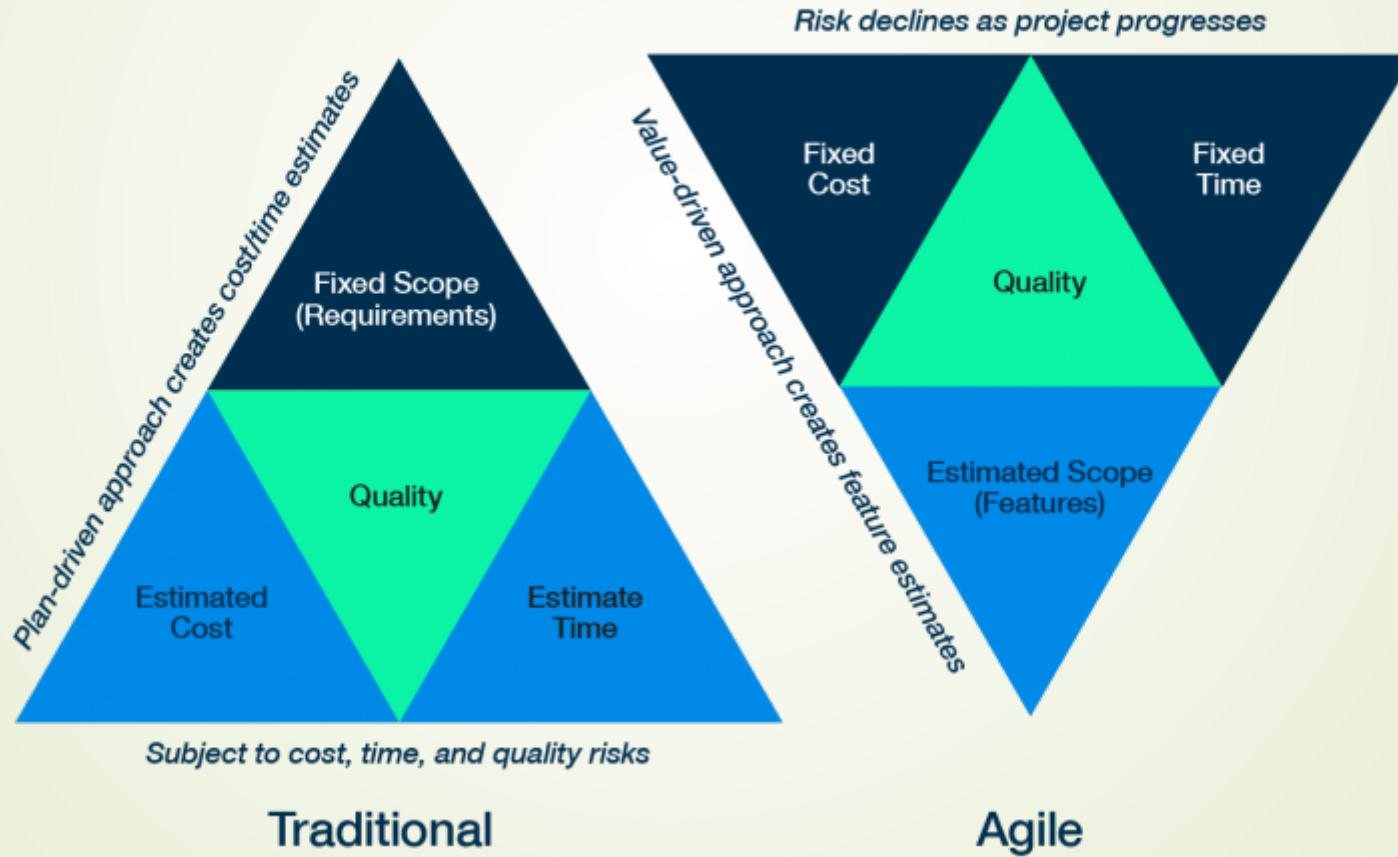
# Méthodes Agiles

- ... ET Adaptive



# Méthodes Agiles : intérêts

## Iron Triangle Paradigm Shift



# Méthodes Agiles : intérêts

- Réduire le temps de mise sur le marché (**Time to market**)
- Améliorer la qualité du produit
- Réduire les activités sans réelle valeur ajoutée  
=> Travailler sur des éléments de plus grande valeur
- Avoir une meilleure prévisibilité
- Améliorer le cadre de travail

