

# Document de définition d'architecture



## Sommaire :

- Introduction
  - État des lieux
- Modifications de l'architecture
  - Framework d'architecture
  - Composants et interactions
  - Mise en place des changements
- Nouvelle solution d'architecture
  - Bonnes pratiques implémentées
  - Maintenabilité
  - Sécurité
  - Migration vers l'architecture cible
  - Analyse des risques
  - Métriques
  - Dette technique
  - Livraison
- Annexe

# Introduction

## État des lieux

L'entreprise LAE (Les Assureurs Engagés) est une entreprise d'assurance spécialisée dans les assurances-vie. Son engagement repose sur la satisfaction des clients, via son service client et sur la sécurité des données, notamment bancaires et à caractère privé.

Dans le cadre de l'évaluation d'un contrat, l'entreprise recueille des données sensibles tel que les maladies dont souffre le client, ses antécédents judiciaires, et son addiction à la cigarette et/ou à l'alcool.

L'entreprise est divisée en quatre services distincts :

- Le service client, qui est en communications directes avec les clients pour rediriger toute demande de leurs parts vers le bon service ou pour modifier des données sur le client dans les bases de données de client/facturation/vente. Il peut également demander au service légal d'apporter une modification à la base de données légale via un appel téléphonique ou un mail.
- Le service de vente, qui a pour fonction de créer des plans d'action dont la réalisation consiste à sélectionner des clients et de leur proposer des produits adaptés qui, s'ils sont intéressés, déclenche la rédaction d'un contrat par le service légal.
- Le service légal, quant à lui, va enquêter sur les clients afin de déterminer les termes du contrat, ainsi que ses risques puis rédiger ce dernier, et cela suite à une demande du service de vente.
- Le service de facturation. Il intervient à la suite de la signature d'un contrat, afin de rédiger une facture et de l'envoyer au client.

Chacun des services présentés ci-dessus utilise une application unique, dont les technologies à leur origine sont toutes différentes. Cela complexifie la maintenance du système dans sa globalité. De plus, les processus ne sont pas fluides, par exemple si un opérateur du service client doit apporter un changement à la base de données légale, alors il doit transmettre sa demande par téléphone ou mail au service légal étant donné qu'il est le seul à y avoir accès.

La base de données du service légal est sauvegardée toutes les semaines, mais cette sauvegarde est réalisée à la main par le responsable informatique de l'entreprise uniquement. Cela implique que lorsque ce dernier est indisponible, aucune sauvegarde n'est réalisée et ces dernières ne sont plus accessibles.

Le Microsoft Access quant à lui recueille l'historique des appels et des mails des clients consignés par les opérateurs du service client. Cependant, il est accessible sans réelle vérification et consiste en un répertoire partagé synchronisé entre tous les opérateurs.

# Modification de l'architecture existante

## Framework d'architecture

La première différence notable avec l'architecture d'origine (figure 1 de l'annexe) est dans le fait qu'il n'y a plus qu'une seule application pour les différents services. Lors de l'utilisation de l'application, l'utilisateur sera invité à se connecter, ce qui permettra de l'identifier et de fournir les vues/accès auxquelles il a le droit.

La deuxième est que nous optons pour une architecture hybride micro-service/événementiel (figure 2 de l'annexe). Dans le but de réduire les délais de traitements ainsi que les erreurs de transmission de données entre les services, un bus d'événement sera placé entre l'interface utilisateur et les bases de données. L'interface utilisateur demandera à un service de produire un événement indiquant quelle donnée(s) doit être changer en précisant des étiquettes correspondantes aux bases de données touchées par cet événement. Une fois l'événement produit et envoyé dans le bus, des services consommateurs reliés à chaque base de données vérifierons de l'étiquette leur correspondant, auquel cas ils la consommeront puis appliqueront le changements à leur base de données. Avec un tel système, nous nous assurons d'une transmission d'information de qualité ainsi que de bases de données synchronisé.

Les changement concernant la base de données légal aura cependant besoin de la confirmation du service légal afin d'être validé.

La troisième modifications apporté concerne la sauvegarde de la base de données légal effectué à la main jusqu'à lors par le responsable informatique. Un service sera créer pour effectuer un dump de la base de données de manières périodique afin de fiabiliser ce processus. Par ailleurs ce service pourrait être étendue aux autres bases de données si le souhait en est émis.

## Composants et interactions

Le point d'entrée de la nouvelle architecture sera une interface homme machine, hébergé sur un serveur, qui demandera à l'utilisateur de se connecter à son compte via une session qui lui sera distribué. Une fois la connexion effectuer l'interface chargera les vues correspondant aux accès/besoins de l'utilisateur. Si ce derniers a besoin d'effectuer une modifications à une base de données, alors les informations nécessaires seront émis à un service producteur qui va générer un événement sur mesure.

Cette création va devoir respecter plusieurs règles afin de convenir à l'architecture. Tout d'abord l'événement devra contenir des étiquettes détaillants quelle(s) base(s) de données est(sont) concernée(s) par ce derniers. Cela permettra de garder actif un événement tant que tous les consommateurs concerné n'aient été atteint, de plus l'identité de l'utilisateur émetteur de l'événement sera consigné dans ce dernier pour une meilleure traçabilité et pour s'assurer que, si la modifications porte sur des données légal, la modification soit d'abord accepté par un membre de ce service.

Un fois qu'un événement est créé, il est envoyé dans le bus où il demeurera tant que toutes ses étiquettes ne soit consommées. Les consommateurs désignés par ces étiquettes vont alors recevoir l'événement et le traduire en modification qu'ils opéreront sur leurs base de données respectives.

