RAPPORT TECHNIQUE
PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE*ƒ*
DANS LE CADRE DU COURS LOG795 PROJET DE FIN D'ÉTUDES EN GÉNIE LOG-TI

Système serveur monolithique vers infonuagique

L'infonuagique pour VIF Télé - Une aventure avec Azure

Aubert Guillemette - GUIA04109509
Xavier Langlais-Savage - LANX03019404
Albert Villeneuve-Nguyen - VILA10108707

Département de génie logiciel et des TI

Professeur-superviseur
Alain April

MONTRÉAL, (15 avril 2020)
Session hiver 2020

Remerciements

L’aboutissement d’un projet logiciel ne dépend pas d’une équipe seule. Les membres de l’équipe aimeraient remercier les personnes suivantes pour leur implication et leur contribution dans la réalisation de ce projet.

Tout d’abord, nous remercions Benoît Johnson, président et fondateur de VIF Télé, pour sa confiance en nous et en l’École ainsi que pour son intérêt envers le projet.

Nous remercions Vincent Chouinard-Laliberté, directeur des opérations chez VIF Télé, pour son soutien, pour l’exactitude de ses informations, pour la précision des besoins de l’entreprise et pour son regard éclairé sur les enjeux liés à ce projet.

Nous remercions également Sébastien Meunier, superviseur du développement logiciel chez VIF Télé, pour sa facilité à nous transmettre l’information technique, pour son souci du détail et pour ses réponses claires à nos questions en lien à l’infrastructure du logiciel Intenso.

Finalement, nous aimerions remercier Professeur Alain April du département de génie logiciel de l’École pour son expertise et ses conseils qui, tout au long du projet, nous ont aidés à obtenir une solution viable et nous ont permis de développer nos connaissances sur les concepts de l’infonuagique.

Résumé

VIF Télé est une entreprise montréalaise spécialisée dans l'affichage numérique. Elle dessert un peu moins de 4000 points d'affichage partout dans le monde. VIF Télé est récipiendaire de deux prix des Global Digital Signage Awards pour ses différents projets d'affichage.

La solution logicielle Intenso est ce qui permet à VIF Télé de se démarquer dans le domaine de l'affichage numérique. La gestion de l'infrastructure sur laquelle repose Intenso représente de nombreux défis pour l'entreprise.

L'objectif visé par ce projet est de d'expérimenter une preuve de concept d'architecture infonuagique sur laquelle VIF Télé pourrait migrer ultérieurement sa solution logicielle sans avoir à se soucier de la gestion de son infrastructure à l'avenir.

Table des matières

Liste des tableaux

Liste des figures

Liste des abréviations, sigles et acronymes

Abréviation, sigle ou acronyme	Définition
PaaS	*Platform as a Service*
IaaS	*Infrastructure as a Service*
SaaS	*Software as a Service*
PME	Petite Moyenne Entreprise
AWS	Amazon Web Services
GCP	Google Cloud Platform
CI/CD	Continuous Integration / Continuous Delivery
IIS	*Internet Information Services*
SDK	*Software Development Kit*
API	*Application Program Interface*

Mise en contexte

À propos du promoteur

VIF Télé est une PME montréalaise se spécialisant dans l'affichage numérique. Elle compte un peu moins de 4000 points d'affichage à travers le monde. VIF Télé est récipiendaire de deux prix des Digital Signage Awards pour ses divers projets.

Logiciel

Intenso est le logiciel d'affichage numérique développé et utilisé par VIF Télé. Ce logiciel est composé de trois éléments principaux : la *WebInterface* permet à VIF Télé et à ses clients de gérer le contenu affiché sur leurs écrans, le *MediaServer* effectue la distribution de ce contenu aux points d'affichages et le *Intenso Player* est l'application installée sur les points d'affichage chez les clients qui permet d'afficher le contenu programmé.

Problématique

La problématique principale que rencontre VIF Télé est qu'actuellement, leurs applications back-end (la *WebInterface* et le *MediaServer*) sont hébergées sur des serveurs physiques loués chez un hébergeur. Ces serveurs sont de type IaaS (*Infrastructure as a Service*), ils sont extrêmement flexibles et répondent amplement aux besoins de l'entreprise en matière de performance. Étant donné la nature des architectures IaaS, le promoteur doit lui-même gérer les couches sous-jacentes à ses applications, par exemple le réseau, le système d'exploitation et ses mises à jour ainsi que l'espace de stockage. L'entreprise ne disposant pas de ressources qualifiées pour effectuer la gestion de ces serveurs, les développeurs doivent effectuer ce travail au meilleur de leur connaissance. Les pannes de serveur qui surviennent représentent donc un casse-tête pour l'entreprise. Les coûts qui y sont associés sont plus élevés, le temps de panne étant parfois plus élevé vu les connaissances limitées des développeurs. La faible fréquence de ces pannes ne justifie pas l'embauche d'une ressource dédiée à ce problème.

Besoins du promoteur

En lien avec la problématique, VIF Télé a identifié quelques points de pression sur lesquels notre équipe travaillera dans le cadre du projet, soit :

- Utiliser les technologies infonuagiques pour migrer la solution vers une architecture PaaS, ce qui réduira la charge sur les épaules de développeurs qui pourront uniquement se concentrer sur le développement d'Intenso ;
- Décomposer Intenso en trois entités indépendantes au niveau de l'infrastructure pour améliorer le processus de déploiement et diminuer les impacts d'une panne de serveur sur la solution.

Présentation du projet

Objectifs du projet

L'objectif principal du projet est de répondre aux besoins du promoteur, tels que mentionnés dans la section précédente. Les membres de l'équipe doivent prendre conscience des problèmes entourant la solution logicielle actuellement en place et adopter des solutions pour les mitiger. Ultimement, l'équipe livrera une solution logicielle fonctionnelle et documentée, ainsi que des recommandations sur les améliorations possibles pour perfectionner la solution.

En second lieu, les membres de l'équipe ont comme objectif d'enrichir leurs connaissances en matière de technologies infonuagiques et d'intégration d'applications sur les plateformes comme Azure de Microsoft. Le partage de connaissances est donc essentiel et une bonne compréhension de la solution livrée au client représente un barème de succès pour le projet.

