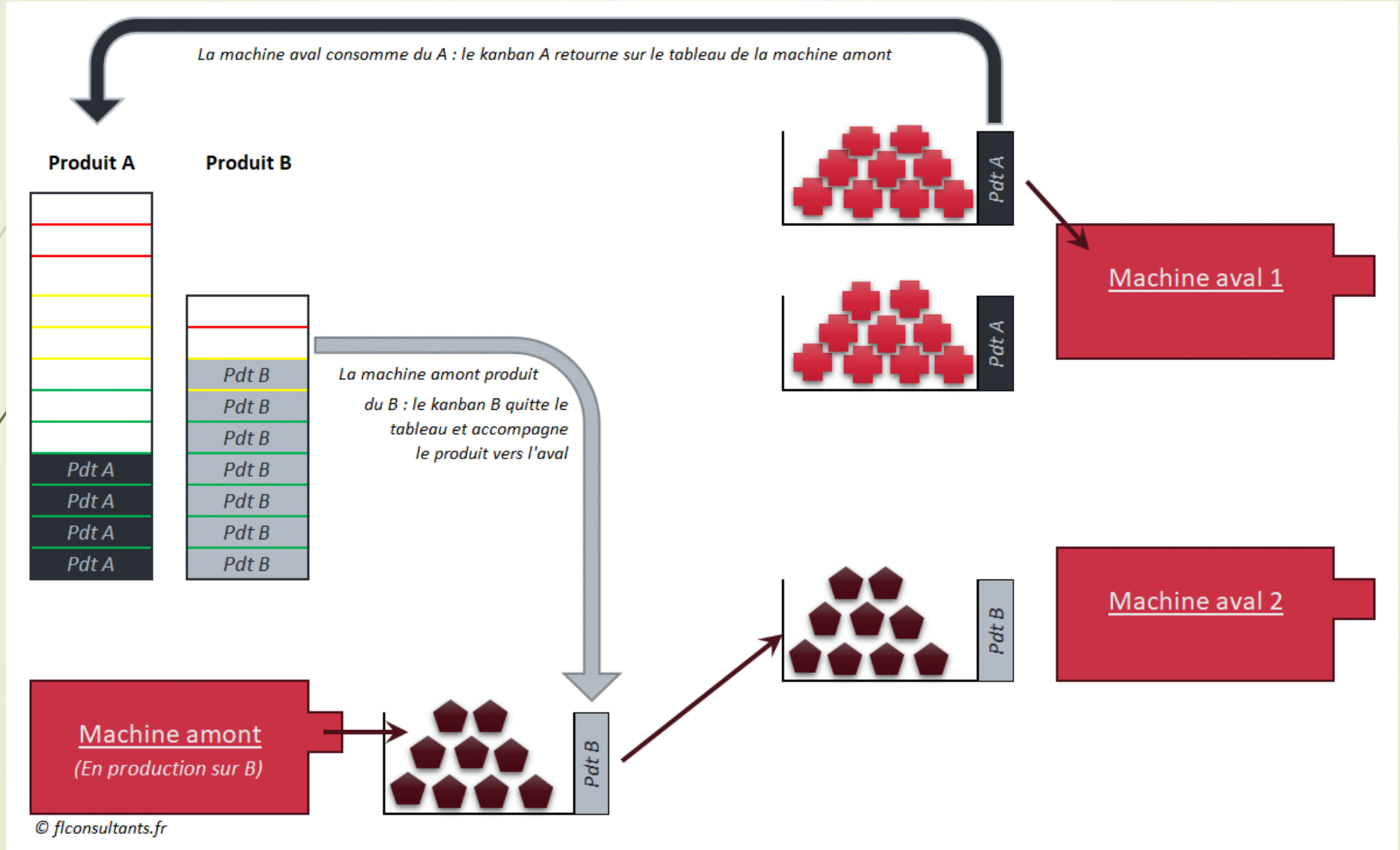
LES PRATIQUES DEVOPS

Kanban

- Définitions

- Issu du mot japonais signifiant « étiquette »
- À l'origine, méthode de réapprovisionnement des conteneurs de pièces sur les lignes d'assemblage des usines Toyota instituée dans les années 1950
- Le but de la méthode est d'obtenir des flux tirés « Pull System » de production sur un poste amont en conditionnant celle ci à la demande sur le poste aval
  - rythme de consommation sur le poste aval
  - état du stock sur le poste aval
- Le kanban véhicule l'information de production ou de consommation d'un lot de pièces

