

DEVOPS

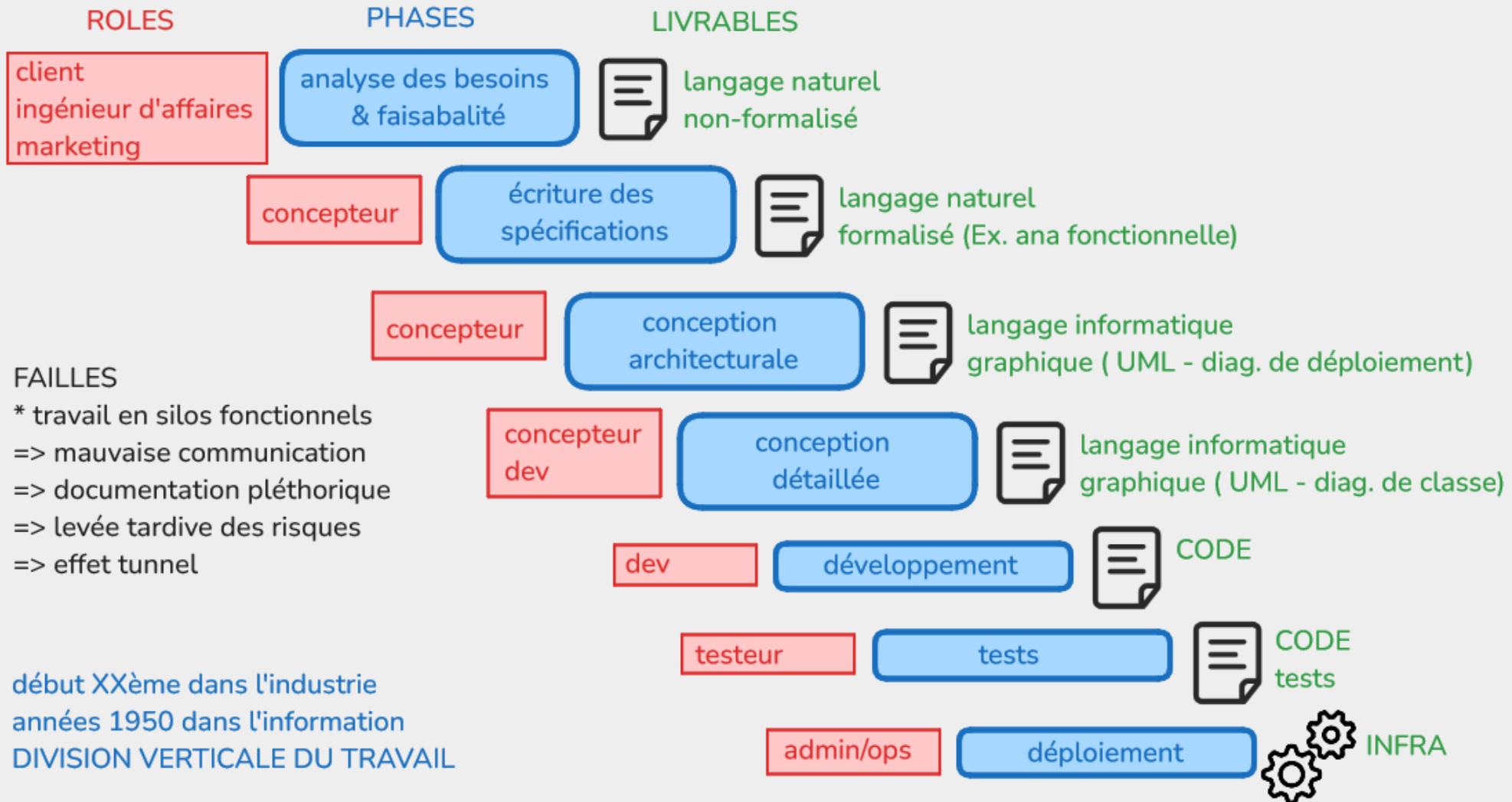
I. Vers l'agilité

II. DevOps historique

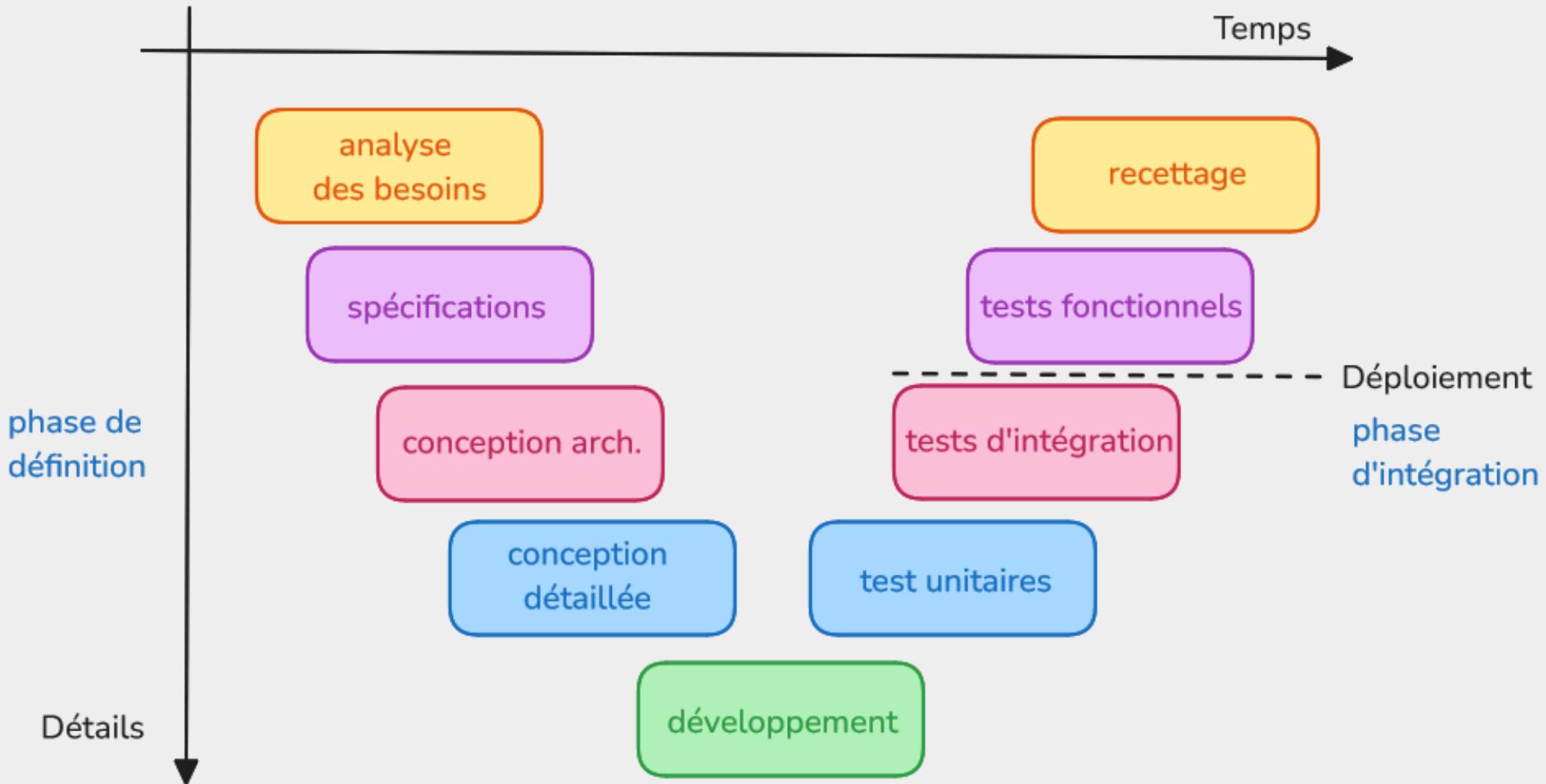
III. Généralisation de DevOps

I. Vers l'agilité

modèle de développement prédictif en cascade



modèle de développement prédictif en V



cycles prédictifs

- **budget, plan et périmètre** - tout est défini *ab initio*
 - les spécifications, *en amont*, doivent être *précises et exhaustives*
- **isolement** des équipes, communication *unidirectionnelle*
- entre les phases les *erreurs et imprécisions* - et donc les **retards et les surcoûts** se multiplient
- effet tunnel: recettage tout à la fin => **opacité de l'avancement**
- scope et délais contractuels : la **qualité** est la variable d'ajustement

historique de l'agilité

- Années 30, 40 : réflexion sur le cycle de développement **itératif**