En parallèles des intervention manuels, deux comportement automatisé surviendront :

- Le premier est un service de sauvegarde cyclique qui, toute les semaines, effectuera une sauvegarde d'un dump de la base de données légal.
- Le second est le service de centralisation et d'archivage des logs retournés par les producteurs et consommateurs d'événements.

## Mise en place des changements

Dans le but de mettre cette nouvelle architecture en place tout en réduisant au maximum l'impact sur le travail des employés pendant la transition, nous allons commencé par déployé une CI/CD (Intégration continue/Livraison continue) qui nous permettra de validé les fonctionnalité via des procédures de tests avant leurs déploiement dans le système. Ensuite la nouvelle architecture reposant grandement sur le bus d'événement, il devra être déployé en second, suivi des outils de monitoring dans le but de s'assurer du bon fonctionnement des nouvelles fonctionnalités.

Une fois la nécessaire réalisé, nous pourrons commencé à développé les fonctionnalité les unes après les autres. Leurs déploiement sera progressif afin de s'assurer de la validité de chaque fonctionnalité avant de passer à la suivante pour ne pas bloqué le fonctionnement de l'entreprise.

Pour le déploiement des nouvelles fonctionnalités, nous allons nous appuyés sur deux points afin de s'assurer que leur introduction soit réalisé avec le moins d'accro possible :

- Le premier point est que l'IHM (Interface Homme Machine) sera disponible en parallèle de l'ancienne application pour chaque service. Elle ne contiendra pas toutes les fonctionnalités dans un premier temps comme décrit précédemment et s'en verra agrémenter au fur et à mesure du développement. Cela permettra de familiarisé les employés progressivement à la nouvelle application.
- Le deuxième point est que nous allons installé pour la phase de transition un service RBAC (Contrôle d'Accès Basé sur le Rôle) en amont de l'IHM permettant de choisir l'attribution des fonctionnalités avant leur déploiement final à des rôles qui sont répartis parmi les employés, ce service pourra également permettre de choisir la version désiré de la fonctionnalité testée. Ce service permettra de réalisé des pré-déploiement sur une faible portion d'employés pour un test dont les retombées seront minimisé. De plus c'est également une manière d'impliquer les employés dans le développement en récupérant leurs retour d'expérience sur les nouvelles fonctionnalités. Le bonus est que si jamais un pré-déploiement se passe mal, un retour en arrière vers la dernière version stable sera simple à réaliser.

## Nouvelle solution de l'architecture

### Bonnes pratiques implémentées

L'utilisation d'une architecture hybride permet de prendre les avantages des architectures micro-services et événementiel de la manière la plus adaptée au projet. De cette manière nous pouvons facilement mettre en places plusieurs pratiques importantes tel que la synchronisation des

bases de données, afin d'éviter toute divergence qui pourrait être nuisible aux clients ou aux processus de l'entreprise.

Ce type d'architecture nous permet également d'améliorer grandement la traçabilité des modifications apporté aux différentes bases de données par le biais des événements. Avec le service de monitoring du système, il sera facile d'identifier un comportement inattendu/indésirable ou une erreur commise par un utilisateur ainsi que les données affectées et à quelle moment.

## Maintenabilité

La maintenabilité du système global va être grandement amélioré par plusieurs points de la nouvelle architecture.

Pour commencer le fait de fusionner les applications en une seul va permettre de diviser par quatre les efforts de maintenance. D'autant plus que nous n'utiliserons qu'une seule technologie là où il y en avait quatre à l'origine.

Ensuite grâce à la CI/CD qui sera mise en place nous allons pouvoir effectuer des procédures de tests poussés dans le but de fiabiliser les fonctionnalités avant leurs déploiement final pour réduire aux maximum l'émergence de bug lorsque celui ci sera effectuer. L'option de l'IHM permettant de choisir quelle version de chaque fonctionnalité est utilisé permettra de pouvoir faire machine arrière sur une version précédente stable en attendant que la nouvelle soit corrigé, minimisant ainsi l'impact d'un bug bloquant sur la continuité de service.

## Sécurité

En ce qui concerne la sécurité, le premier point est qu'avec la gestion des rôles, un utilisateur n'aura accès qu'à ce dont il a droit. Ensuite nous pouvons séparer la sécurité en quatre aspects :

- Disponibilité : L'utilisation de l'architecture hybride micro-service/événementiel permet de bénéficier d'un suivi de l'utilisation des différents services et cloisonné les bugs s'il en survient afin de ne pas impacter l'ensemble du système à chaque fois.
- Intégrité des données : Via le service de sauvegarde cyclique, nous avons à disposition des dumps de la BDD (base de données) légale. Si cette dernières se retrouve corrompu, nous serons en mesures de rétablir la dernière sauvegarde réalisé.
- Confidentialité des données : Nous allons chiffrés les communications inter-service, ainsi que les BDD. Bien entendu toutes les BDD n'ont pas le même niveau de criticité, notamment la BDD Légal, donc si son chiffrements est prioritaire, la méthode employés pour cette dernières pourrait être inadapté à une utilisation généralisé due à un problème d'efficacité et/ou de capacité. Par ailleurs nous allons nous assurer de pseudonymisé les données afin d'éviter que, si une fuite de données survient, elles ne puissent être utilisé de quelque façons que ce soit.
- Traçabilité : Les services consommateur/producteur vont émettre des logs vers le service de centralisation, ce qui permettra de suivre tout le cycle de vie des événement afin de détecter au mieux un comportement inattendu ou un événement non traité. De plus un outil de monitoring offrira la possibilité de garder un œil sur la solution en entière aux administrateurs.