Kanban


La machine aval consomme du A : le kanban A retourne sur le tableau de la machine amont
Produit A
Produit B
Pdt A
Pdt A
Pdt A
Pdt A
Pdt B
Pdt B
Pdt B
Pdt B
Pdt B
Pdt B
Pdt B
La machine amont produit du B : le kanban B quitte le tableau et accompagne le produit vers l'aval
Pdt A
Pdt A
Machine aval 1
Pdt B
Machine aval 2
Machine amont
(En production sur B)
Pdt B
© flconsultants.fr

Kanban

- Application au développement logiciel

• Un tableau kanban dispose et déplace des étiquettes représentant des tâches dans des colonnes fonctions d'états d'accomplissement dont les principaux sont :

- à faire (« backlog » qui tient lieu de spécifications formelles)
- en cours
- accompli

• Chaque étiquette contient

- le nom de la tâche, de l'auteur assigné, du flux associé ou de la catégorie
- la date de création, la date de finition envisagée
- la priorité
- le pourcentage d'avancement

• Outils kanban en ligne comme Trello ou Jira

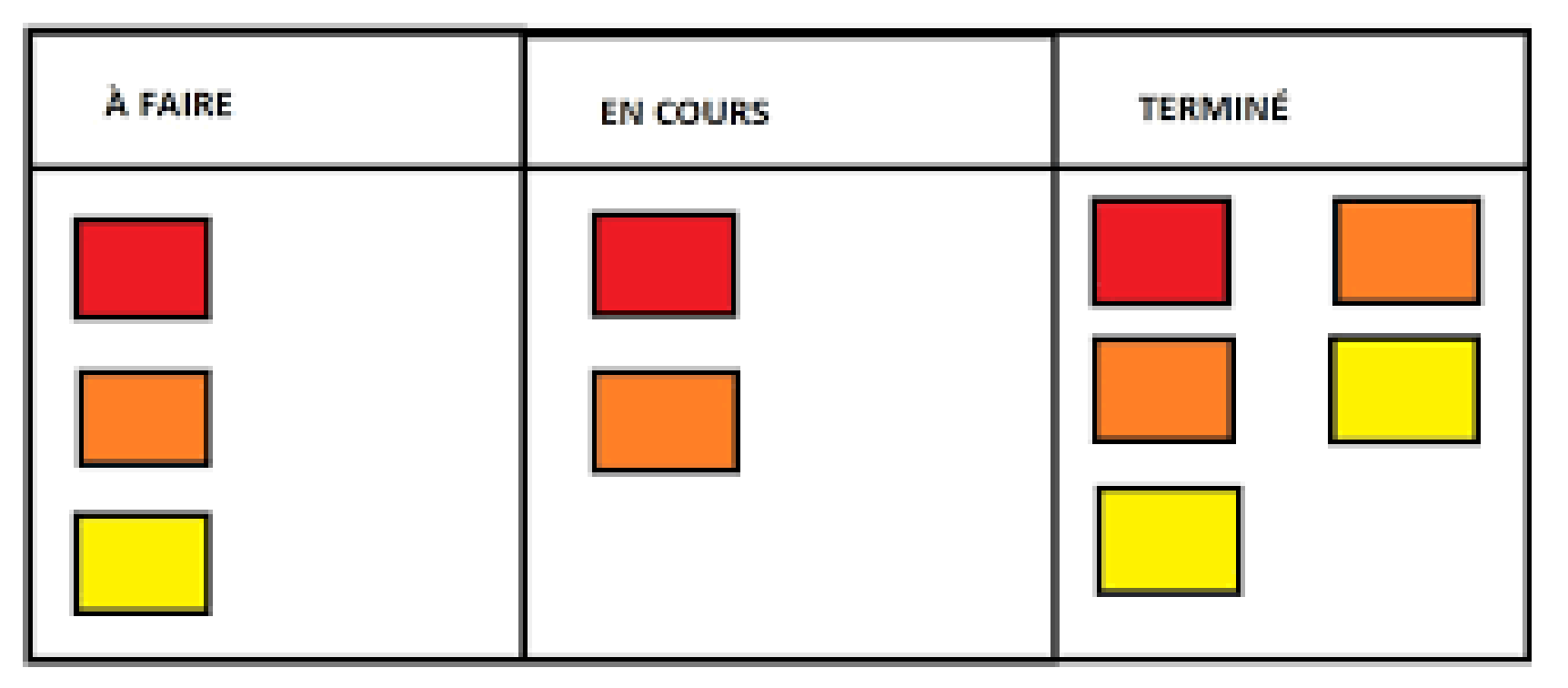
Kanban

- Description d’une étiquette : INVEST

- Independent : une étiquette doit être auto-informative
- Negociable : une étiquette n’est pas un contrat, elle autorise la discution
- Valuable : une étiquette doit ajouter de la valeur à l’existant
- Estimable : on doit pouvoir mesurer le coût d’une étiquette (temps / jour homme)
- Small : une étiquette doit être suffisamment petite pour être réalisable / estimable
- Testable : doit contenir l’information suffisante à l’écriture d’un test

Kanban

- Exemple simple



Kanban

- Utilisation avancée

• On peut ajouter à gauche du backlog une colonne « stories » contenant une expression informelle de besoin (spécification fonctionnelle)

• On peut ajouter à droite du backlog des colonnes de tâches à faire planifiées dans le temps :

- « cette semaine », « aujourd'hui »

- sprint SCRUM : lot de tâches composant un livrable, durant quelques semaines

• On peut décomposer la colonne « en cours » en plusieurs colonnes fonction de l'opération exécutée :