- 1976 : Méthode EVO => **cycles courts**
- 1977 : Jay Galbraith : « organisational design »
 - **auto-organisation** des *opérationnels* soumis à l'atteinte des *objectifs* livrés par les *décisionnels*
- 1986 : **Scrum** - *adaptativité, multidisciplinarité, collaboration*
- 1991 : RAD - cycle itératif, **incrémental** et adaptatif

1980's : l'ITSM / ITIL

- Information Technology Service Management
 - IT Infrastructure Library
 - cadre technique **complexe**: *40 livres de processus* en ITIL V1
 - v2 (2002) *simplification* - 7 livres
 - v3 (2007/11) **intégration de LEAN/Agile/DevOps** - 5 livres
 - v4: 2019: *modernisation* - LEAN/Agile/DevOps natifs
- “ l'IT est vue comme un **fournisseur de services** répondant aux *besoins métier* des **utilisateurs/clients** ”

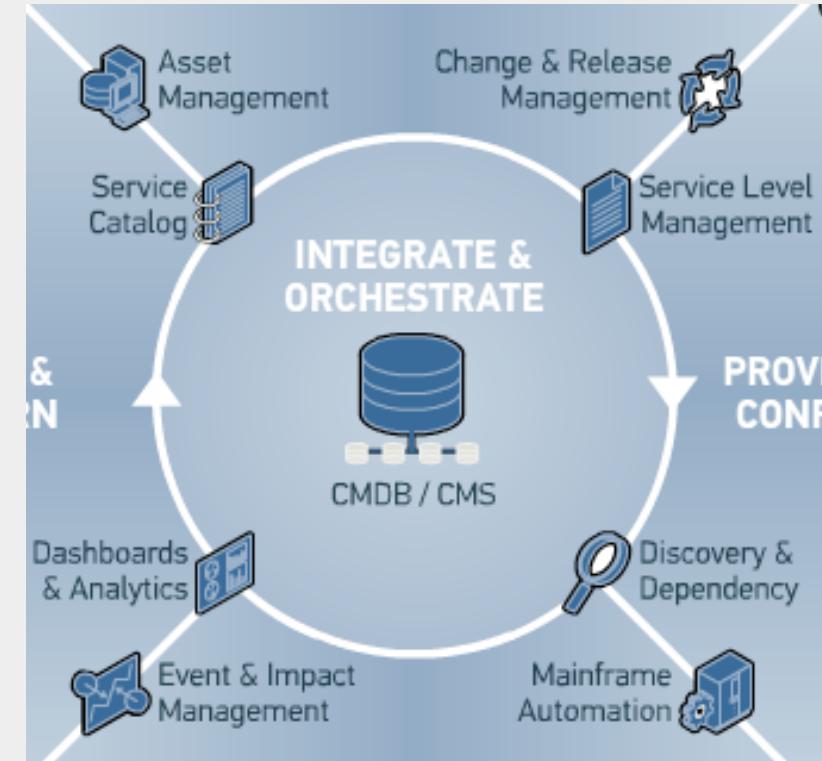
ITSM/ITIL: Le vocabulaire usuel de l'IT !