Le deuxième point réside dans la connexion à une session via l'IHM. Cette mesure permet d'éviter une persistance des données en local sur les machines des collaborateurs tout en ajoutant la possibilité de configurer des déconnexions pour inactivité ou autre bonne pratique.

## Migration vers l'architecture cible

L'utilisation de la nouvelle solution sera disponible pour tous les services de l'entreprise au fur et à mesure de son développement. L'IHM sera disponible en parallèle des anciennes applications tant que les fonctionnalités de ces dernières ne seront pas totalement remplacées, comme précisé plus haut.

Une architecture de transition (figure 3 de l'annexe) sera nécessaire pour effectuer un déploiement fluide de la nouvelle solution. Elle nous permettra de déployer les fonctionnalités de la nouvelle solution au fur et à mesure, mais pour cela nous avons besoin de trois ajout temporaire absent de l'architecture d'origine.

Le premier est le RBAC (Contrôle d'Accès Basé sur le Rôle) qui sera le service nous permettant d'associer les différentes fonctionnalités à des rôles, puis d'attribuer ces rôles aux utilisateurs selon les besoins des tests de déploiement. Ainsi, nous pourrons mener des tests à petite échelle pour valider la qualité d'une fonctionnalité avant son déploiement final.

Le deuxième est le service de conversion. Il nous servira à convertir les informations reçu du consommateur au format de la base de données d'origine, si cette dernière est d'un type différent de la nouvelle.

Et enfin le troisième ajout est le service de synchronisation qui aura pour fonction de s'assurer que la nouvelle base de données reçoivent bien les changements qui pourraient être effectués depuis l'application d'origine. Le but étant que les tests des fonctionnalités par un échantillon d'employé ne soit pas une activité séparée de leurs tâches habituelles, au contraire, ils s'agit simplement d'une nouvelle façon de les effectuer.

Avec cette architecture de transition, nous nous offrons la possibilité de fiabiliser le développement de la solution par des tests en condition réelle sans perturber outre-mesure les employés des différents services.

Cette architecture de transition sera l'une des étapes du processus de CI/CD et surviendra une fois que la fonctionnalité à tester aura validé les tests qui lui sont propres (test unitaires, de performance, etc.) pour valider le bon fonctionnement de cette dernière en condition réelle.

## Analyse des risques

Voici la liste des principaux risques identifiés pour ce projet :

ID	Risque	Facteur de mitigation
R1	Fuite de données sensibles	<ul style="list-style-type: none"> <li>Chiffrement des communications inter-service</li> <li>Chiffrement des bases de données</li> </ul>
R2	Rejet de la nouvelle solution par les employés	<ul style="list-style-type: none"> <li>Implications des équipes pendant le développement (déploiement à échelle réduite)</li> <li>Campagne de communication</li> <li>Sessions de formation interactives</li> </ul>
R3	Insertion d'erreur dans les BDD via une fonctionnalité défailante	<ul style="list-style-type: none"> <li>Centralisation des logs (traçabilité de l'erreur)</li> <li>Procédure de tests en amont du déploiement</li> <li>RBAC (retirer la fonctionnalité défailante)</li> </ul>
R4	Perte de données	<ul style="list-style-type: none"> <li>Sauvegarde séparé de l'hébergement courant</li> </ul>
R5	Désynchronisation des BDD	<ul style="list-style-type: none"> <li>Modifications des BDD via les événements</li> </ul>

Et en voici la matrice Probabilité/Impact :

		Impact				
		Négligeable	Mineur	Modéré	Majeur	Critique
Probabilité	Très probable					
	Probable		R3			
	Peu probable				R4	R1
	Improbable			R5	R2	

## Métriques

Dans le but de mesurer l'efficacité de la nouvelle solution, nous allons mettre en place et suivre l'évolution de différentes métriques que voici :

- Temps d'exécution des processus opérationnel : L'un des principaux maux de l'entreprise est la durée de traitement du fait que les informations clients doivent être modifiées à plusieurs endroits, que des répertoires partagés doivent se synchroniser, que certaines informations sont synchronisées à la main ou encore que les demandes clients sont redirigées par téléphone ou email.
- Fréquence d'incident de désynchronisation entre des BDD : Avec l'architecture d'origine les informations clients peuvent ne pas être mises à jour partout, nous allons donc mesurer la fréquence d'incident issue d'information contraire dans les BDD.
- Fréquence d'incident de sécurité : La solution traitant des données sensibles des clients de l'entreprise, la sécurité est un point essentiel. Aussi bien que des tests seront effectués afin d'éviter le déploiement de fonctionnalités présentant des failles, il est prudent de vérifier via cette métrique qu'aucune faille ne soit passée entre les mailles du filet.

## Dettes techniques

L'architecture d'origine présente plusieurs points problématiques dont découle des choix pour la nouvelle architecture.

Premièrement, chaque service possède une application qui lui est propre développée dans des technologies uniques dont certaines sont obsolètes ce qui complexifie inutilement l'effort de maintenance requis.

Deuxièmement, le service client enregistre des données essentielles sur un dossier partagé étant stocké par tous les collaborateurs en local et synchronisé entre eux. Cela allonge la durée de traitement des requêtes client à cause de la durée de synchronisation.