# Méthodes Agiles : quand ?

## Favorisants :

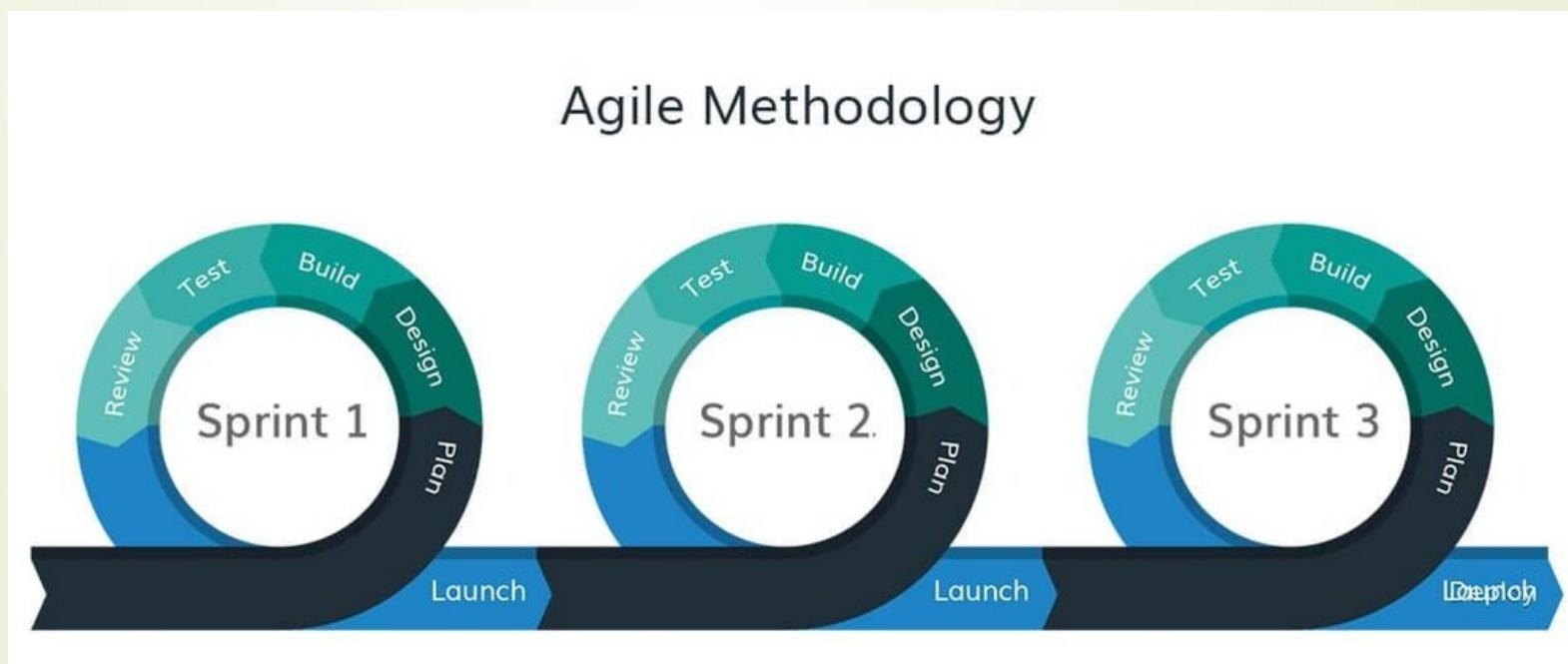
- Besoin rapide de mise à disposition du produit
- Imprévisibilité des besoins du client
- Nécessité de changements fréquents
- Besoin de visibilité du client sur l'avancement des développements
- Présence de l'utilisateur assurant un feedback immédiat