Architecture existante

Comme mentionné plus haut, les applications dorsales (c.-à-d. du back-end) de VIF Télé et leur dépendance sont présentement hébergées sur un même serveur physique fonctionnant avec un système d'exploitation *Windows Server*. Ce serveur exécute une instance d'IIS sur laquelle les applications reposent ainsi qu'une instance de SQL Server contenant une seule base de données. Le stockage de médias est fait directement sur le disque dur du serveur et le MediaServer s'assure de desservir ces fichiers aux clients. Pour gérer les versions du code source, VIF Télé utilise un serveur Mercurial interne, lui aussi hébergé sur un serveur physique loué chez le même hébergeur.

Membres de l’équipe

Nom	Responsabilités
Aubert Guillemette	● Communiquer avec VIF Télé ● Transférer ses connaissances de la plateforme Intenso aux autres membres de l’équipe ● Migrer la base de données d’Intenso vers Azure ● Migrer la WebInterface sur un conteneur Docker Windows ● Intégrer le Blob Storage dans la WebInterface
Albert Villeneuve-Nguyen	● Scinder le projet en plusieurs modules ● Intégrer le media-server dans un Application Service ● Créer les projets indépendants pour faciliter l’intégration continue ● Rechercher sur l’intégration *serverless* du projet
Xavier Langlais-Savage	● Élaborer une pile technologique ● Créer, documenter et déployer l’architecture de base sur Azure ● Mettre en place la solution d’intégration continue des services déployés sur Azure avec GitLab ● Optimiser les conteneurs Docker

Tableau 1: Membres de l’équipe

Gestion du projet

L’équipe et le superviseur ont convenu, au début du projet, de faire des rencontres de type SCRUM hebdomadairement, pour faire le point sur l’avancement du projet. Plus tard au courant de la session, l’équipe et le superviseur ont convenu de changer la fréquence de ces réunions pour en faire des rencontres bimensuelles. Un espace Slack a été mis en place pour pouvoir communiquer avec le superviseur du projet. De plus, un groupe Messenger a aussi été créé par l’équipe du projet pour la communication intermembre de manière plus directe. Pour faire la liste et le suivi des tâches à effectuer, un tableau Trello a été créé pour pouvoir en suivre la progression. Ce tableau a été affiché sur un écran durant toutes les rencontres SCRUM faites entre les membres de l’équipe de projet et le superviseur.

En plus des rencontres entre l’équipe et le superviseur, des rencontres avec le client ont aussi été organisées afin de confirmer les besoins du client et faire le suivi avec eux de l’avancement du projet.

Livrables

Nom	Description
Présentation finale du projet	Présentation du projet, de son déroulement et de ses résultats au professeur superviseur.
Rapport final du projet	Document présentant le projet en détail, le déroulement de celui-ci, son architecture et ses choix/contraintes de conception.
Code source	Code pour déployer l'infrastructure décrite dans ce document.
Document de recommandations	Document faisant part des recommandations de l'équipe de projet à VIF Télé en ce qui concerne le futur de ce projet.

Tableau 2: Livrables du projet

Risques

Description du risque	P	C	E	Réduction du risque
Manque d'expérience avec la plateforme Azure de Microsoft	3	2	6	Lire la documentation et poser des questions aux personnes-ressources à notre disposition.
Manque de familiarité avec le logiciel Intenso	3	3	6	Explorer le code source et poser des questions aux développeurs au besoin.
Présence de composantes très couplées avec le système d'exploitation Windows	2	2	4	Utiliser des conteneurs Windows afin de mitiger le problème.
Manque de temps causé par les autres cours	3	3	9	Se réserver une période par semaine pour travailler sur le projet
Manque d'expérience avec les outils d'intégration continue (GitLab CI, Terraform, Azure CLI)	2	2	4	Lire la documentation disponible, rechercher des articles et exemples sur le net.

Tableau 3: Grille de documentation des risques

Technologies

Une grande partie du travail qui a été effectué dans le cadre du projet concerne l'adoption de nouvelles technologies. Différents choix technologiques ont donc dû être faits par l'équipe et d'autres nous ont été imposés par le client et les limitations de son logiciel actuel. En somme, l'adoption de ces technologies vise à combler les besoins du client et à apporter des solutions à ses problèmes.

Fournisseurs infonuagiques

Initialement, l'équipe avait identifié deux fournisseurs cloud majeurs : Amazon Web Services et Azure. Amazon étant un concurrent direct à plusieurs clients de VIF Télé œuvrant dans le commerce de détail, AWS a rapidement été écarté de nos options. Conjointement avec le client, nous avons choisi d'opter pour la plateforme Azure de Microsoft. C'est sur cette plateforme que sera hébergée la base de données, les applications et les fichiers de la solution d'affichage numérique offerte par VIF Télé.

Conteneurs

Les applications « back-end » fournies, c'est-à-dire le serveur de fichiers et l'interface web, étaient originalement conçues pour être exécutées sur un serveur physique. Elles sont présentement déployées chez un fournisseur et gérées par les employés de VIF Télé. Nous avons écarté l'option de créer des machines virtuelles sur la plateforme Azure afin d'y déployer les applications, pour plutôt utiliser des conteneurs. Faciles à construire et à déployer, les conteneurs éliminent le besoin de gérer des machines physiques, en plus d'épargner des coûts d'exploitation sur la plateforme Azure.

En raison des fortes dépendances des applications envers l'environnement de Windows et des outils de développement de Microsoft, les conteneurs doivent supporter ce système d'exploitation. Pour la création de conteneurs, nous utilisons Docker. En plus d'être le plus présent sur le marché, le logiciel Docker est le seul système qui supporte actuellement les conteneurs Windows.

Gestion de versions et intégration continue

Lorsque le code source des applications originales nous a été fourni, il était sous la forme d'un dépôt Mercurial. Comme mentionné plus haut, ce dépôt était hébergé sur un serveur physique administré par VIF Télé, une chose qui n'était pas désirée par le client. Étant donné la dette technique qu'est le choix de Mercurial comme système de contrôle et la grande présence de Git sur le marché, autant en ce qui concerne les connaissances des développeurs que les outils qui leur sont disponibles, nous avons décidé de convertir le dépôt vers Git.

En ce qui concerne l'intégration continue, le client n'avait aucune solution en place et les déploiements étaient faits manuellement, avec l'aide d'un script exécuté sur le serveur en production.