- **processus**: ensemble d'activités contribuant aux objectifs d'une organisation = **Flux de travail** « *Workflow* » = **Flux métier**
- **Service** : *moyens* utilisés pour produire de la **Valeur** pour un **client** sans lui faire supporter les *risques* associés
- Ex : Gestion des incidents - *Incident / SLA / MTRS*
- Ex: Gestion des changements - *RFC / Rollback*
- Ex: Autres - *Ticket / Service Desk / KPI / Post-Mortem*
- V4: Gestion des services Agiles « **Agile Service Management** »

ITSM/ITIL : CMDB

- BDD de *gestion des configurations*
- aujourd'hui décentralisée avec
 - inventaire réseau IPAM / OCS
 - référentiel personnes: LDAP / AD
 - portfolio de services: gitlab/hub...
 - capacités / règles: SLAs, compta...
- utilisée par des **processus**

“ *composant emblématique de l'ITSM* ”



Manifestation Agile

- 2000 : Réunion de *factorisation* des méthodes précédentes
 - déduction d'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>
 - création de l'Agile Alliance - association chargée de la promotion de l'agilité et du soutien aux équipes:
<http://www.agilealliance.org/>

préambule

- **individus et leurs interactions** >>> *processus et outils*
- **logiciels opérationnels** >>> *documentation exhaustive*
- **collaboration clients** >>> *négociation contractuelle*
- **adaptation au changement** >>> *suivi d'un plan*

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

12 principes agiles

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

#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*.

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

#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**.

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

#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.

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

#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.

“ Appliquer une méthode agile ne rend pas votre organisation agile
Vous pouvez être agile sans utiliser de méthodes agiles ”

méthode agile

- Une méthode Agile est une approche **itérative, incrémentale et adaptative**, 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**.