Troisièmement, le service légal dispose d'une base de données hébergée sur un serveur local, un dispositif efficace pour protéger les données sensibles des attaques extérieures. Cette configuration présente toutefois deux inconvénients majeurs :

- La sauvegarde de la base de données est manuelle et assurée par le seul responsable informatique, ce qui peut engendrer des risques de pertes de données en cas d'absence. D'autant plus qu'il est le seul à avoir accès au stockage des sauvegardes.
- Une communication par email ou téléphone est nécessaire pour toute demande de modification en provenance du service client, ce qui ralentit le processus.

L'un des objectifs principaux du projet étant de réduire cette dette technique accumulée, voici comment la nouvelle solution va traiter ces différents points :

- Les différentes applications vont être fusionnées en une seule IHM adaptative afin de réduire la complexité technologique et l'effort de maintenance.
- Le service client va utiliser une base de données unique pour tous les services dont l'édition se fera via la création d'événement, cela permettra d'augmenter l'efficacité des opérateurs puisque les modifications pourront s'enchaîner sans que les collaborateurs n'aient à patienter pour cause de synchronisation.
- Le service légal verra sa base de données inscrite dans l'architecture hybride avec un service de sauvegarde cyclique, effectuant et sauvegardant des dumps de la BDD, et un consommateur spécial à l'écoute d'une modification en provenance du service client afin que le service légal puisse d'abord l'autoriser.



## Livraison

Pour ce projet nous allons utiliser la méthodologie Agile. User de cette pratique va offrir plusieurs point intéressant. L'entreprise souffrant actuellement d'une dette technique prononcé, il est préférable de s'orienter vers un développement avec des cycles courts centrés sur une fonctionnalité. Cela en éprouvera la fiabilité, via des séries de tests à chaque itération, ainsi que la conformité, notamment grâce aux nombreux retours que nous pourrons récolter à la fin d'une itération. En appliquant cette méthodologie, nous pourrons donc minimiser au maximum la génération de dette technique supplémentaire lors du déploiement final de chaque fonctionnalité.

Pour appliquer au mieux ces directives nous allons également mettre en place une CI/CD comme déjà énoncé précédemment. Ce processus servira à automatisé la procédure de tests (test unitaires, fonctionnels, etc.) qui assurera une qualité minimal des fonctionnalité avant de les validés grâce à des tests ciblé rendu possible par la mise en place du RBAC.

Vous pouvez retrouver un diagramme de gantt découper en quatre partie (la première pour détailler les tâches à réalisé et les trois autres pour la représentation de ces dernières ainsi que leurs dépendance/hierarchie) qui sont les figures 4, 5, 6, et 7 en annexe de ce document.

Le projet va être séparé en deux phases distinct :

- Préparation de l'environnement : L'objectif de cette phase est de mettre en place tous les outils nécessaire à la création de la solution.
- Développement de fonctionnalité : L'objectif de cette phase est de développer dans l'IHM les différentes fonctionnalités que contenait les applications d'origine de chaque services.

La première tâche qui devra être réalisé est la création de la CI/CD, puisqu'elle servira à tout le reste. En partant de là deux séries de tâches pourront être réalisé en parallèle afin de réduire le temps de développement (cela va néanmoins nécessité une ressource humaine supplémentaire, on y reviendra dans la prochaine section). La première série correspond à la mise en place du BUS d'événements suivi de la création du service « Producteur » et des « Consommateur ». La deuxième série, quant à elle, à pour but de créer les services de bases de la solution, ainsi que ceux nécessaires à l'architecture de transition, soit, dans l'ordre, l'IHM, le RBAC, les nouvelles BDD, le service de conversion (d'un événement vers l'ancienne BDD), et le service de synchronisation (de l'ancienne BDD vers la nouvelle). Ces deux série ont une durée estimé de respectivement 40 et 34 jours travaillés, d'où l'utilité de les développer en parallèle afin de réduire la durée de développement.

Une fois les deux série terminées, une troisième commence avec la deuxième phase. Il s'agit de l'apport des fonctionnalités des applications d'origine de chaque service à l'IHM en commençant par le service client, puis le service légal, le service de vente et enfin le service de facturation. Cela représente un travail de 20 jours par service pour un total de 80 pour la série.

En cumulant la première phase (42 jours puisque la création de la CI/CD est estimé à 2 jours et que la première série est de 40 jours) et la deuxième phase (80 jours), nous obtenons une estimation de la durée du projet de 122 jours. Étant donné que la durée limite du projet est de 7 mois, nous avons environs 213 jours de disponible. Donc en suivant les estimations le projet devrait respecter les délais.

## Chiffrage de la solution

LAE met à disposition pour ce projet, en dehors la collaboration entière de toute l'équipe, un ingénieur généraliste en informatique qui aura pour tâche de développer la plupart des tâches comprises dans ce projet. Ce faisant, comme expliquer dans le détail des tâches de la section Livraison, en parallèle de la création de l'IHM, du RBAC, etc, il y a trois tâches portant sur la création du bus d'événement. Ces tâches sont respectivement la création du bus, du producteur, et des consommateurs. Cette série de tâches représente un total de 40 jours, si l'on considère le taux journalier moyen d'un ingénieur DevOps comme étant à ~550€ (moyenne réalisé à partir des données du site « Malt »), cela représente un investissement total de 22 000€.

D'autre part pour une raison de mitigation des risques il est préférable de ne pas conserver les dumps de sauvegarde des BDD au même endroit que l'hébergement courant de la solution. Aussi, comme il est préférable d'héberger la solution avec des serveur en interne pour garantir un contrôle dessus, nous avons réalisé une estimation d'une location de stockage longue durée pour archivage via Google. Cette estimation à été faite pour des serveurs en Belgique avec un total de 100TB (taille pouvant être modifié selon le besoin) au mois pour 117,89€, donc si nous partons pour un premier devis avec 10 ans de stockage soit 120 mois, le coût total s'élève à 14 146,8€.