Pour héberger le code et gérer les tâches d'intégration et de déploiement en continu, nous avons identifié deux potentielles offres SaaS présentes sur le marché : Azure DevOps et GitLab. Ces deux produits offrent des fonctionnalités équivalentes à des prix similaires. Le premier se démarque par sa facilité d'adoption dans un environnement Azure, ce qui élimine la gestion de comptes et de facturation sur une plateforme additionnelle. De son côté, GitLab se démarque par sa maturité, la qualité de sa documentation et sa grande communauté d'utilisateurs, manifestée par de nombreux articles et exemples disponibles sur le Web. C'est pour cette raison que nous avons opté pour GitLab.

Terraform

Terraform est un outil du domaine du logiciel libre axé sur l'approche « *infrastructure as code* ». Plus précisément, il permet de définir et déployer des ressources sur les plateformes de différents fournisseurs infonuagiques, dont : Azure, AWS et GCP. Dans le cadre du projet, il est utilisé pour définir toutes les ressources que nous utilisons sur Azure, dont le registre de conteneurs et la base de données.

.NET Core / .NET Framework

La code d'Intenso utilise une ancienne version du cadriciel .NET. À partir de la première ébauche d'architecture, il a été convenu par les membres du projet de migrer Intenso vers la dernière version du cadriciel .NET pour ensuite faire la migration de la solution vers le cadriciel .NET Core. Cependant, dû à une réévaluation de l'architecture du projet de la part de l'équipe, nous avons décidé de rester sur le cadriciel .NET. Ce choix sera expliqué plus en détail dans les sections subséquentes.

Services Azure utilisés

Azure SQL

Azure SQL est un service de base de données offert sur la plateforme Azure. Il permet la création et l'utilisation de bases de données de type SQL Server selon différents modèles d'exploitation. Le tableau ici-bas fait la description des différents types de déploiement de bases de données Azure SQL et des modèles d'exploitation infonuagiques qui y sont associés.

Type de déploiement	Description	Modèle d'exploitation
Base de données SQL	Permet de déployer des bases de données unique ou des « pools » élastiques de bases de données. Complètement géré, option de calcul « serverless » disponible.	PaaS
Instances gérées SQL	Similaire au type *Base de données SQL*, mais ajoute une couche de compatibilité supplémentaire avec les instances SQL Server physiques ainsi qu'une gestion approfondie du réseau virtuel.	Hybride PaaS / IaaS
Machines virtuelles SQL	Offre un accès complet au système d'exploitation et à l'instance SQL Server. Ce type de déploiement permet aussi la gestion des versions de SQL Server et l'automatisation de tâches. Équivalent à avoir une instance SQL Server s'exécutant sur une machine virtuelle.	IaaS

Tableau 4: Types de déploiement de bases de données Azure SQL

Dans le cadre de ce projet, puisque l'architecture d'origine utilise une seule base de données qui est utilisée par deux applications différentes, nous avons décidé d'utiliser le type de déploiement *Bases de données SQL* et d'utiliser une base de données unique. Ce choix permet d'enlever la complexité associée à la gestion d'une instance SQL Server complète, ce qui répond aux besoins de VIF Télé.

Blob Storage

Azure Blob Storage est un service de stockage infonuagique qui se concentre sur le stockage de données non structurées. Microsoft décrit ce service comme idéal pour, entre autres, desservir des images et du contenu vidéo ainsi que de fournir un accès distribué à des fichiers. Puisque le *MediaServer* d'Intenso doit desservir des fichiers à un nombre importants de clients, ce système de stockage de données est plus avantageux que les autres systèmes fournis par Azure, comme par exemple, « *File Storage* ».

Container Registry

Azure Container Registry est un service effectuant principalement le stockage et le bâtissage des images de conteneurs Docker. Ce service gère l'authentification afin de sécuriser les images et permettre aux développeurs d'y accéder par l'interface en ligne de commande de Docker, sur leurs postes. Dans le cadre du projet, ce service est essentiel puisqu'il permet de construire des images Docker avec Windows comme système d'exploitation, chose qui serait impossible dans un contexte d'intégration continue sans l'utilisation de machines virtuelles.

Container Instances

Ce service d'Azure sert à instancier un conteneur et l'exposer sur une adresse IP publique. Ce conteneur peut provenir d'un registre de conteneurs quelconque, par exemple celui d'Azure.

App Service

Azure App Service est un service permettant de construire, déployer et mettre à l'échelle des applications web. C'est un service clés en main permettant de déployer une application de façon sécuritaire et performante, sans se soucier de l'architecture sous-jacente.

Active Directory

Riche en fonctionnalités, Azure Active Directory est une implémentation du logiciel Active Directory de Microsoft sous forme de services Web. Ce service permet entre-autres l'authentification unique, c'est-à-dire l'utilisation d'un seul compte pour accéder à différents services. Il pourrait par exemple être utilisé pour authentifier les clients de VIF Télé à l'interface web (Web Interface) de la solution logicielle du client. C'est aussi sur ce service que sont gérés les utilisateurs se connectant à la plateforme Azure pour gérer les services, par exemple des développeurs. Enfin, Azure Active Directory peut être utilisé pour définir un *Service Principal*, c'est-à-dire un utilisateur virtuel effectuant la gestion de l'infrastructure Azure. Dans le cadre du projet, celui-ci est utile pour automatiser les déploiements sur la plateforme d'intégration continue.

Application Gateway

Utilisé comme point d'entrée pour diriger les clients vers les applications déployées avec Container Instances et App Service, ce service permettra d'effectuer des déploiements en réduisant la durée d'indisponibilité, tout en permettant de déployer l'infrastructure dans différentes régions.

Virtual Networks

Ce service permet de créer un réseau virtuel interne à la plateforme Azure permettant d'isoler et sécuriser les machines virtuelles, conteneurs et autres applications déployées sur la plateforme. Virtual Networks offre la possibilité de bloquer, limiter ou filtrer le trafic entrant et sortant. Ce service permet aussi d'établir des connexions sécurisées inter régions et avec des serveurs hébergés chez un client. Une adresse IP publique peut être assignée au réseau afin d'y créer un point d'entrée. Virtual Networks est essentiel pour répondre aux besoins de certains clients de VIF Télé de conserver leurs données à l'intérieur d'un pays ou d'une région.

Intégration des technologies

Architecture applicative initiale

La figure ici-bas représente l'architecture initiale de la solution Intenso sur le serveur. Comme le décrit la figure, tous les services sont regroupés sous le même serveur. Les clients, ici représentés par *Utilisateur*, se connectent à la *WebInterface* pour faire la programmation des médias à jouer sur leurs écrans. Les fichiers téléversés par l'utilisateur sont stockés dans un dossier sur le système de fichiers du serveur et les informations relatives à ce fichier et sa programmation est stockée dans la base de données. L'application client (*Intenso Player*) peut ensuite contacter le *MediaServer* pour télécharger les configurations et le média programmé sur l'écran.