itératif ET incrémental

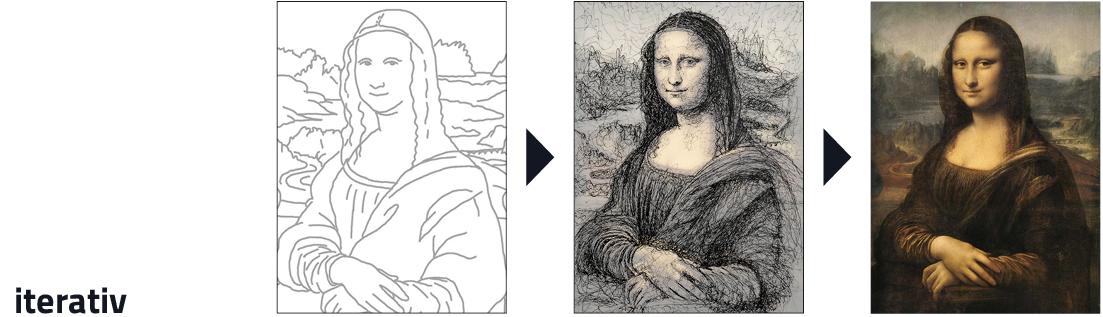
montée en versions /

- itérations => qualité

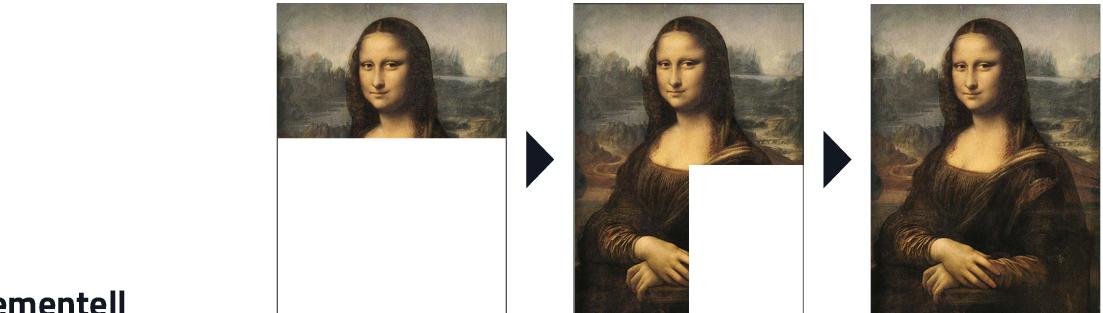
découpe en incrément /

- objectif => valeur ajoutée

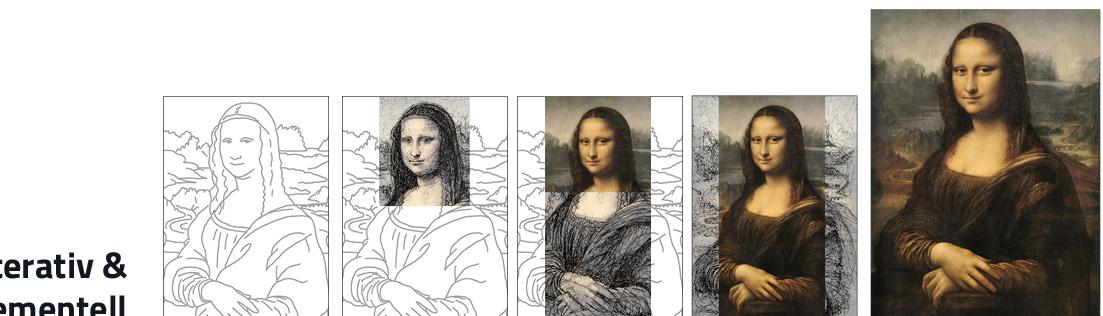
choix selon la
satisfaction client !!!