Avec ces deux points, la solution consomme 36 146,8€ du budget alloué de 200 000€, il resterait donc un total de 163 853,2€ de budget. Comme précisé un peu plus tôt nous recommandons d'héberger la solutions sur site afin de garder un meilleur contrôle sur cette dernières, ce faisant les serveurs de l'entreprise devront être adapté à cette fin, et le budget restant devrait permettre de couvrir ces frais.

Pour mettre en place le bus d'événement → devops tjm à ~500/600€ → 22000€ au total  
on héberge le site sur site avec le matos recyclé voir upscale  
stockage longue durée (archive) chez google (en belgique) = 100TB au mois pour 117,89€ → pour 10 ans = 14 146,8€  
36 146,8€ de consommé donc achat matériel pour site sur site large

## Annexe :

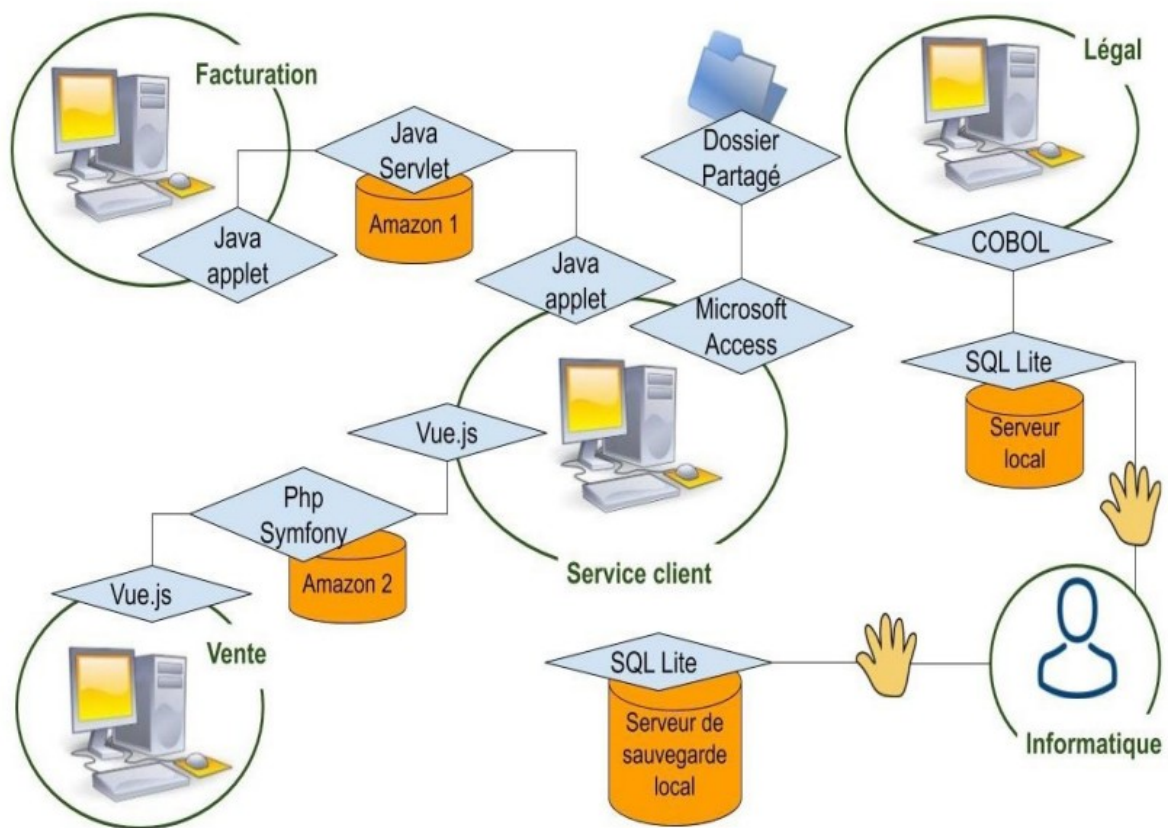


Figure 1: Architecture d'origine

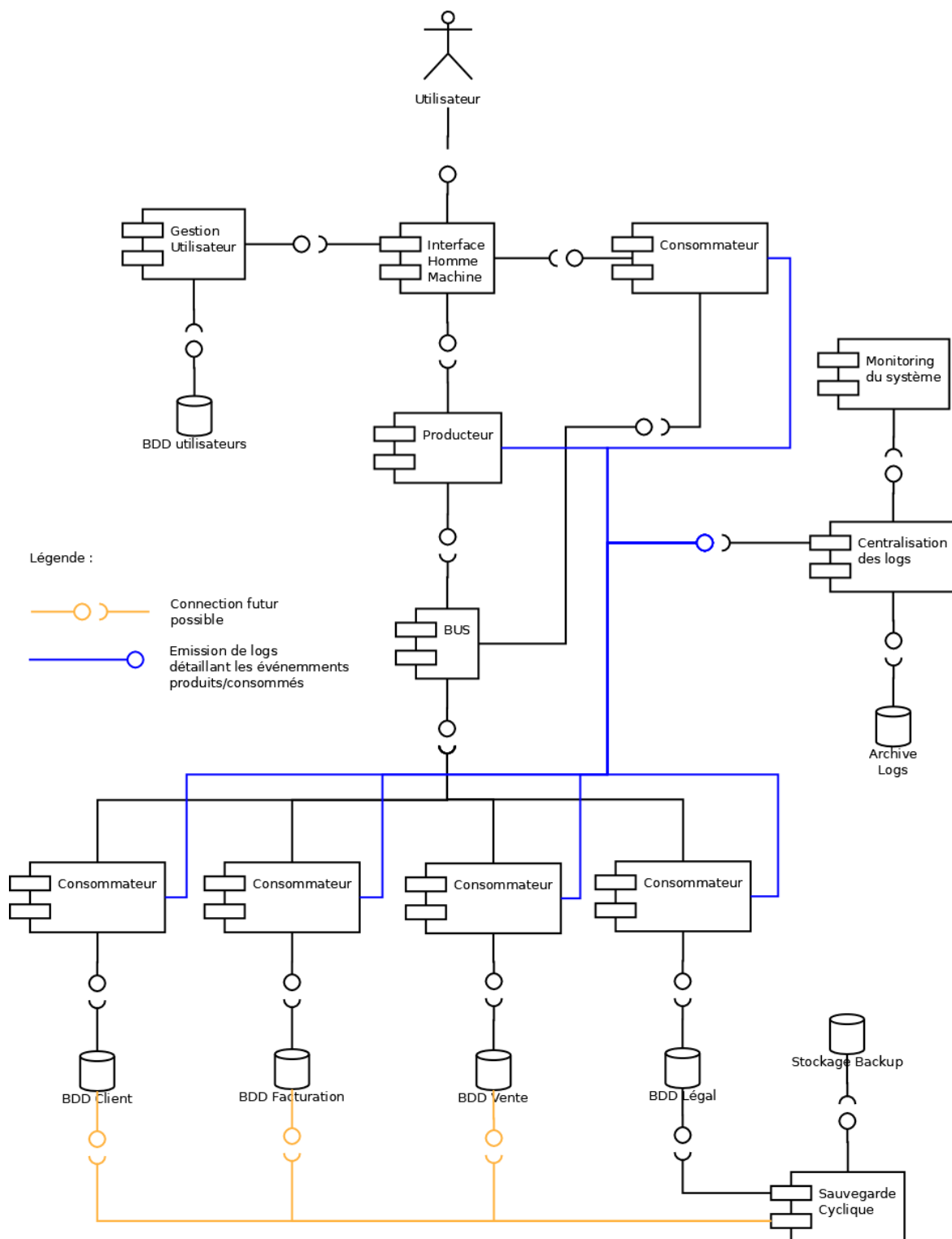
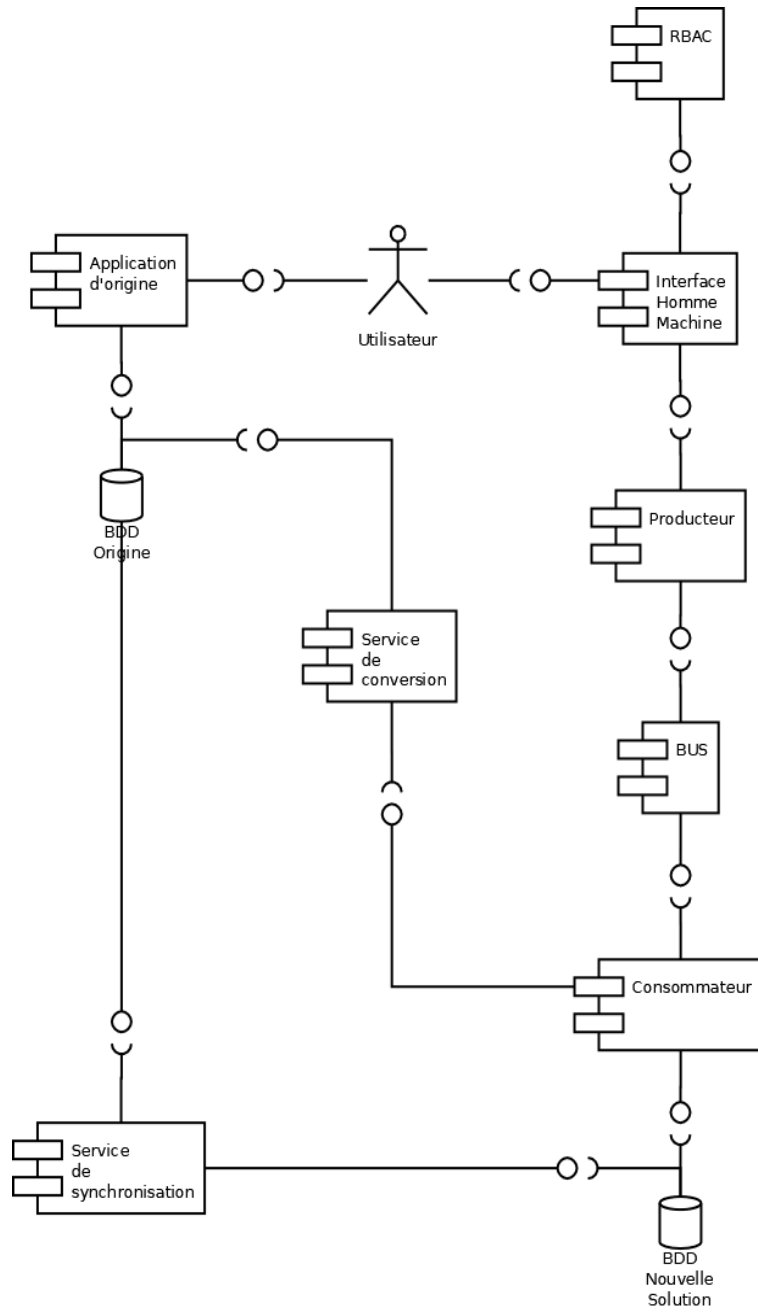