## Défavorisants :

- Indisponibilité du client ou de l'utilisateur
- Dispersion géographique des ressources humaines
- Inertie des acteurs du projet ou refus des changements
- Gouvernance complexe de la DSI

# Le cycle Agile

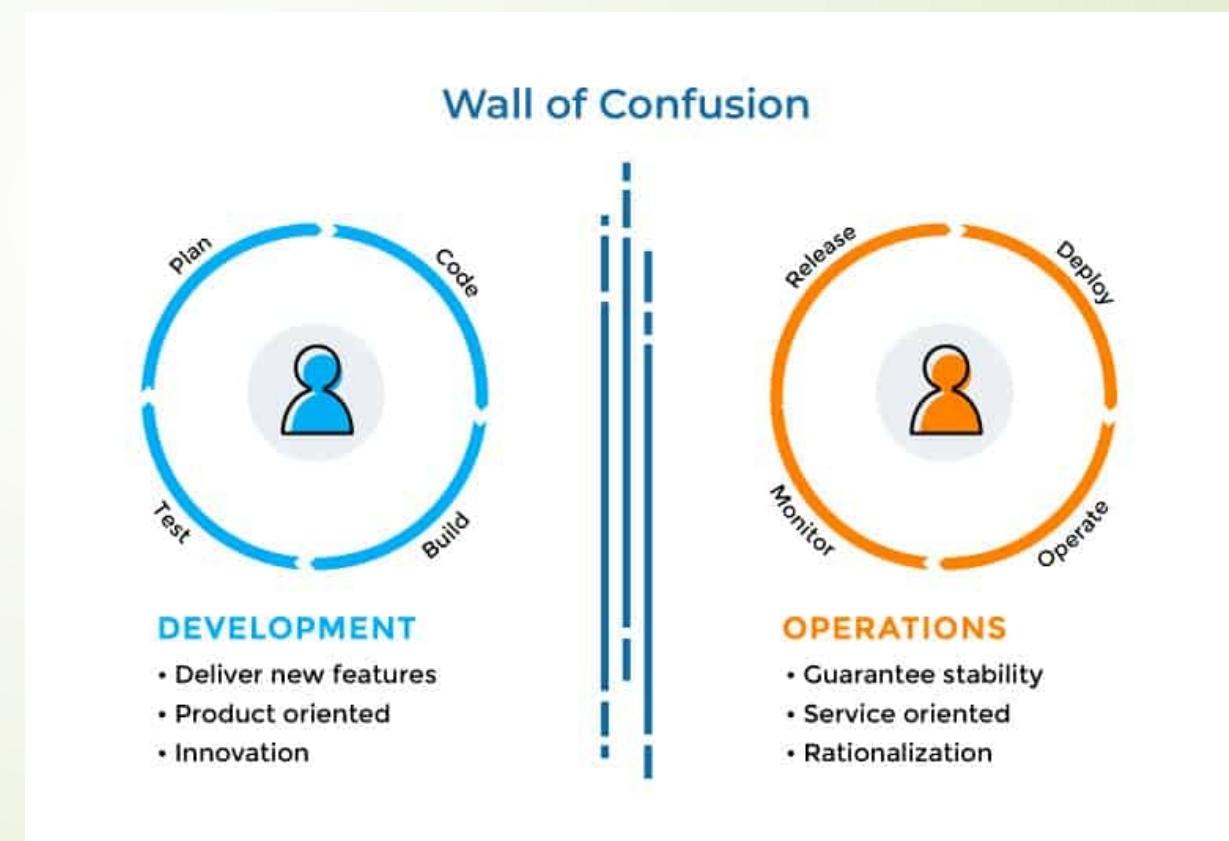
- Sprint : itérations sur un incrément
- Chaque sprint représente un cycle de développement
- Augmente la fréquence de livraison



# Introduction DevOps

- Besoin d'Agilité dans la relation Dev / Ops
  - Stéréotypes Dev      VS      Stéréotypes Ops

