

Comment gérer l'obsolescence de la chaîne CI/CD au sein d'une entreprise ?

Victorien GBAGUIDI

Organisme d'accueil : **Société Générale CIB**
Tuteur Pédagogique : **Khalid BELHAJJAME**
Maître d'apprentissage : **Olivier CHANTEFOIN**

Master 2 MAGE SITN
Année Universitaire 2025–2026

Introduction

Au sein de l'industrie informatique moderne, les chaînes d'intégration et de déploiement continues (CI/CD) jouent un rôle crucial dans la livraison rapide et fiable des logiciels. Cependant, avec l'évolution rapide des technologies et des pratiques de développement, ces chaînes CI/CD peuvent rapidement devenir obsolètes, posant des défis significatifs pour les entreprises d'un point de vue de l'efficacité opérationnelle et de la sécurité des systèmes d'information. Nous allons voir dans ce mémoire comment gérer l'obsolescence des chaînes CI/CD au sein d'une entreprise en explorant les stratégies adoptées pour maintenir leur pertinence et leur efficacité dans un environnement en constante évolution. Une solution dans le cadre de la Société Générale CIB sera proposée à travers la migration de la chaîne CI/CD vers des solutions plus modernes en passant de Jenkins vers Github Actions et en passant de serveurs on-premises vers un cluster de serveurs Kubernetes pour un microservice. Dans un premier temps, nous présenterons le contexte et les enjeux liés à l'obsolescence des chaînes CI/CD, avant d'explorer les différentes approches pour y faire face. Nous allons notamment présenter les stratégies de remédiation de l'obsolescence ainsi que des outils et technologies émergentes qui peuvent être intégrés dans les chaînes CI/CD pour améliorer leur résilience et leur adaptabilité. Enfin, nous conclurons en soulignant l'importance de la gestion proactive de l'obsolescence des chaînes CI/CD pour assurer la compétitivité et la sécurité des entreprises dans le paysage technologique actuel.

Contexte

Le service Alltra est en charge d'une application éponyme de gestion documentaire de transactions commerciales de matières premières. L'application est composée d'une quarantaine de microservices mais repose sur une infrastructure ci/cd vieillissante (Jenkins/XLDeploy). Cela pose des problèmes d'automatisation car aujourd'hui, les microservices sont en partie déployés manuellement allongeant considérablement la mise en production. De plus, du fait d'une configuration vieillissante (Jenkinsfile / Groovy) la correction des erreurs de build est plus compliquée. Certains plugins Jenkins présentent en outre des failles de sécurité (Common Vulnerabilities and Exposures ou CVE) et ne sont plus maintenus par la communauté.

Passer à l'usage de Github Actions permettra de rationaliser la chaîne ci/cd. Aujourd'hui le doublon Jenkins/Github Actions crée un surcoût injustifié quant aux coûts des licences Github et au maintien des serveurs Jenkins tandis que Github et Github Actions forment une solution intégrée. De plus, d'un point de vue technique les compétences de l'équipe Alltra sont plus alignées avec l'usage de Github Actions, plus aisée (configuration YAML), qu'avec Jenkins (configuration Groovy). La configuration YAML, déclarative, est plus aisée à lire et donc à maintenir que la configuration Groovy. Enfin la connexion au serveur Jenkins depuis le serveur Github crée une complexité injustifiée (à préciser) dans la

chaîne ci/cd. Le passage à Github Actions supprime ce besoin de connexion et regroupe l'hébergement du code et son déploiement. Les plugins sur Github Actions sont appelés des actions. La Société Générale dispose de d'actions internes sur Github Actions qui offrent des fonctionnalités d'automatisation ci/cd. Une gouvernance quant à la gestion du cycle de vie de ces actions sera nécessaire et discuté au cours de ce mémoire. La Société Générale peut également bénéficier d'un écosystème d'actions officielles venant de comptes certifiés (Verified Creators) contrairement aux plugins communautaires de Jenkins qui ne sont pas certifiés. Ma stratégie de migration consiste à exploiter les actions certifiées de Github Actions pour automatiser la chaîne ci/cd et à utiliser les plugins internes de la Société Générale pour ses besoins spécifiques.

Aussi, les serveurs on-premises actuellement supporté seront décommissionnés cette année et doivent donc impérativement être remplacés. Je propose de passer à un cluster Kubernetes. Ce choix stratégique répond à trois enjeux :

1. **L'Immutabilité** : Contrairement aux serveurs on-premises historiques sujets à des conflits de configuration au fil des déploiements, l'usage de conteneurs éphémères assure une isolation de l'environnement d'exécution en isolant chaque microservices et limite ainsi les effets de bord. Les conteneurs permettent également de conserver la même configuration sans impacter l'infrastructure matérielle sous-jacente. Cela limite les besoins de maintenance au niveau de la configuration et simplifie la maintenance de l'infrastructure ci/cd. Les conteneurs garantissent également une configuration dite iso-prod assurant que les dépendances du code source en environnement de développement soit identique à celles en environnement de production.
2. **L'Élasticité** : La capacité d'*autoscaling* permet d'adapter dynamiquement les ressources à la charge réelle, assurant la performance sans intervention humaine.
3. **L'Optimisation des ressources** : En lançant plusieurs microservices sur un même serveur à travers des conteneurs sur un cluster Kubernetes au lieu de les déployer isolément sur des serveurs distincts, on optimise l'usage des ressources et on réduit les coûts de maintenance des serveurs existants ainsi que la complexité de l'infrastructure informatique.

Enfin, nous démontrerons que la réponse à l'obsolescence n'est pas uniquement technologique, mais culturelle. La migration vers des standards modernes impose une **conduite du changement** rigoureuse pour accompagner la montée en compétence des équipes vers une culture *DevOps* autonome.

De plus, afin d'éviter que la nouvelle solution ne devienne la dette technique de demain, nous aborderons les stratégies de **gestion des connaissances** (*Knowledge Management*). À travers l'adoption de pratiques *InnerSource* et la documentation *as-code*, l'objectif est de pérenniser le savoir-faire et de garantir la maintenabilité à long terme de la chaîne de livraison.