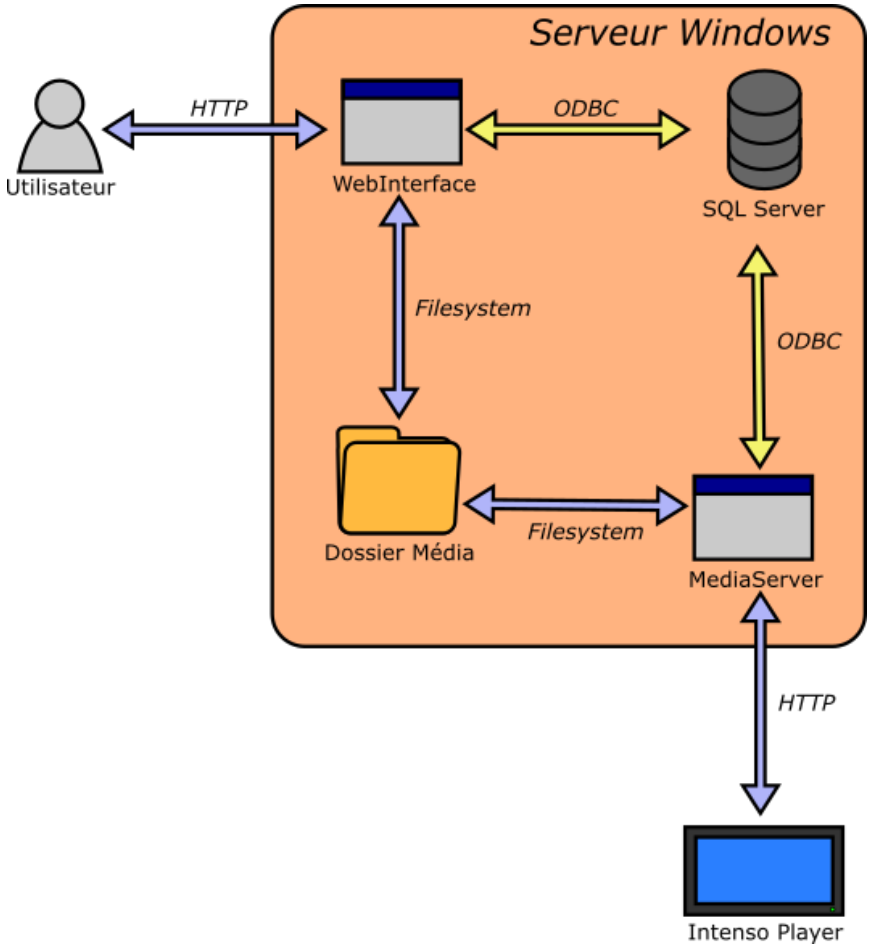

Figure 1: Architecture initiale d'Intenso

Architecture applicative actuelle

Comme on peut le constater avec la figure ici-bas, bien que les éléments de l'architecture changent, le fonctionnement lui reste relativement semblable. On peut cependant observer une séparation plus distincte de la *WebInterface* qui se retrouve maintenant dans un conteneur Docker et du *MediaServer* qui est désormais contenu dans un « *App Service* ». La base de données est migrée vers une base de données *Azure SQL* mais la nature connections à celle-ci ne change pas (toujours via ODBC). L'élément ayant le plus changé est le stockage de fichiers qui est maintenant fait sur le *Blob Storage*. Le *MediaServer* et la *WebInterface* accèdent maintenant aux fichiers via l'API du « *Blob Storage* » fourni par Azure et l'application client télécharge directement les fichiers du « *Blob Storage* » au lieu de passer par le *MediaServer*, ce qui allège donc la charge sur celui-ci.

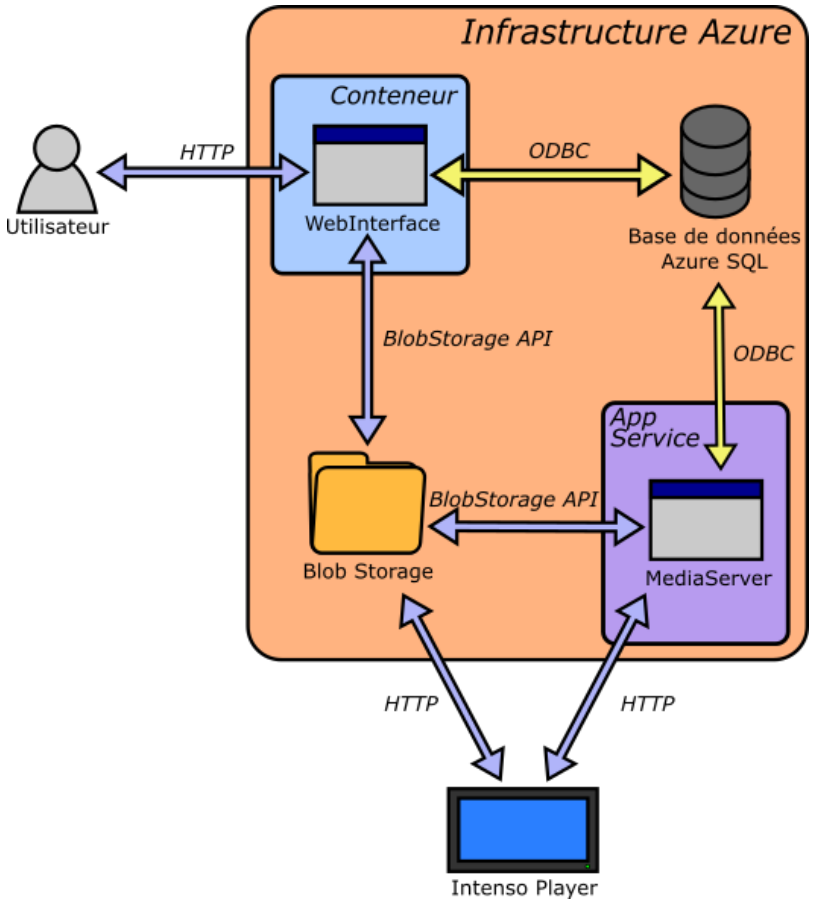

Figure 2: Architecture proposée pour Intenso

Architecture des services Azure

La figure ci-dessous représente l'architecture des services de la plateforme Azure de Microsoft pour supporter le logiciel Intenso, ainsi que leurs interactions. Afin de simplifier le diagramme, les protocoles de communication n'ont pas été mentionnés. Mis à part la communication avec la base de données réalisée avec le protocole ODBC, toutes les communications se font avec le protocole HTTP.