Figure 2: Architecture cible



*Figure 3: Architecture de transition*

▼ Préparation de l'environnement	28/01/2025	26/03/2025
Création de la CICD	28/01/2025	29/01/2025
Création du BUS	30/01/2025	12/02/2025
Création du Producteur	13/02/2025	26/02/2025
Création des consommateurs	27/02/2025	26/03/2025
Création IHM	30/01/2025	12/02/2025
Création RBAC	13/02/2025	17/02/2025
Création des BDD	18/02/2025	20/02/2025
Création Service de Conversion	21/02/2025	06/03/2025
Création du service de synchronisation	07/03/2025	18/03/2025
▼ Développement de fonctionnalité	27/03/2025	16/07/2025
Service Client	27/03/2025	23/04/2025
Service Légal	24/04/2025	21/05/2025
Service Vente	22/05/2025	18/06/2025
Service Facturation	19/06/2025	16/07/2025

*Figure 4: Détail des tâches du diagramme de Gantt, représenté dans les figure 5, 6, et 7*

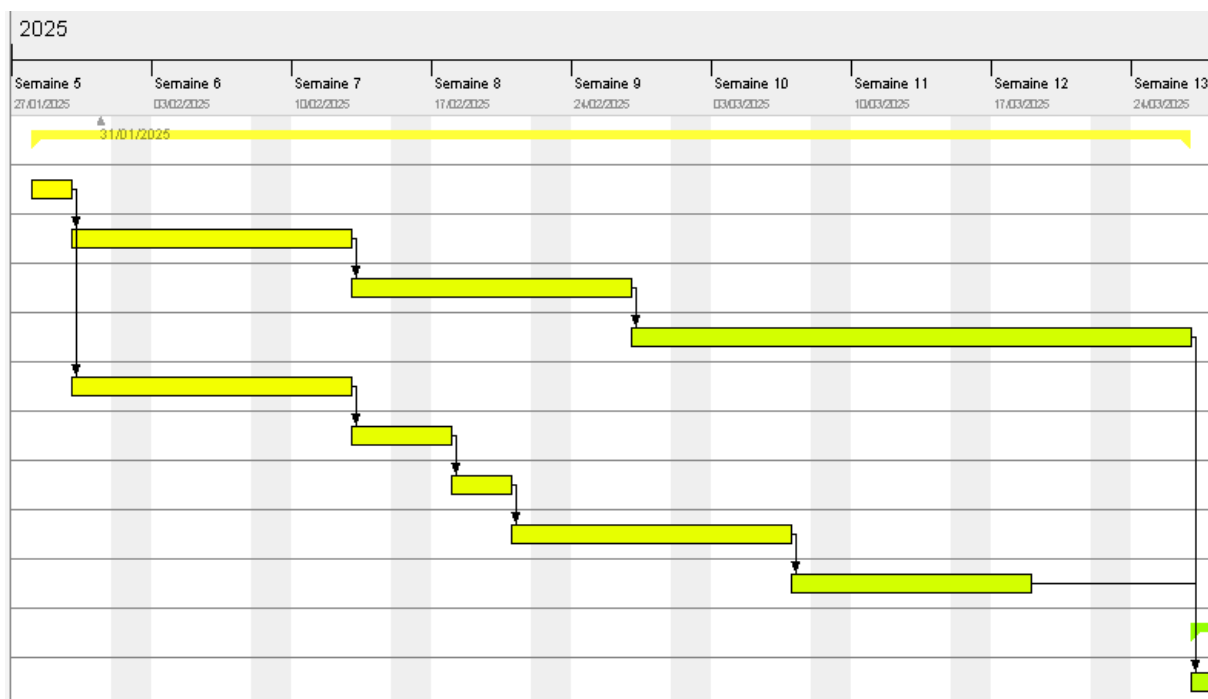


Figure 5: Première section du diagramme de Gantt représentant la phase de préparation de l'environnement