C'est pour résoudre ce problème, dans le contexte de la conférence Agile 2008 à Toronto sur l'infrastructure agile, que le mot « **DevOps** » a été employé pour la première fois par l'ingénieur système belge Patrick Debois



# Introduction DevOps

- Atteindre une organisation DevOps
  - Se recentrer sur les relations humaines ...
    - construire une **culture de la collaboration** entre Devs, Ops, et toutes les parties prenantes aux projets. Les actions de chacun sont améliorées par la compréhension des besoins des autres
      - appliquer une **gouvernance collective et responsabilisante** au sein de l'équipe DevOps
      - agréger les compétences en formant des **squad DevOps multidisciplinaires**
  - ... grâce à l'automatisation de la production
    - automatiser les opérations techniques amenant un code développé à l'état de livrable, puis automatiser son déploiement, ainsi que les opérations de gestion des serveurs => **Intégration / livraison /déploiement continu, gestion de configuration, orchestration**
    - automatiser les remontées d'informations des outils jusqu'aux acteurs pour alerter et faciliter les décisions => **monitoring, conformité, analyse qualité**

# Introduction DevOps

- Atteindre une organisation DevOps
  - Associer les acteurs au moyen d'un **management horizontal**
    - « Le Product Owner », qui possède la vision du produit et qui exprime le besoin, doit pouvoir observer régulièrement l'avancée du projet au rythme de la diffusion de livrables ; il est soit client, soit en interface avec le client
    - « Le Scrum Master » (ou assimilé) est le garant du cadre méthodologique de l'équipe. Il doit veiller à ce que les processus agiles mis en place soient respectés. Ce n'est pas nécessairement le chef de projet
    - L'« Ops Master » est le relais privilégié de l'équipe Ops auprès de l'équipe Dev sur les pratiques et outils DevOps

# Introduction DevOps

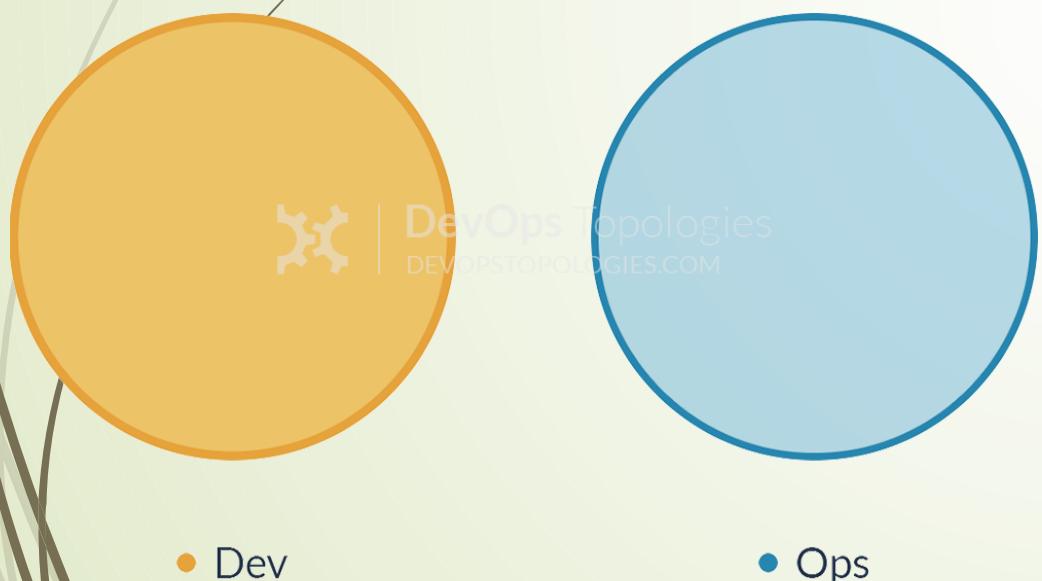
## • Les rôles DevOps

- L' « évangéliste » : rompu aux pratiques DevOps, forme et conseille les membres de l'équipe
- Le « release manager » : responsable principal des processus DevOps mis en oeuvre
- L' « automation architect » : s'occupe de l'automatisation des processus DevOps
- Le développeur / testeur : en charge de l'évolution du produit
- L'ingénieur Qualité : évalue la qualité du code, les performances l'expérience utilisateur « **UX** »
- L'ingénieur sécurité : fournit les bonnes pratiques de sécurité aux développeurs et aux opérateurs au coeur des cycles de développement
- L'opérateur : Administrateur système et réseau en charge de la stabilité des environnements
- Ces rôles sont cumulatifs et distribués : ils forment un **squad**, divisés en « **tribus** », dans les grandes organisations