Les services qui pourraient être utilisés pour répondre aux besoins de surveillance (monitorage), de sécurité ainsi que de maintenance de la base de données ne font pas partie de la solution logicielle livrée au client et ont été omis du diagramme. Ces services feront néanmoins l'objet des recommandations qui seront livrées au client.

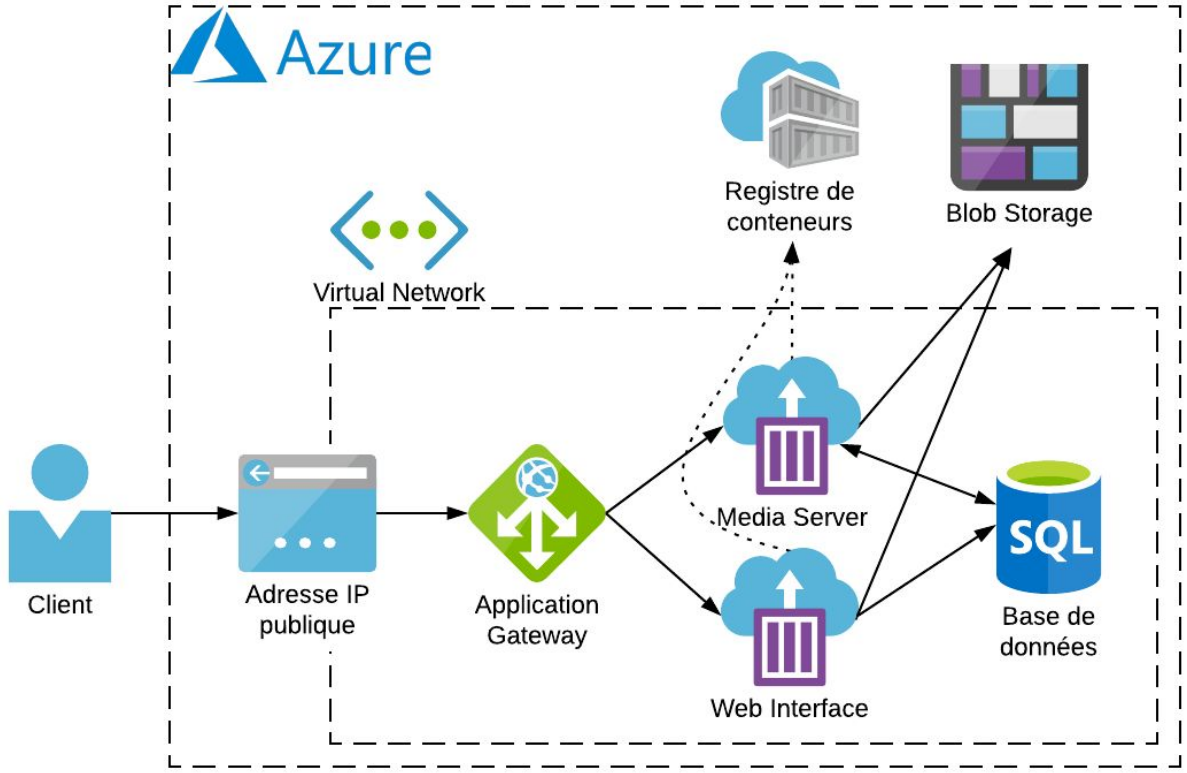

Figure 3: Architecture des services Azure proposée

Architecture CI/CD

Souvent utilisé dans l'industrie, le terme CI/CD a deux différentes définitions. D'abord, CI signifie « *Continuous Integration* », et CD signifie à la fois « *Continuous Delivery* et *Continuous Deployment* », respectivement livraison et déploiement en français. On peut comparer les concepts de livraison et de déploiement de logiciels à celui pour un meuble. Un meuble livré correspondrait à une boîte contenant toutes les pièces et un manuel d'instruction pour les assembler. Le meuble déployé serait déjà assemblé et fonctionnel. Dans le cadre du projet, CI/CD est utilisé dans les deux contextes.

D'abord, l'architecture de base, nommée *azure-baseline* dans ce contexte-ci, est déployée dans un environnement Azure. Ensuite, les processus de CI/CD de chaque application qui sera déployée grâce aux services Azure se font dans un contexte de livraison continue, et l'objectif de ces processus est la génération de l'image d'un conteneur. Enfin, un dépôt de code, nommé *intenso-azure*, a la responsabilité de regrouper l'information des services déployés sur Azure afin de définir la solution logicielle d'Intenso. C'est à cet endroit que sont spécifiées les configurations pour le déploiement dans différents environnements.