iterativ

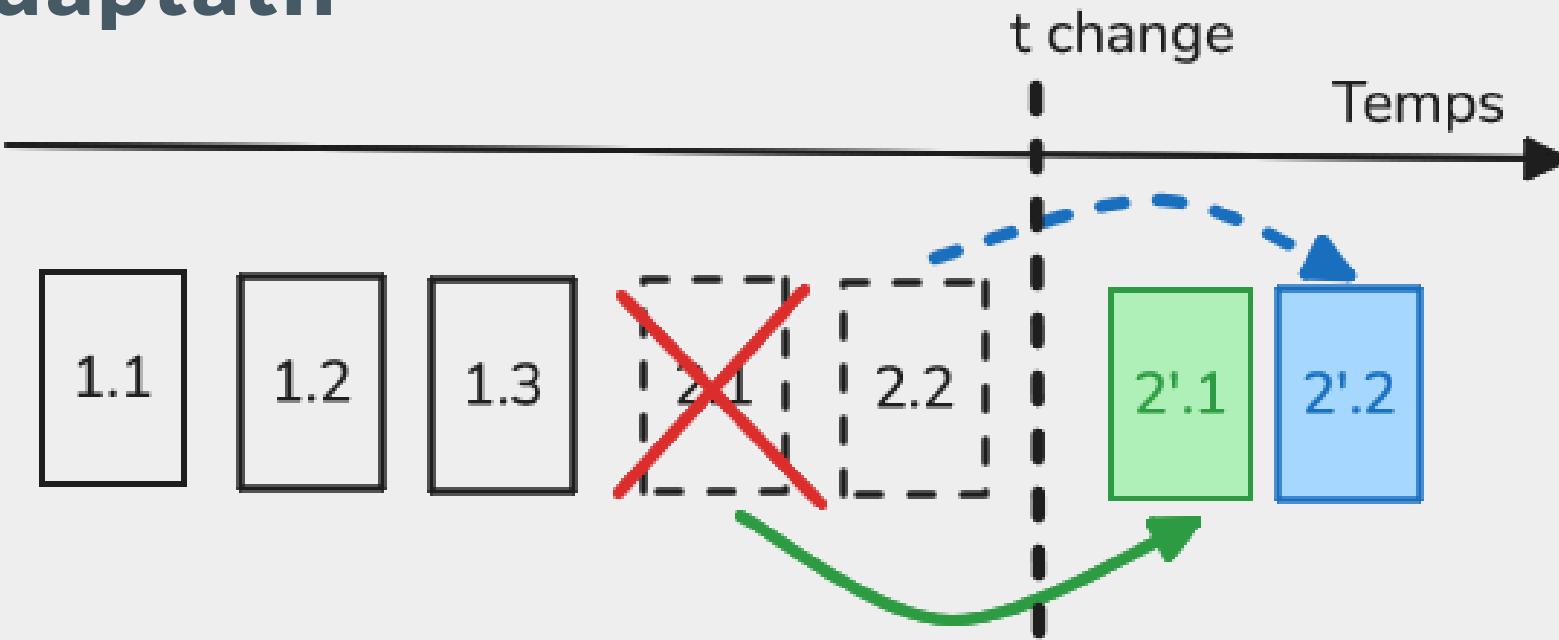


inkrementell



iterativ &
inkrementell

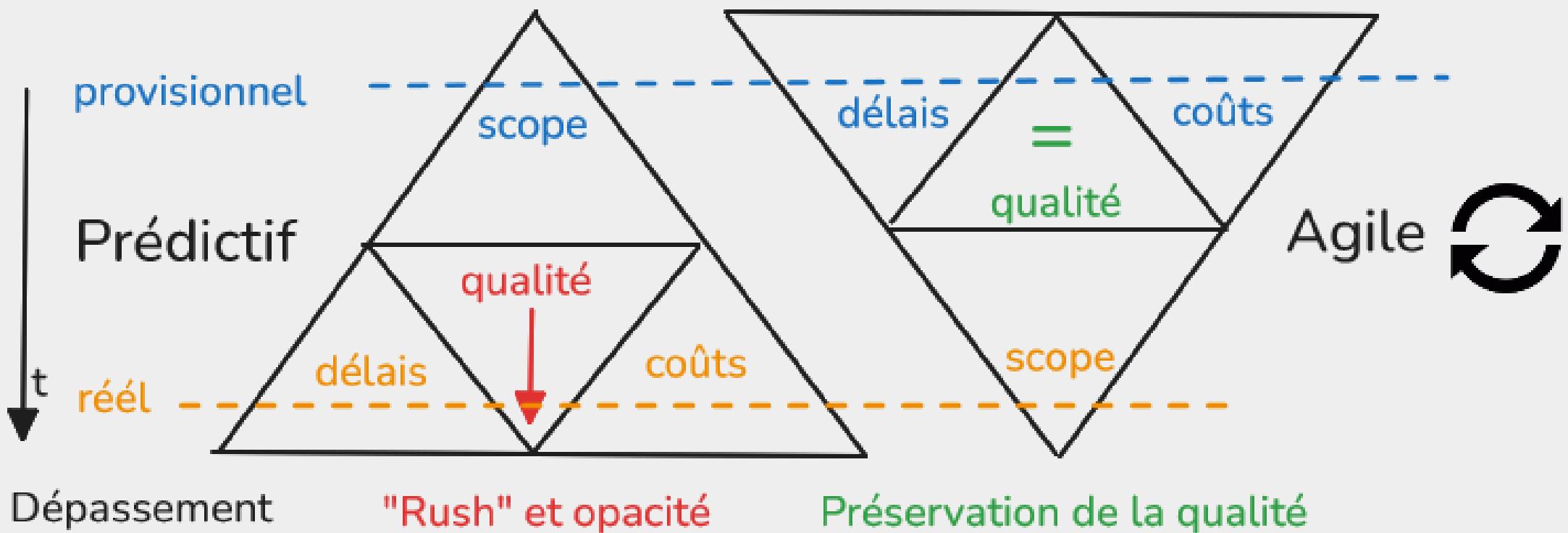
adaptatif



Enjeux

- I. cycles courts => nb min. de changements de 2.1
- II. Collaboration client => nb min. de changement 2.x

basculement des contraintes projet

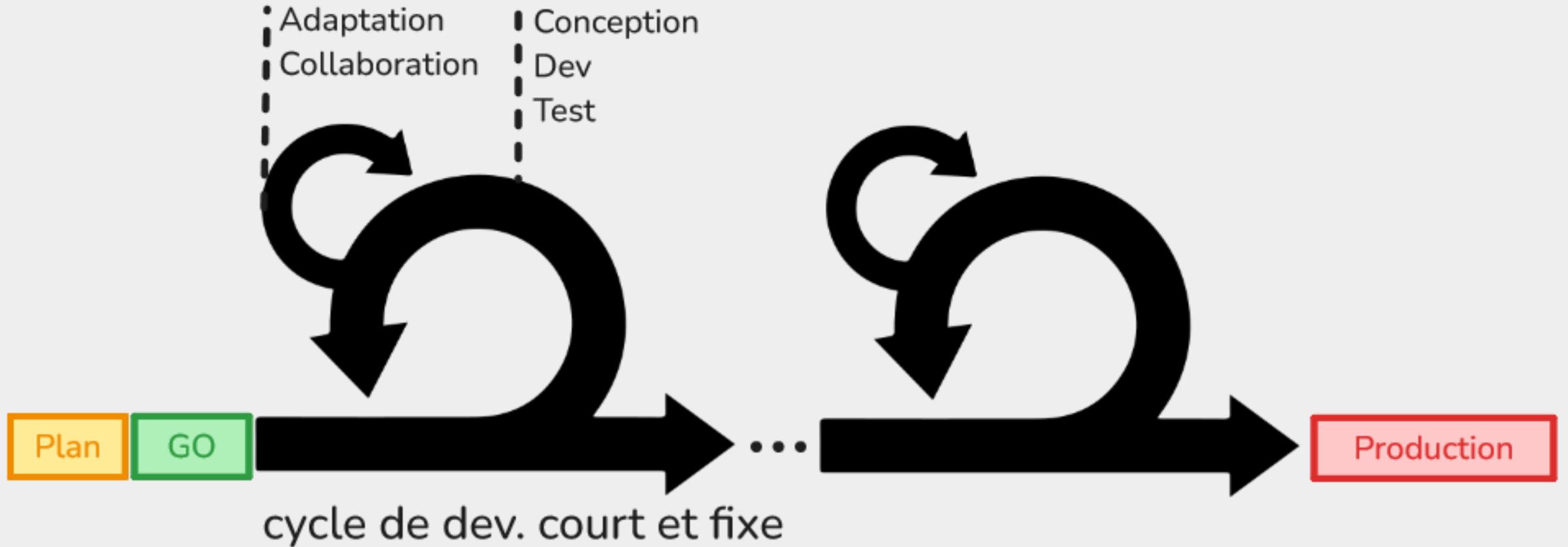


“ loi de hofstadter: Il faut toujours plus de temps que prévu, même en tenant compte de la loi de Hofstadter.