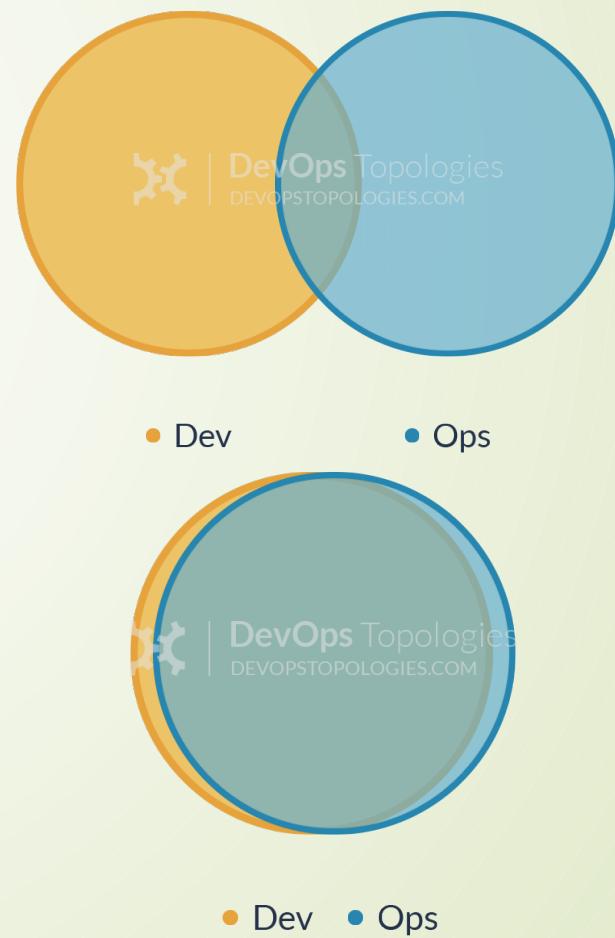
# Introduction DevOps

- Topologies DevOps

NOK



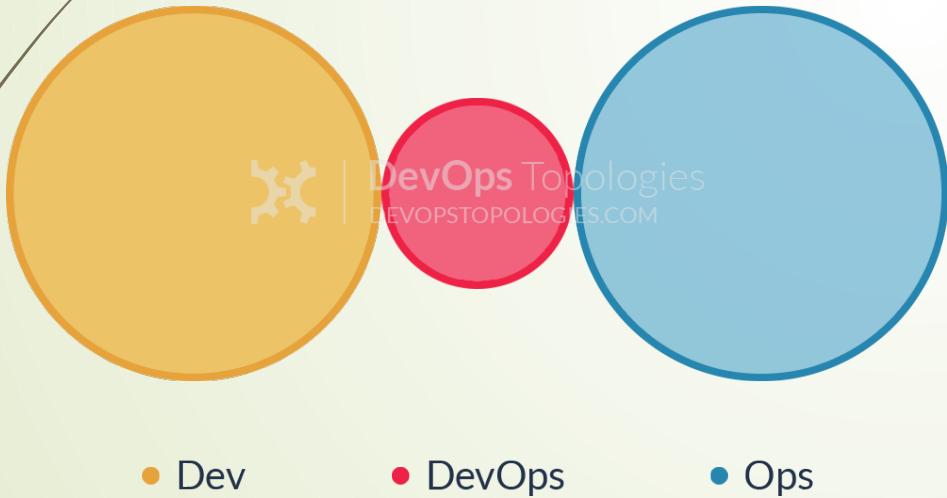
OK



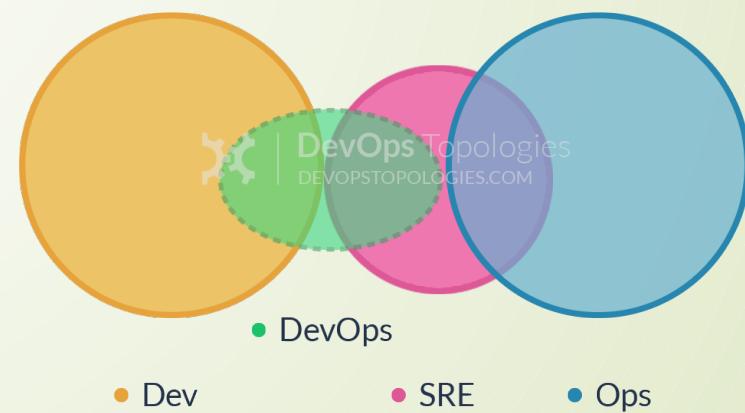
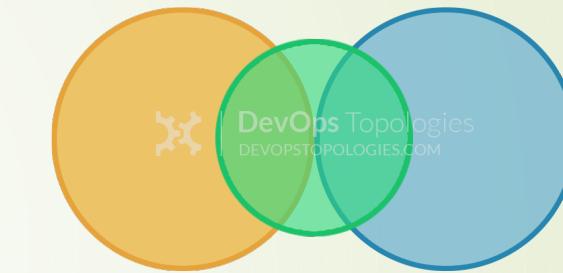
# Introduction DevOps

- Topologies DevOps

NOK



OK



# Introduction DevOps

## • Sociocratie, Holacratie

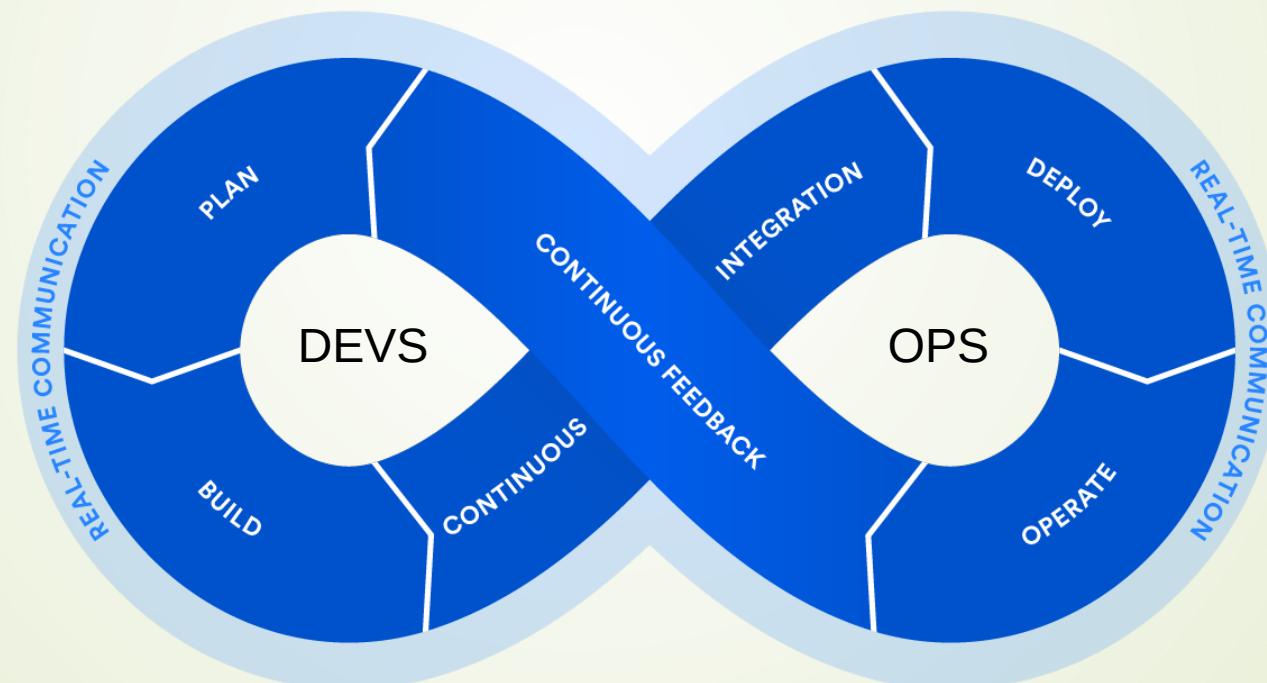
- Sociocratie : du latin *socios* « Alliés » et du grec *kratos* « pouvoir, force », désigne un mode de gouvernance de structures auto-organisées rassemblant des individus autour d'objectifs communs, partageant les responsabilités.
- quatre concepts clés
  - **le consentement** : aucune décision stratégique n'est prise tant qu'existe une objection raisonnable
  - **les cercles** : chaque structure opérationnelle associe un cercle de décision rassemblant tous ses membres, décidant du pilotage, de l'exécution et de la mesure des processus mis en place.  
les cercles sont organisés de manière **hierarchique** sur le plan de la satisfaction des objectifs
  - **le double lien** : les communications ascendante et descendante entre cercle sont assumées par deux membres distincts de chaque cercle
  - l'attribution des rôles et des tâches s'effectue par **élections sans candidats**. Le premier candidat dont la proposition par un membre est consentie par le cercle est élu
- Holacratie : du grec *olos* : « tout », désigne une variante de la sociocratie dans laquelle la hiérarchie est remplacée par un système de réunions pendant lesquelles l'autorité (**gouvernance**) et les tâches (**triage**) sont réparties sur les membres des cercles, auxquels sont attribués des **rôles**. Elle intègre également des concepts Agiles et Lean d'adaptation appliqués à la structure organisationnelle, autour de la notion de **raison d'être**