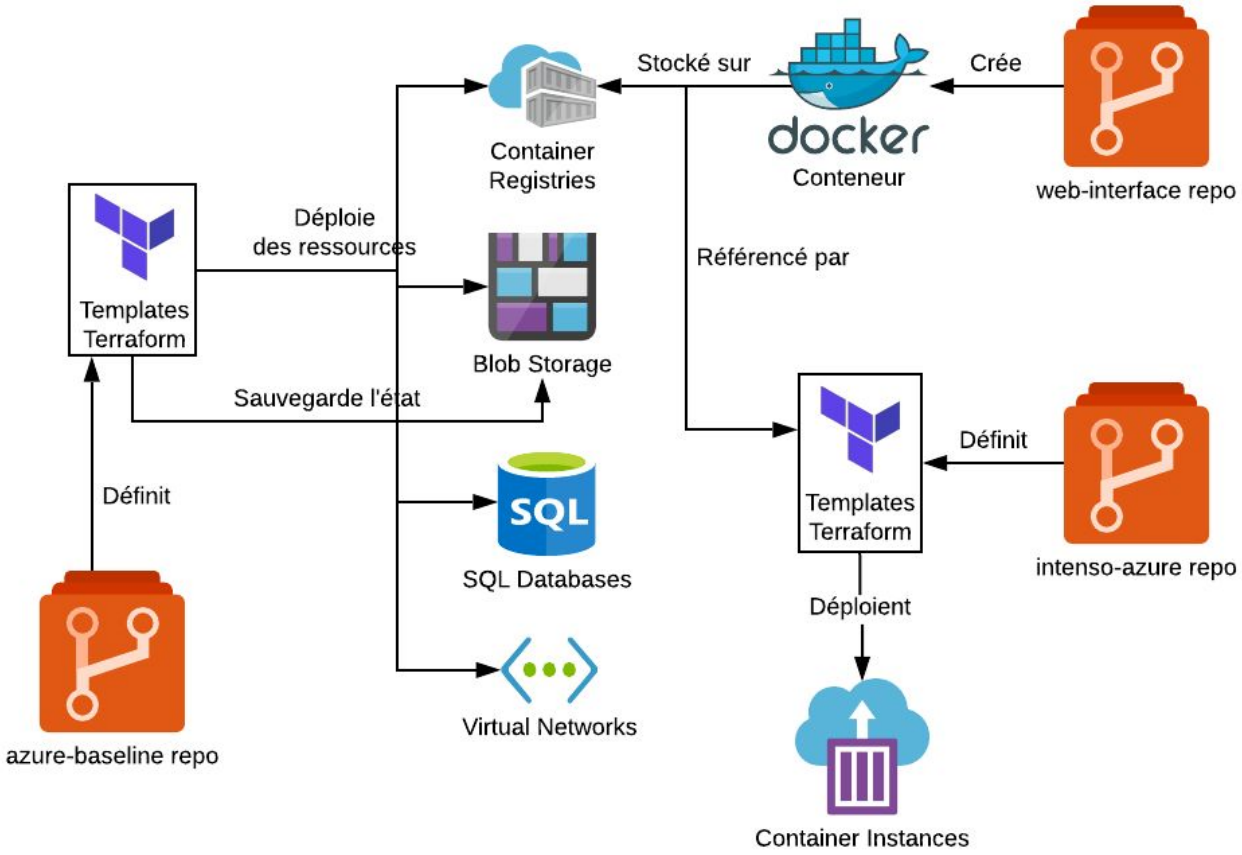

Figure 4: Architecture CI/CD du projet

Déroulement

Première rencontre avec Vif

La première rencontre avec les représentants de VIF Télé nous a permis d'établir avec eux la problématique encourue par l'entreprise, ses besoins ainsi que les attentes des représentants envers le projet. Cette rencontre nous a aussi permis de prendre connaissance du contexte d'affaire de l'entreprise et des contraintes auxquelles ferait face le projet, par exemple l'impossibilité d'utiliser AWS, tel qu'expliqué plus haut.

Ébauche d'architecture

Au début du projet, le code devait converti puis déployé sur une architecture « *serverless* ». Ce type d'architecture, quoique supérieure d'un point de vue monétaire et au niveau de l'évolutivité, demandait d'importants changements au niveau du système d'authentification déjà en place sur les logiciels *back-end* de VIF Télé.

La complexité principale dans l'implémentation d'une architecture « serverless » dans le projet fut que plusieurs paramètres passés aux applications *back-end* (dont les informations de connexion) étaient contenus dans une variable de session. Le fait qu'une variable de session se trouve à être des données qui sont stockées dans un cookie de manière permanente ou du moins à long terme rendait la mise en place de l'application beaucoup plus complexe. C'est donc pourquoi la migration vers ce type d'architecture a rapidement été écartée. Nous avons plutôt opté pour une architecture migrant l'application dans un conteneur, ce qui nous a permis de conserver le système d'authentification original.

Élaboration de la pile technologique

L'adoption de nouvelles technologies représente un enjeu pour une entreprise. Les décisions prises ont potentiellement un énorme impact sur les processus de l'entreprise. C'est pourquoi il faut analyser ses besoins, ceux de ses clients et ceux de son personnel.

D'abord, VIF Télé doit gérer de nombreuses données, certaines d'entre elles étant critiques. Pour cette raison, ces données doivent être chiffrées, autant en transit qu'au repos. Pour plusieurs clients, ces données ne doivent pas quitter le Canada, ce qui peut réduire les options de l'entreprise dans le choix de produits tiers.

En ce qui concerne les développeurs, les enjeux suivants ont été documentés :

- Adopter des technologies bien documentées et simples d'utilisation, étant donné la rotation du personnel et le grand nombre de stagiaires engagés par l'entreprise ;
- Automatiser ou permettre l'automatisation de toute tâche répétitive ;
- Ne pas empêcher ou rendre plus complexe le développement local des applications web, comme leur développement est coûteux en temps et en argent.

À la lumière des enjeux et besoins mentionnés précédemment, les éléments suivants ont été décidés :

- Choix d'Azure comme fournisseur infonuagique ;
- Utilisation des services Azure, dans une architecture telle que représentée à la figure 3 de ce document ;
- Migration des médias vers Azure Blob Storage ;
- Utilisation de GitLab pour la gestion du code source, de l'intégration et du déploiement en continu ;
- Utilisation de Docker pour la création d'images de conteneurs ;
- Utilisation de Terraform pour gérer le déploiement de l'infrastructure de base, des applications et de leur configuration.

Mise en place de l'infrastructure de base

Suite à l'élaboration de la pile technologique, des modèles (gabarits) pour l'outil Terraform ont été créés et documentés pour la création de différentes ressources Azure. Plus précisément, il s'agit d'un registre de conteneurs, d'un réseau virtuel et d'un espace de stockage Azure Blob. Sur GitLab, un dépôt de code et les pipelines de déploiement ont été créés et mis en œuvre. Les tâches d'initialisation d'un espace sur la plateforme Azure, celles-ci ne pouvant pas être automatisées, ont été documentées et testées.

Migration de la base de données

La migration de la base de données a commencé par l'étude des différentes options offertes par *Azure SQL*. Comme mentionné plus haut dans la section *Technologies*, nous avons opté pour une instance unique de l'option de déploiement *Base de données SQL*. Cette option a l'avantage de nous fournir une base de données complètement gérée, ce qui cadre avec les besoins de VIF Télé. Microsoft décrit cependant ce type de déploiement incompatible avec le « *Lift-and-Shift* ». Or, l'outil *SQL Server Management Studio* de Microsoft fournit une fonctionnalité de déploiement de bases de données vers Azure, cette fonctionnalité permet même de choisir quel type de déploiement utiliser. Nous avons donc pris la structure de la base de données de VIF Télé déjà existante et avons utilisé cet outil pour la déployer vers *Azure SQL*. Par la suite, la seule opération restante à effectuer a été de changer les chaînes de connexion dans les deux applications « *back-end* ».