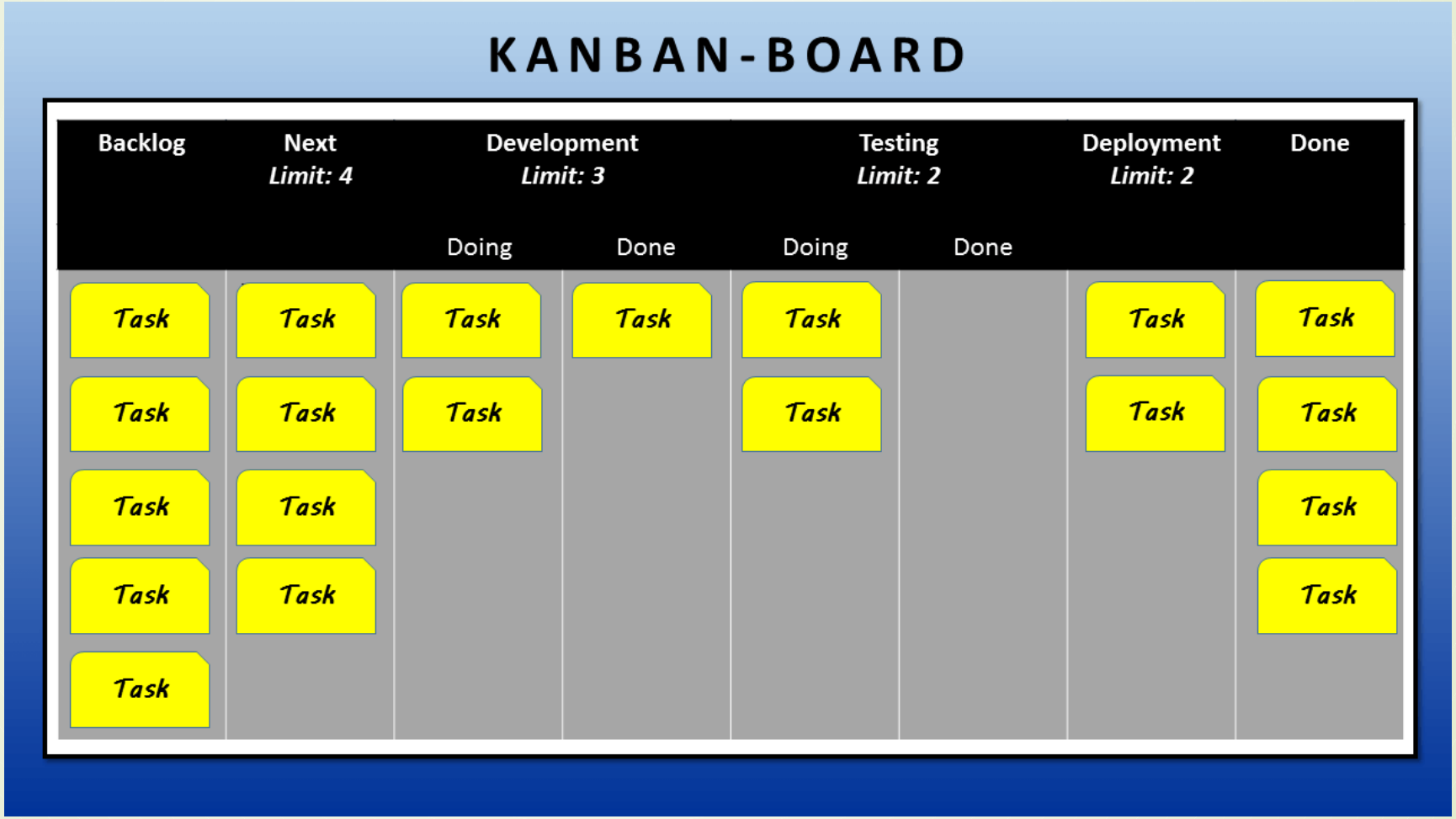
- développement

- test(s)

- analyse qualité

Kanban

- Utilisation avancée



Kanban

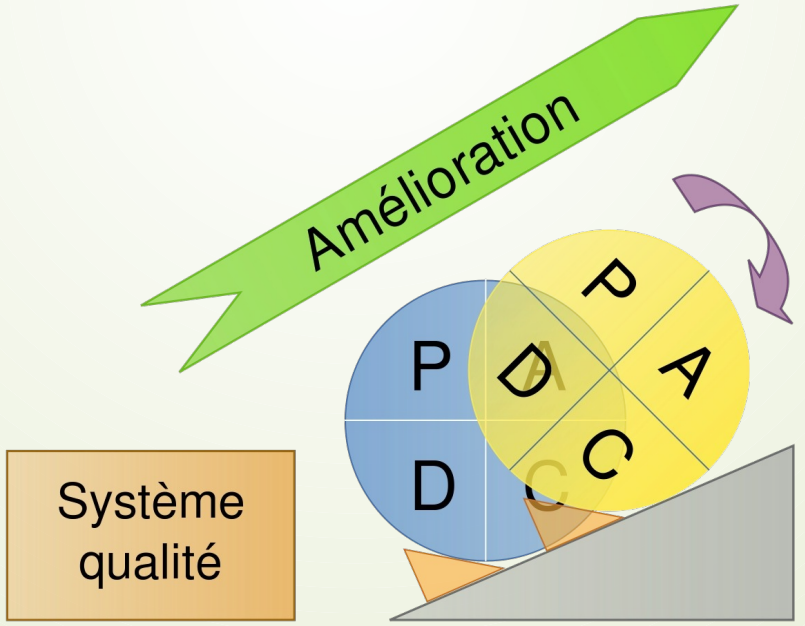
- extras

- On doit actualiser la place des étiquettes toutes les 2 heures à une fois par jour
- On peut vouloir fixer les nombres minimum et maximum de tâches dans une colonne, pour tenir compte de la force de travail disponible, et détecter les problèmes
- Il est conseillé de faire une utilisation interactive du kanban, notamment en organisant des petites réunions matinales quotidiennes (~15 min) pendant lesquelles chaque collaborateur résume son travail de la veille, et explique son travail de la journée.