# Introduction DevOps

- Shift left, Shift Right
  - Le Shift Left représente la volonté insérer les actions de la branche droite du V parmi celle de la branche gauche. Cela se traduit de deux façons principales :
    - publier des outils en libre accès permettant aux clients / utilisateurs de pouvoir s'informer sur les produits / projets / processus du service IT, ce qui dégage du temps aux équipes pour se concentrer sur des tâches plus critiques
    - insérer les tests et la sécurité dès la phase de développement en rappelant les bonnes pratiques et mettant en place des techniques de développement guidés par les tests (TDD, BDD)
  - Le Shift Right, par analogie, consiste à analyser le comportement d'une version en production pour préparer la prochaine version du produit :
    - surveillance, mesure de la performance, comportements utilisateurs

# Introduction DevOps

- Le cycle DevOps





# Lien avec l'Open Source

- Rappels

- Logiciel libre : logiciel dont le code est libre d'accès, d'utilisation, de modification et de redistribution
- Logiciel libre ne signifie pas logiciel gratuit
- l'Open Source stricto sensu désigne la méthodologie de développement des logiciels libres
- Un logiciel Open Source est conçu par une communauté de développeurs indépendants les uns des autres
- Les valeurs principales portées par l'Open Source
  - l'indépendance vis-à-vis des fournisseurs : le code est toujours disponible
  - la collaboration ouverte : chacun peut examiner les travaux de tous => effort de lisibilité
  - la flexibilité du code : un code open source peut aboutir à plusieurs solutions logicielles (« fork »)
  - la fiabilité du code : le code est commenté, revu et testé par l'ensemble de la communauté
  - la créativité : l'indépendance des développeurs favorise l'innovation en continu

# Lien avec l'Open Source

- Dans DevOps, l'équipe est vue comme une communauté d'acteurs pluridisciplinaire
- Dans DevOps, des outils sont mis en place pour favoriser une collaboration harmonieuse
- Dans Devops, tous les acteurs sont responsables du projet et du produit
- Dans DevOps, la réactivité au changement est cruciale
- Dans DevOps, l'amélioration Continue est un objectif permanent