Migration de la WebInterface sur un conteneur Docker

Le déploiement de la *WebInterface* dans un conteneur Windows a été plus complexe, puisque les dépendances sur lesquelles le projet repose étaient mal définies. Nous avons découvert qu'en plus de reposer sur le cadriciel .NET Framework, Intenso utilise aussi des composantes du SDK de Windows 8 et un fichier « *target* » retrouvé dans Visual Studio. Après avoir identifié ces dépendances, nous avons créé une image Docker contenant les dépendances d'Intenso, une instance d'*IIS* ainsi que les scripts pour compiler l'application. Lors de la génération de cette image, les dépendances d'Intenso sont installées, le logiciel

lui-même est compilé puis le serveur *IIS* contenu dans l'image dessert l'application en production. L'un des problèmes rencontrés avec cette approche est que le temps de déploiement de l'image est relativement grand. C'est pourquoi nous avons décidé de construire une image intermédiaire avec toutes les dépendances d'Intenso pour alléger le temps de déploiement de l'image desservant l'application.

Intégration du Blob Storage

L'intégration de Azure Blob Storage s'est faite sans trop d'encombres, la complexité principale rencontrée étant la gestion des références d'*assembly* manquantes lors de l'ajout du paquet NuGet de l'API du *Blob Storage* dans les applications. L'intégration de l'API dans les applications a nécessité simplement le remplacement des fonctions interagissant directement avec le système de fichier. Autrement, cette intégration a été complètement transparente.

Migration vers Azure App Services

Comme pour la migration de la *WebInterface* sur un conteneur, la migration du *MediaServer* sur l'*App Service* a été plus ardue, car les dépendances d'Intenso étaient mal définies. Hormis les difficultés liées aux dépendances du projet, l'intégration d'une application dans l'*App Service* se fait de manière intuitive avec Git pour le déploiement d'une application.

Problèmes rencontrés

Communication entre les membres de l'équipe

Au début du projet, l'espace de travail sur la plateforme Slack a été créé dans le but d'être la principale plateforme de communication pour le projet. Tout au long de celui-ci, cette plateforme s'est avérée sous-utilisée. Comme les trois membres de l'équipe se côtoient dans divers contextes autres que celui du projet, une majorité des communications se sont effectuées de vive voix, dans un contexte informel. De plus, les membres de l'équipe ayant aussi créé une conversation de groupe dans l'application Messenger de Facebook, la plupart des communications entre les membres ont été effectuées sur celle-ci.

En somme, les différents choix de l'équipe ont contribué à réduire la transparence du projet envers le professeur-superviseur, celui-ci ayant seulement accès à l'espace de travail sur Slack.

Architecture proposée du MediaServer incompatible avec le système d'authentification en place

Le *MediaServer* est l'application *back-end* à laquelle se connectent tous les clients Intenso (*Intenso Player*). Ceux-ci doivent s'authentifier au MediaServer pour pouvoir ensuite pouvoir se servir de ses différentes fonctions. Un des changements proposés au début du projet était de convertir l'architecture du *MediaServer* vers une architecture *serverless*. Une architecture de ce type implique une application sans état (« *stateless* »). Or, le système d'authentification du *MediaServer* utilise des variables de session, ce qui fait du *MediaServer* une application avec états (« *stateful* »). Transformer cette application pour la rendre sans état implique une réingénierie complète du système d'authentification entre les clients et le MediaServer. Les membres de l'équipe, évaluant l'effort et le temps requis trop élevés pour les avantages apportés par ce type d'architecture, ont donc décidé de conserver l'architecture originale du *MediaServer* et de simplement migrer l'application vers un service de type « *App Services* ».

Conclusion

Ce projet de fin d'études avait pour but de développer une architecture infonuagique sur laquelle VIF Télé pourra déployer sa solution logicielle Intenso et continuer son développement sans avoir à se soucier de la gestion de son infrastructure.

Acquis relationnels et techniques

Les acquis relationnels et les apprentissages faits au cours de ce projet furent nombreux. Les relations développées entre les membres de l'équipe, l'enseignant superviseur de ce projet et l'entreprise qui a commandité le projet vont nous permettre d'avoir des contacts d'une grande valeur pour nos carrières.

En plus de ces nouvelles relations, l'équipe a développé des habiletés techniques sur plusieurs sujets, entre autres sur le langage C# avec le cadriciel .NET, sur le déploiement d'une architecture infonuagique en intégration continue et sur la conteneurisation. Ces acquis techniques pourront être réutilisés dans nos projets futurs et seront bénéfiques pour nos carrières.

Déroulement

Lors de ce projet, nous avons rencontré quelques difficultés. D'un point de vue technique, l'architecture d'Intenso s'est avérée différente que ce nous avions prévu, ce qui nous a forcés à revoir certaines décisions architecturales, notamment la décision d'utiliser une architecture « *serverless* ».

En plus des complexités techniques rencontrées durant le projet, la gestion des communications entre les membres de l'équipe et le professeur superviseur étaient parfois plus difficiles, de par nos horaires différents et de la mauvaise utilisation des plateformes de communication.

Recommandations

Finalement, nous aimerions faire quelques recommandations à VIF Télé pour améliorer sa plateforme Intenso afin qu'elle puisse utiliser le plein potentiel de l'architecture développée dans le cadre de ce projet et ainsi bien évoluer dans le futur:

1. Ajouter des tests unitaires au code source ;
2. Passer à une architecture « *stateless* » pour le *MediaServer* et la *WebInterface* ;
3. Migrer la base de données de clients pour l'authentification ;
4. Utiliser un « modèle de branching » Git, tel que le « *Trunk Based Development* » ;
5. Utiliser Git pour la gestion des versions ;
6. Automatiser les déploiements du code vers la *WebInterface* et le *MediaServer* ;
7. Automatiser les tests pour les version « builds »;
8. Centraliser la paramétrisation des applications et supporter les variables d'environnement.