La roue de Deming

- Diagramme PDCA

- La roue de Deming ou diagramme PDCA (Plan, Do, Check, Act ou Adjust) est la représentation générique d'un flux d'opération transformant une idée en acte- plusieurs itérations de la roue représentant autant d'amélioration.
- Les démarches d'intégration continue et DevOps sont fondés sur cette représentation



La roue de Deming

- Plan, Do, Check, Act, en expérimentation et intégration

- Plan : Hypothèse / Planification des changements et du résultat attendu
- Do : Expérimentation / implémentation
- Check : Évaluation du résultat par rapport aux résultat attendu
- Act (Adjust) OK : Recherche d'amélioration / déploiement de la solution
- Act (Adjust) KO : Analyse de l'echec
- Recommencer

« Improved Kata »

- Principe

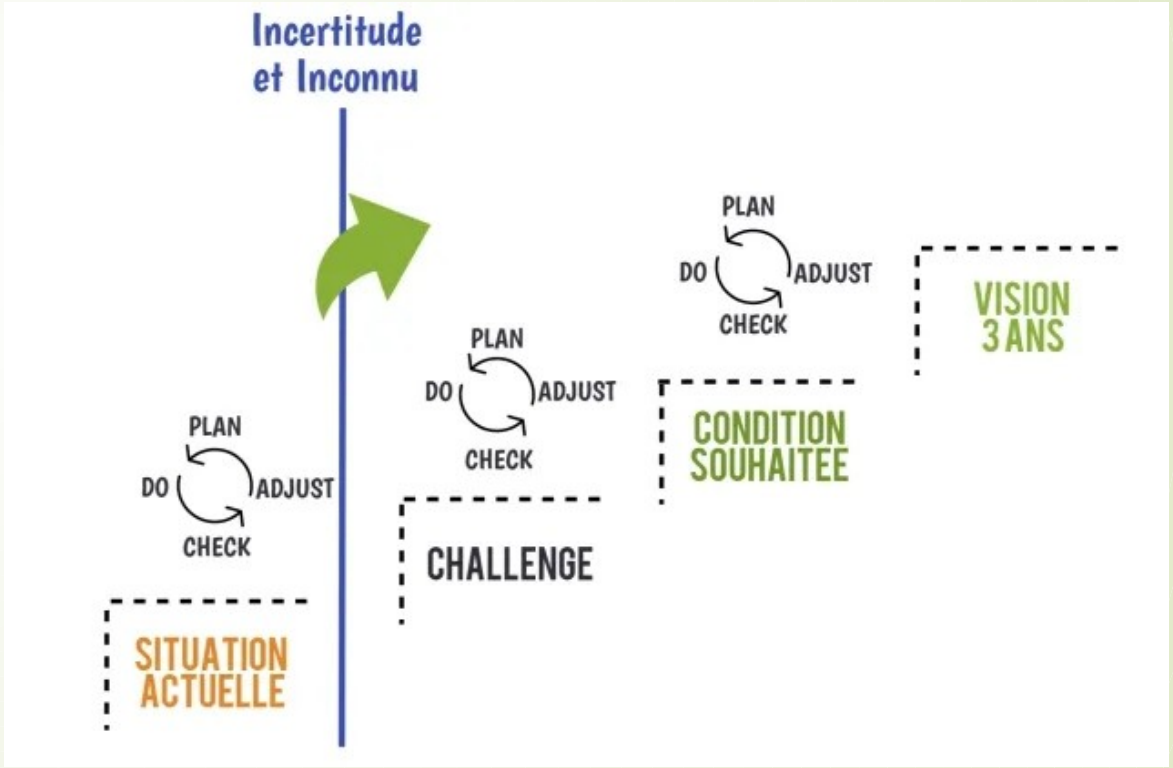
• Dans les arts martiaux, un Kata est un ensemble de mouvements coordonnés, codifiés et répétés jusqu'à ce qu'ils deviennent un réflexe ou une seconde nature.

• La méthode d' « improved Kata », ou kata d'amélioration continue, consiste à :

- considérer une fonctionnalité qui a le potentiel d'être améliorée
- l'analyser attentivement afin d'en comprendre les spécificités et les subtilités
- définir cette même fonctionnalité telle qu'on souhaite la voir évoluer à long terme
- appliquer des cycles de PDCA, de proche en proche

« Improved Kata »

- Schéma



« Improved Kata »

- Les 5 questions à poser

- Quelle est la situation cible à atteindre ?
- Quelle est la situation actuelle (résultat des précédentes expérimentations) ?
- Quels obstacles bloquent la validation ?
- Quelle est la prochaine étape (intermédiaire) ?
- Quand peut-on vérifier le résultat de cette expérimentation et en ressortir un apprentissage ?