défavorisants de l'agilité

- **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 DSIT

Cycle agile sans OPS

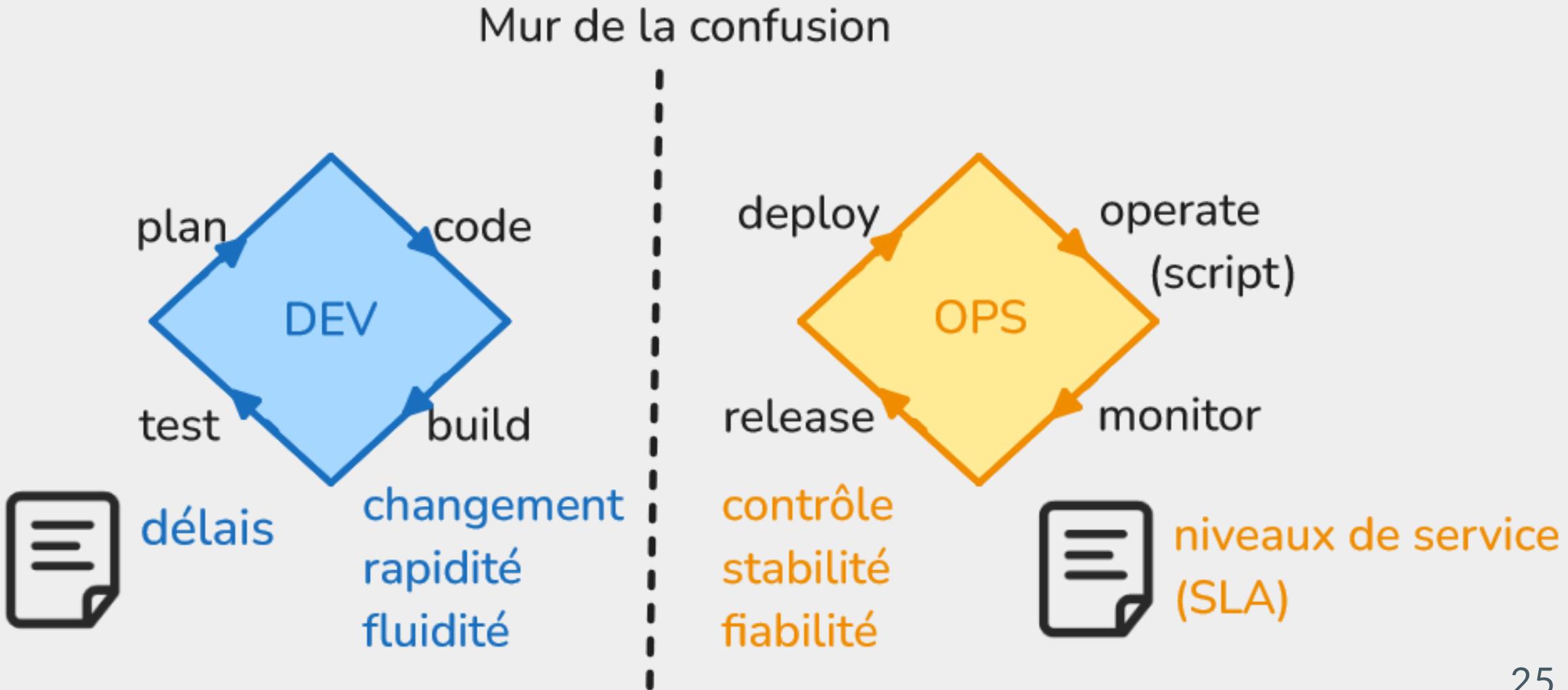


II. DevOps historique

extension de l'agilité aux Opérations

- **2007:** *Patrick Debois*, consultant admin/sys pour le gouvernement belge, décrit une migration de données problématique, causée par une *mauvaise communication Dev / Ops*
- **2008:** conférence d'*Andrew shafer* consultant dev., sur l'infrastructure agile
 - => les deux créent le « *Agile Systems Administration Group* ».
 - => rapidement rebaptisé en « **DevOps** »