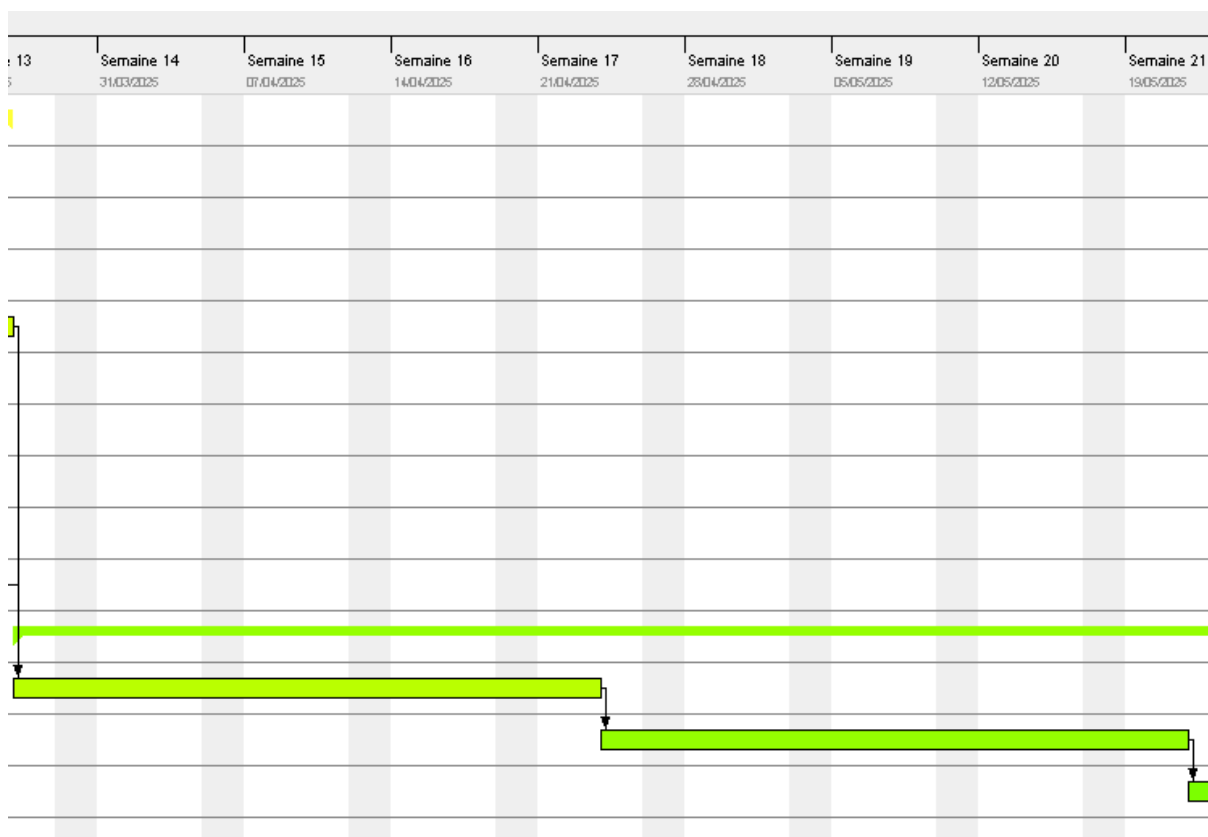
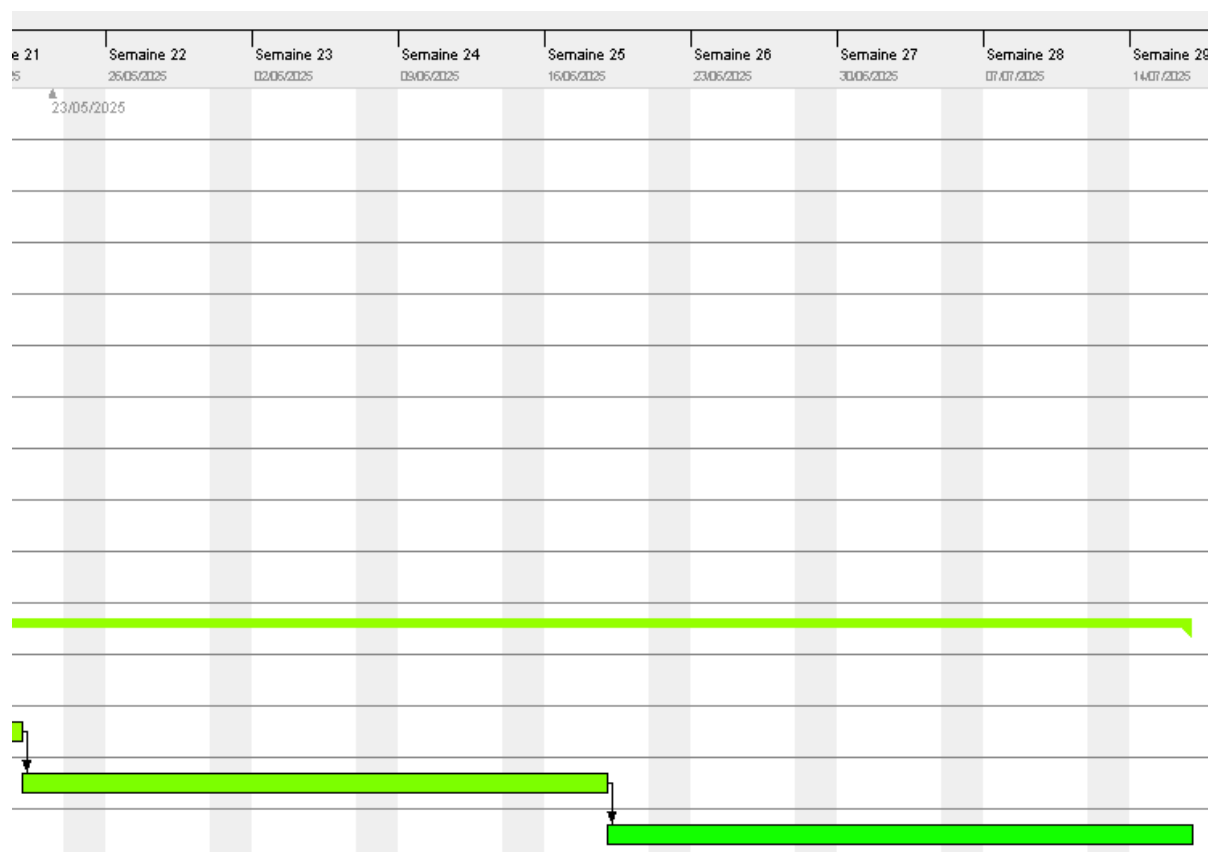


Figure 6: Deuxième section du diagramme de Gantt représentant le début de la phase de développement de fonctionnalité



*Figure 7: Troisième et dernière section du diagramme de Gantt représentant la fin de la phase de développement de fonctionnalité*