DevSecOps

- Intégration de la sécurité

- La complexification progressive des infrastructures, la modularité des applications, la dynamique d'allocations de ressources sur les architectures microservices imposent des nouvelles pratiques de la sécurité

- Les pratiques classiques d'analyse statique de la sécurité (audits, checklists de contrôles) ne suffisent plus

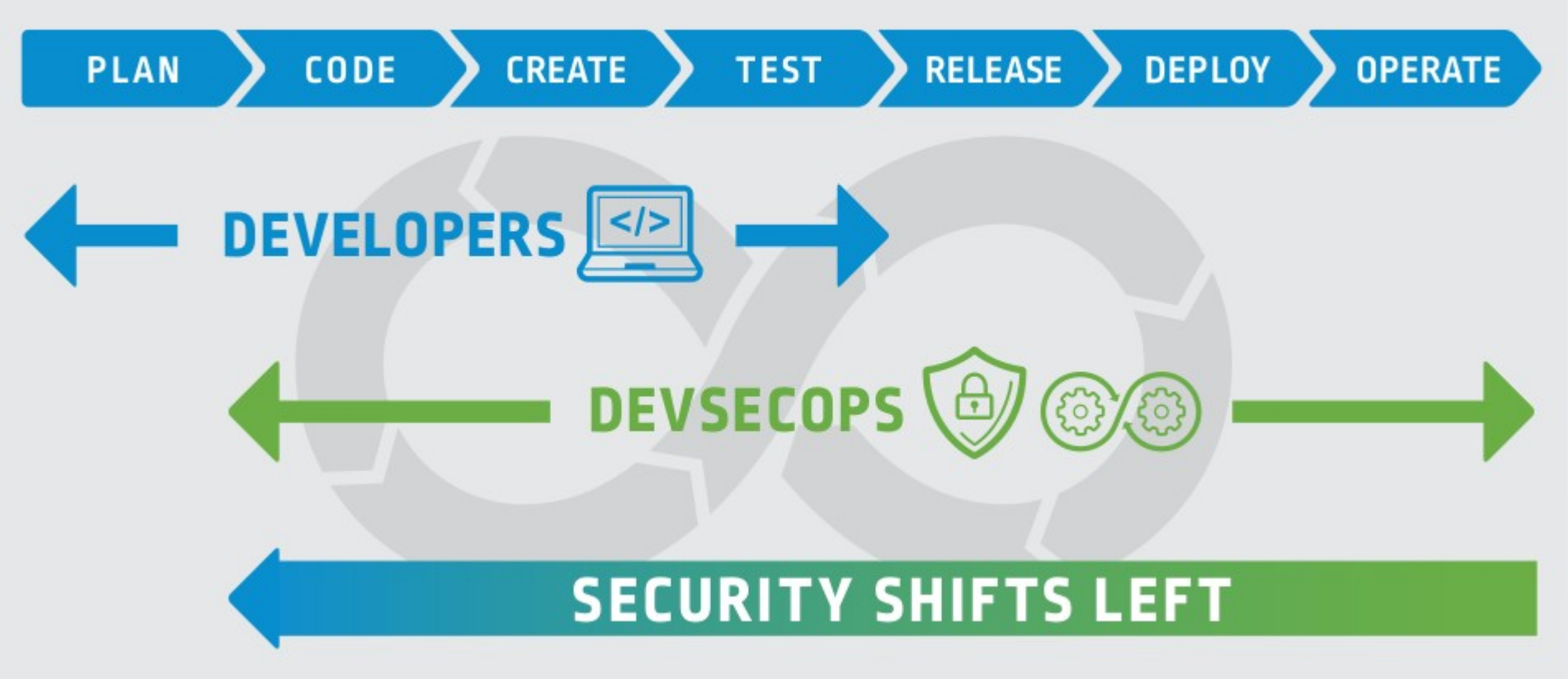
- Le modèle DevSecOps détaille l'intégration de la sécurité au sein du cycle développement, dès la planification et jusqu'à la maintenance serveur

- Cela implique l'automatisation des opérations de tests de la sécurité du code développé ainsi que des environnements d'exécutions du code