stéréotypes dev et ops



atteindre un cycle DevOps

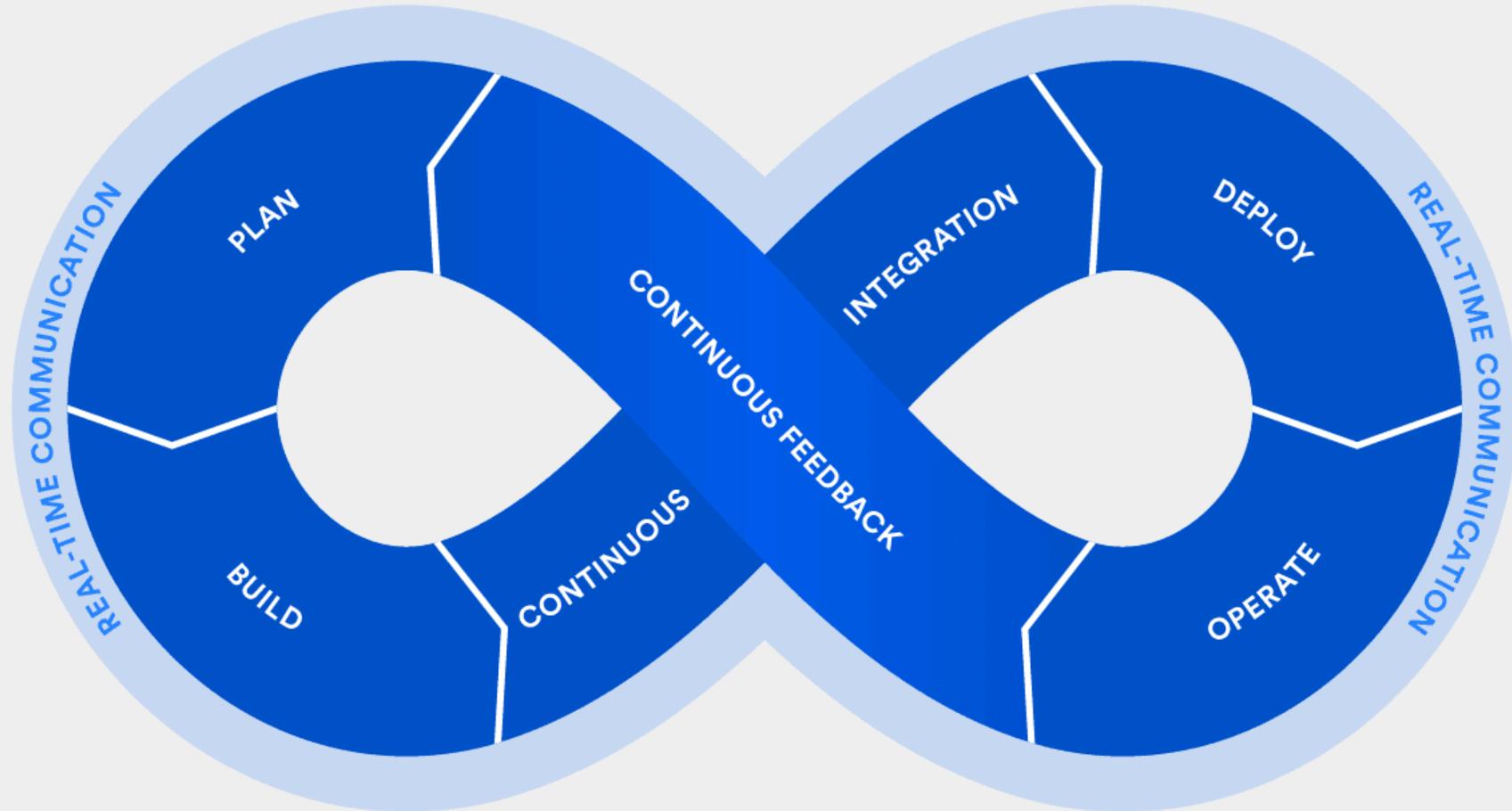
1. Se concentrer sur les relations humaines ...

- **collaboration** Dev / Ops quotidienne + adopter un *vocabulaire/nomenclatures communs*
- privilégier des profils aux *compétences croisées* => **squad multidisciplinaire**

2. grâce à l'**automatisation** de la production de valeur

- Dev: *Intégration / Livraison / Déploiement Continue*
=> usine logicielle i.e, pipeline de jobs i.e, **SCRIPTS !!!**
- Ops: *Gestion de configuration, Orchestration, Monitoring*
=> INFRASTRUCTURE AS **CODE !!!**

Cycle DevOps



III. Généralisation de DevOps

Définition globale de DevOps

- « DevOps est avant tout un **mouvement culturel** fondé sur des interactions *humaines et techniques* destinées à améliorer les *relations et les résultats* d'une organisation. »
- résumée par l'acronyme **CALMS**
 - **Culture**
 - **Automatisation**
 - **Lean => Optimisation**
 - **Mesure**
 - **Solidarité**

généralisation

- « **Dev:** » ressource ayant une activité de *Création de Valeur*
 - « **Ops:** » ressource ayant une activité de *Perennité de la Valeur*
 - **valeur:**
 - ROI: Retour Sur Investissement
 - satisfaction client
 - avantage concurrentiel
 - bien-être au travail
- “ *DevOps transforme une organisation en un pipeline automatisé de valeur actionné par les parties prenantes !!!* ” 30