Références

1. Microsoft, 2019. Choose the right deployment option in Azure SQL
   https://docs.microsoft.com/en-us/azure/sql-database/sql-database-paas-vs-sql-server-iaas

2. Microsoft, 2019. What is Azure Blob storage?
   https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blobs-overview

3. Microsoft, 2020. App Service Documentation
   https://docs.microsoft.com/en-us/azure/app-service/

4. VIF Télé, date inconnue. Notre histoire
   http://www.viftele.com/notre-histoire

5. HashiCorp, date inconnue. Terraform Documentation
   https://www.terraform.io/docs/index.html

6. GitLab, date inconnue. GitLab Pricing
   https://about.gitlab.com/pricing/

7. Microsoft, date inconnue. Azure DevOps Services pricing
   https://azure.microsoft.com/en-ca/pricing/details/devops/azure-devops-services/

8. Microsoft, 2020. AWS to Azure services comparaison
   https://docs.microsoft.com/en-us/azure/architecture/aws-professional/services

9. CloudBees, date inconnue, What is GitOps?
   https://www.cloudbees.com/gitops/what-is-gitops

ANNEXE 1 : Questions

1 - Évaluez l'étape de Formulation du problème (**Q4-i1**) : comment avez-vous réussi à bien décrire le problème à le résoudre en tenant compte des besoins et contraintes identifiées dans votre projet.

Afin de résoudre la problématique de notre client, nous avons d'abord rencontré ses représentants (soit le directeur des opérations et le superviseur du développement logiciel) pour comprendre leurs besoins, mais aussi prendre connaissance de leurs attentes envers notre solution. Cette compréhension du problème a été facilitée dû au fait qu'un des membres de l'équipe du projet a effectué deux stages au sein de VIF Télé, l'entreprise cliente. Il a donc été beaucoup plus facile de comprendre les besoins du client et les implications de la problématique.

2-Évaluez l'étape de Formulation de solutions (**Q4-I2**) : comment avez-vous réussi à décrire des solutions possibles et choisir celle qui a été préférée pour votre projet.

Pour trouver les solutions possibles à la problématique du client, nous avons d'abord consulté notre professeur-superviseur et fait avec lui un remue-méninge des possibilités s'offrant à nous. Ensuite, c'est après avoir fait des recherches et des expérimentations de notre côté que nous avons trouvé la solution idéale qui permettrait de répondre à tous besoins de notre client. Par la suite, nous avons fait quelques diagrammes (tel que présenté dans ce rapport) afin de comprendre les implications de la solution choisie.

3-Évaluez les Communications (**Q7-I3**) : comment avez-vous communiqué avec le client/utilisateurs et entre vous durant ce projet.

Dans un premier temps, les rencontres entre l'entièreté des membres de l'équipe du projet et le client ne se sont fait qu'à deux occasions dans les bureaux de VIF Télé. Ces rencontres nous ont permis de définir la problématique encourue et les besoins du client ainsi que de lui faire part de notre avancement. Cependant, la communication entre le client et l'équipe de projet s'est toujours faite régulièrement: un des membres de l'équipe s'occupait de l'échange d'information entre les parties prenantes chez le client et les membres de l'équipe. De plus, cet échange d'information était facilité par le fait que le membre responsable des communications dans l'équipe a complété deux stages chez le client. Cela a donc permis d'éliminer des ambiguïtés possibles dans l'échange d'information puisque ce membre connaissait la réalité du client ainsi que celle de l'équipe de projet.

Dans un deuxième temps, les communications entre les coéquipiers étaient très fréquentes. Les membres de l'équipe discutaient sur une base régulière de l'avancement du projet et des prochaines étapes à entamer.

4-Évaluez votre approche de gestion des Enjeux environnementaux (**Q9-I2**) : expliquez comment vous avez tenu compte de la réduction de votre empreinte écologique durant le projet.

Dans le cadre du projet, nous avons privilégié les transports en commun pour se rendre aux réunions organisées avec le client ou le professeur superviseur. Au niveau de la solution développée dans le cadre de ce projet, nous avons déployé nos applications dans des emplacements où les infrastructures utilisent des énergies renouvelables, par exemple, l'hydroélectricité. De cette manière, nous avons pu nous assurer de réduire l'impact écologique des activités liées à notre projet.

5-Évaluer votre capacité d'effectuer une recherche judicieuse de solutions et de la nécessité de compréhension de nouveaux concepts qui s'appliquent pour proposer des solutions pour ce projet.

Dans le cadre de ce projet, il nous a été demandé de comprendre des concepts avec lesquels nous n'étions pas familiers, notamment l'infonuagique, les architectures « serverless » (dans le contexte de Microsoft Azure), les conteneurs (Docker) et certains concepts propres à la plateforme Azure (App Service, services managed et autres). Après avoir compris les besoins du client et énoncé certaines pistes de solution, nous avons lu la documentation disponible sur ces concepts afin de trouver l'information nécessaire afin de réaliser nos tâches.

ANNEXE 2: Comptes rendus des réunions avec le promoteur

Première rencontre:

- Ordre du jour :
  - Présentations
  - Cerner la problématique
  - Choix du fournisseur infonuagique
  - Architecture existante et architecture proposée
  - Contraintes
  - Livrables
- Présentations :
  - VIF Télé:
    - Benoit Johnson, Président et fondateur
    - Vincent Chouinard-Laliberté, Directeur des opérations
    - Sébastien Meunier, Superviseur du développement logiciel
    - Robert Stampfler, Consultant
- Cerner la problématique :
  - Éviter de gérer une infrastructure matérielle
  - Se débarrasser de certaines contraintes matérielles
- Choix du fournisseur infonuagique :
  - Amazon est *persona non grata* chez VIF Télé
    - Plusieurs clients étant en concurrence directe avec Amazon
  - VIF Télé a déjà un compte commercial chez Azure
  - Dois être hébergé au Canada
- Contraintes :
  - Application en C# / .NET
  - Impossible d'utiliser Amazon
- Livrables :
  - Code source de l'infrastructure dans un dépôt Git

Deuxième rencontre:

- Ordre du jour
    - Suivi de l'avancement du projet
    - Précisions sur les livrables
    - Support financier pour Azure
    - Précision de certaines décisions architecturales
- Suivi de l'avancement du projet
    - Mise à jour sur l'avancement du projet
    - Mise à jour sur les prochaines étapes du développement
- Précisions sur les livrables
    - Rapport final du projet (guide d'infrastructure)
    - Demande de recommandations
    - Dépôt de code Git
- Support financier
    - L'équipe aura possiblement besoin d'un budget de développement pour la migration et l'hébergement Azure
- Précision de certaines décisions architecturales
    - Pas d'infrastructure « *serverless* »
        - Implique trop de changements architecturaux
        - Implique une réingénierie du système d'authentification d'Intenso
            - Migration vers LDAP de la base de données des clients