- DevSecOps est particulièrement dédié à la sécurisation des environnements conteneurisés

DevSecOps

- « Shift Left » de la sécurité



DevSecOps

- Pratiques de sécurité

- Déterminer dans le code la stratégie de privilèges minimum dans les connections et accès ressources
- Centraliser la gestion des identités et des secrets et mettre en place des modes d'authentification multi facteurs (mot de passe + SMS ou question de sécurité ou captcha...)
- Analyse de la composition des logiciels : Méthode recherche de composants (bibliothèques, foncitons) sources de vulnérabilités connues
- Déterminer les services minimaux à activer sur l'OS hôte, et fixer leur profil de sécurité via des modules de sécurité de l'OS (SELinux, AppArmor…)
- Déterminer les ports de communication minimum à autoriser (par-feu)
- Chiffrer les données échangées entre les services et les applications (APIs)

DevSecOps

- Les Tests de sécurité

- Automatisation des tests de validations d'entrées (authentification, formulaires), en contrôlant les valeurs et les types données, et détectant les injections SQL, CSRF Cross Site …

- Automatisation des mises à jour de sécurité OS et des bases de données antivirus

- Automatiser les tests d'intrusion, ou pentesting, autrement dit des scénaris d'attaques de types « black box », donc sans informations préalables sur le système, qui comprennent :
  - des analyses de trames TCP /IP
  - le balayage des ports ouverts
  - les recherches d'accès publics pour installer une porte dérobée
  - les attaques « brute force » pour voler des identifiants trop faibles
  - les accès trop généreux aux ressources (777)
  - les injections
  - les failles systèmes

DevSecOps

- Sécurisation des conteneurs

  - Analyse de la fiabilité de la source des images initiales du conteneur (INSTRUCTION FROM)
  - Analyse du Dockerfile pour déceler des failles potentielles, et les retravailler à des fins de customisation
  - Utilisation d'un registre privé avec contrôle d'accès pour protéger les images.
  - Maximiser l'isolation des conteneurs par l'utilisation d'espaces de nom réseau (cluster virtuels)

Conformité serveur

- Audits de conformité

- Choisir un référentiel de sécurité dédié à l'activité
  - CIS (Center for Internet Security)
  - SOC2 (Service Organization Control)
  - PCI DSS (Payment Card Industry Data Security Standard)
  - Swift (Society for Worldwide Interbank Financial Telecommunication)
  - ANSSI (Agence Nationale de la Sécurité des Sytèmes d'Information)

- Chaque réferenciel consiste en une série de règles de sécurité

- Ces règles sont formalisables en un programme exécutable par un agent d'automatisation (Ex. RUDDER)