organisation DevOps



Culture

Personnes

Tous

Processus

Agile / Lean

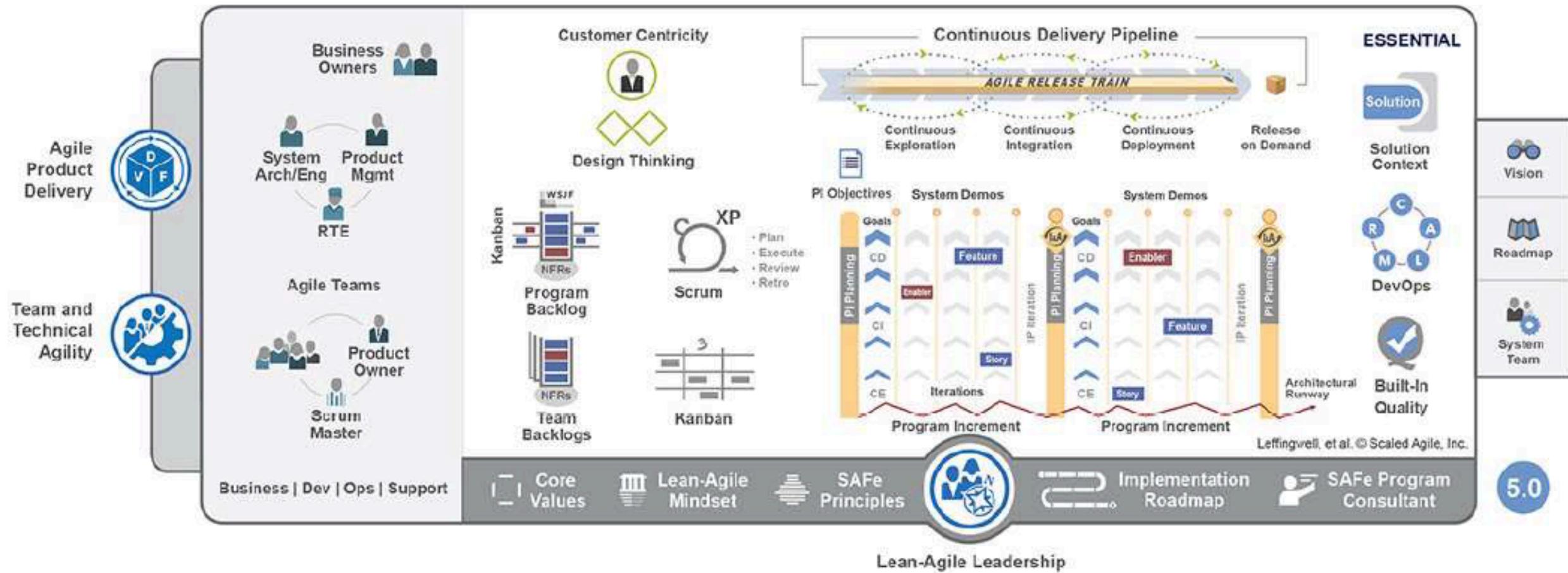
sont connectés

Automatisation

CI / CD / AI

aux pipelines

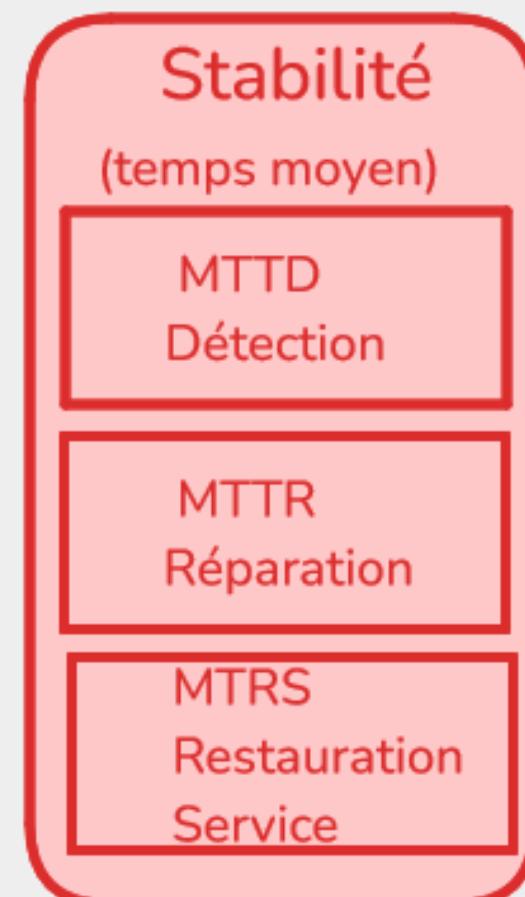
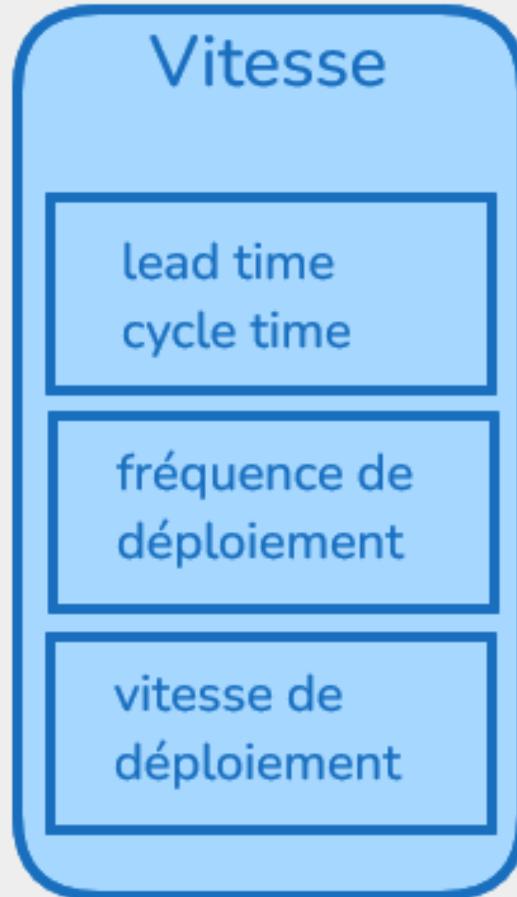
SAFe pour les entreprises Lean



Le « LEAN Manufacturing »

- 50's : méthode de management industriel chez Toyota « *Toyotisme* »
 - Exploitation des connaissances/créativité des acteurs de terrain « *bottom / up ou gemba* »
 - Flux de production **Juste à Temps** => *Kanban*
 - Elimination du **gaspillage** : « muri » *excès*, « muda » *inutile*, « mura » *non conforme*
 - **amélioration continue**: « *kaizen* »
- “ <https://www.leanproduction.com/top-25-lean-tools.html> ”

contrôler le cycle DevOps





ITSM/ITIL v4

- reformulation de LEAN/Agile/DevOps