# Relation avec le LEAN

## • Le « LEAN Management »

- 50's : méthode de management industriel mis en place chez Toyota « **Toyotisme** »
- Basé sur l'exploitation des connaissances et de la créativité des acteurs de terrain « **bottom / up** ou **gemba** »
- Conçu pour réaliser des flux de production **Juste à Temps** (cf Kanban)
- ... en éliminant les formes de gaspillage : « **muri** » (excès), « **muda** » (inutile), « **mura** » (non conforme)
- Utilise des outils conceptuels :
  - « **5S** » : étapes de rationalisation d'une tâche ( débarrasser, ranger, nettoyer, standardiser, discipliner)
  - les « **5 pourquoi** » : questionnement approfondi pour remonter aux **causes initiales** d'une défaillance
  - Calcul des **KPI**, Indicateurs clés de performance
- - « **kaizen** » : amélioration continue, PDCA « Plan, Do, Check, Act »  
<https://www.leanproduction.com/top-25-lean-tools.html>
- 2 objectifs parallèles : la satisfaction client et l'épanouissement des travailleurs

# Relation avec le LEAN

- LEAN Software Development
- Application du LEAN dans un contexte IT
- Le gaspillage représente toute activité n'apportant aucune valeur au produit du point de vue client
- La formation, la pluridisciplinarité et la responsabilisation des équipes sont favorisées
- Retarder les décisions (conserver des options) et livrer vite => développement itératif
- Engager la démarche qualité dès la conception  
=> qualité globale du projet et de l'environnement projet

# Bénéfices de DEVOPS

- Pour l'IT

- Fiabilisation du cycle de vie des services (confiance)
- Concentration des équipes sur les tâches à valeur ajoutées
- Réduction des **temps de réparation** (incident) et des **temps de mises en œuvres** (fonctionnalités)
- Meilleure Réactivité au changement
- Incontournable pour les projets big data : on ne peut tester qu'en environnement de prod



# Bénéfices de DEVOPS

- Pour l'entreprise

- Meilleur **time to market** par l'augmentation de la livrabilité => avantage concurenciel
- Augmentation de la production
- Baisse des coûts de production à moyen terme et de maintenance à court terme
- Meilleur qualité de vie au travail et meilleures collaborations entre services
- Meilleur prévisibilité des roadmap

# Cas : L'agilité chez Netflix

- Netflix ne produit pas d'épisodes pilotes. Au lieu de perdre un temps et des ressources considérables à produire un épisode dont le succès n'est pas garanti, Netflix s'appuie sur la collecte et l'analyse de données pour prédire les habitudes des utilisateurs, augmenter les taux de réussite et créer des programmes qui généreront une fréquentation maximale.
- Netflix prend constamment en compte le feedback clients. Ainsi, la plateforme est constamment consciente du positionnement de ses utilisateurs et peut s'adapter en temps réel pour que tout le monde soit gagnant.
- Netflix est diffusable sur 900 interfaces différentes, ce qui signifie que l'entreprise maîtrise parfaitement l'usage des tests en navigateurs croisés en fonction des demandes des consommateurs.

# Cas : L'agilité chez Netflix

- Netflix privilégie les sorties fréquentes et régulières. La plateforme crée du contenu qu'elle diffuse en cycles plus courts, comme le préconise la méthode agile de gestion de projets
- Netflix fonctionne sur la base d'objectifs agiles détaillés qui prennent en compte chaque spectateur, en créant du contenu, des recommandations et analyses personnalisées basées sur l'historique de navigation et le parcours du client.
- Chiffres :
  - 193M membres
  - 190 pays
  - 15% de la bande passante mondiale en 2018

# Synthèse sur Devops

- DevOps n'est pas un produit, une méthode ou un métier
- C'est tout à la fois une culture, des pratiques et des outils
- Crées par des techniciens pour des techniciens et issus de leur expérience
- Originaires du monde de l'IT, mais généralisables à tous les processus de production