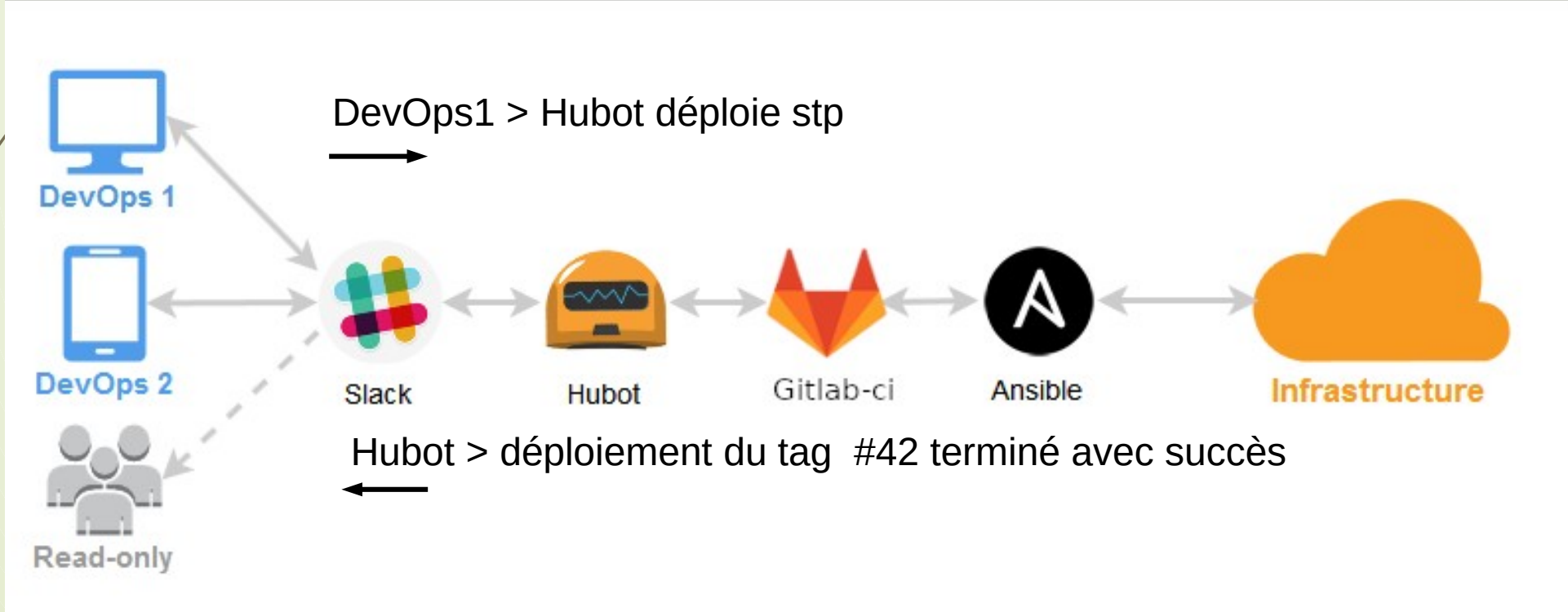
ChatOps

- Description
  - Un ChatOps est un outil de communication et de collaboration ayant pour but inital d'offrir des canaux de communication pour les équipes (chat , audio, visio, partage documentaire…) comme Slack ou Microsoft Teams
  - En connectant cet outil à la toolchain DevOps ou à l'infrastructure au moyen d'un outil de type chatbot (Hubot, Lita...), il permet également :
    - de conserver et centraliser l'historique de l'activité DevOps (conversations, rapports issu du CI/CD)
    - de déclencher certaines opérations pour réduire le nombre d'interfaces utilisateur DevOps (création de tickets / backlog, déploiement, orchestration…)
  - Le passage par un chatbot permet d'accéder à la toolchain par des commandes plus proches du langage naturel

ChatOps

- Exemple : déploiement

- Le ChatBot est programmé pour traduire des consignes écrites dans le chat en commandes envoyées au gestionnaire de CI/CD, qui à son tour déclenche l'exécution d'un outil
- Le ChatBot délivre sur le chat le résultat de l'action entreprise.



SRE

- Définition

- L'ingénierie de la fiabilité des sites (SRE, Site Reliability Engineering) est une approche d'ingénierie logicielle pour l'exploitation informatique.
- Elle consiste à utiliser des logiciels pour gérer des systèmes, résoudre des problèmes et automatiser des tâches liées à l'exploitation.
- Cette pratique est utile pour créer des systèmes logiciels évolutifs et extrêmement fiables, pour gérer des systèmes volumineux, composés de milliers, voire de centaines de milliers de machines.

SRE

- Rôle

- L'ingénieur en fiabilité de site remplit un rôle unique et son profil est celui d'un Dev ou d'un Ops avec des connaissances en développement ET en administration.

- Il est responsable :
  - du déploiement
  - de la gestion de configuration
  - du monitoring
  - des services en production (disponibilité, latence, intervention d'urgence et gestion de la capacité).

- Il ne doit pas consacrer plus de 50 % de son temps à l'exploitation.
  Le reste du temps doit être alloué aux tâches de développement :
  - mise à l'échelle du système
  - standardisation
  - automatisation.

SRE

- Tâches

- Optimiser les déploiements.
- Gestion de l’infrastructure :
  - gestion de configuration (Chef, Puppet, Ansible)
  - orchestration (Terraform, Kubernetes)
- Monitoring et gestion des alertes / prévention des incidents basé sur des symptômes (métriques)
- Autoscaling basé sur les métriques et les capacités.
- Opérations manuelles sur l’infrastructure :
  - Correction de problèmes, incidents, accidents