UNIVERSITÉ SORBONNE PARIS NORD

IUT Villetaneuse
Université Sorbonne Paris Nord

RAPPORT ARCHITECTURE LOGICIELLE

M.AOMAR OSMANI

Sommaire :

Présentation :

L'architecture est l'élément le plus important d'un logiciel, c'est un choix qui détermine de la facilité développement, du traitement des données, des communications et des performances. De ce fait, on comprend que le choix adéquat de l'architecture peut permettre à un projet de répondre efficacement aux besoins recherchés par les développeurs.

Pour ce projet, il nous a été demandé de trouver un logiciel, de comprendre comment il fonctionnait et quelle architecture serait la mieux adapté à ses besoins. Nous avons décidé de faire ce travail sur Twitch.

Twitch est une plateforme de streaming en direct qui permet aux utilisateurs de diffuser et de regarder des vidéos en temps réel crée en 2011. L'attrait principal de la plateforme est l'aspect communautaire qui s'en dégage. Le principe de Twitch est de pouvoir s'enregistrer en direct tout en interagissant avec des spectateurs via un chat, le contenu principal de la plateforme sont les jeux vidéos mais il est également possible de diffuser d'autres types de contenus comme l'art, la musique, le sport, et même de simples discussions avec le chat.

Aspect Business :

Croissance et Popularité

Twitch, une plateforme de streaming en direct axée principalement sur les jeux vidéo, a connu une croissance exponentielle depuis sa création. En 2021, Twitch comptait environ 31 millions de visiteurs quotidiens et plus de 2,5 millions de spectateurs simultanés à tout moment. Cette popularité a attiré l'attention des marques et des publicitaires, faisant de Twitch un acteur majeur dans le domaine de la publicité numérique et du marketing d'influence.

Modèle Économique

Le modèle économique de Twitch repose sur plusieurs piliers :

Abonnements : Les utilisateurs peuvent s'abonner à des chaînes pour obtenir des avantages exclusifs.

Publicité : Twitch génère des revenus importants grâce à la diffusion de publicités pendant les streams.

Bits et Cheers : Les spectateurs peuvent acheter des Bits, une monnaie virtuelle, pour soutenir leurs streamers préférés.

Partenariats et Sponsoring : Twitch collabore avec des marques pour des partenariats sponsorisés et des événements spéciaux.

Acquisition par Amazon

En 2014, Twitch a été acquis par Amazon pour près de 970 millions de dollars, intégrant ainsi les ressources et l'infrastructure d'Amazon pour améliorer ses services et son évolutivité. Cette acquisition a permis à Twitch de bénéficier des capacités d'Amazon Web Services (AWS) pour héberger ses streams et d'intégrer des fonctionnalités telles que l'achat direct de jeux via Amazon.

Aspect Technique :

Architecture Microservices

Twitch utilise une architecture de microservices, ce qui signifie que différentes fonctionnalités de la plateforme sont divisées en services indépendants. Cette approche facilite la scalabilité et la maintenance. Chaque microservice peut être développé, déployé et mis à jour indépendamment des autres.

Ingestion et Diffusion Vidéo

Pour la gestion des flux vidéo en direct, Twitch utilise un système appelé Intelligest. Ce système comprend des proxys médias situés aux Points de Présence (PoPs) dans le monde entier, qui dirigent les flux vidéo vers les origines appropriées en fonction de décisions en temps réel prises par le service Intelligest Routing (IRS). Cette configuration optimise l'utilisation des ressources informatiques et assure la fiabilité et l'évolutivité des flux vidéo.

Réseau de Points de Présence (PoPs)

Twitch maintient un réseau global de PoPs pour minimiser la latence et garantir une haute qualité de diffusion. En réduisant la dépendance aux CDN tiers, Twitch peut mieux contrôler la qualité et la livraison de ses streams. Cette infrastructure est soutenue par AWS, offrant l'évolutivité et la fiabilité nécessaires pour les opérations globales de Twitch.

Sécurité et Conformité

La sécurité est intégrée dans l'architecture logicielle de Twitch. Cela inclut des pratiques de sécurité DevSecOps, des tests de pénétration réguliers, l'utilisation de services de gestion des identités et des accès (IAM), et le chiffrement des données en transit et au repos. Ces mesures garantissent la protection des données des utilisateurs et la conformité aux réglementations.

Architecture :

Avant d'identifier l'architecture, il faut rapidement comprendre le fonctionnement de la plateforme

Front end	Ember.js
Back end	Go
Communication	RTMP (Transfert des données) HLS (Transfert du stream pour les utilisateurs)
Recherche et Recommandation	API + Machine Learning
Applications mobiles	Java/Kotlin
Application sur consoles	Python, C++, Typescript

Etant donné que Twitch doit traiter des vidéos, des chats et des notifications en temps réel, de manière fluide et interactive, cela veut dire que l'architecture de Twitch doit répondre aux critères suivants :

- Scalabilité : Twitch doit gérer un grand nombre de flux vidéo simultanés et des millions d'interactions en temps réel, comme le chat et les notifications. L'architecture doit répondre efficacement aux pics de trafic.
- Résilience et tolérance aux pannes : Twitch doit être hautement disponible et résilient aux pannes, car toute interruption affecte directement l'expérience utilisateur. La défaillance d'une fonctionnalité ne doit pas affecter les autres.
- Déploiement continu : Twitch évolue constamment avec de nouvelles fonctionnalités et améliorations. Le développement des mises à jour doit être simplifié au mieux.

- Flexibilité technologique : Les besoins de Twitch évoluent et nécessitent parfois l'utilisation de différentes technologies pour différents composants. Twitch doit être développé de manière à proposer ses services adéquatement.
- Gestion de la complexité : Twitch est une plateforme complexe avec de nombreuses fonctionnalités variées, comme le streaming, le chat, la gestion des utilisateurs et la publicité. La plateforme doit pouvoir proposer ces fonctionnalités en même temps.
- Adaptation aux besoins des utilisateurs : Twitch doit offrir une expérience utilisateur fluide avec des temps de réponse rapides.

L'architecture de Twitch doit faire preuve de flexibilité et d'un haut débit pour répondre aux demandes des utilisateurs, une architecture intéressante à noter est le SOA (Architecture Orientée Service), une architecture séparée en différents services, qui se partagent des ressources, qui offre des avantages tel que :

- L'architecture est adaptée pour les grandes entreprises
- La communication est sécurisée et fiable (usage de SOAP)
- Développement simplifié à l'aide des ressources partagées

Le problème est que la réutilisation de service entraine de grandes dépendances ce qui peut entrainer des défaillances et des difficultés de développement.
Le choix le plus adapté est donc l'architecture microservice, une architecture séparant en services complètement indépendants les uns des autres, pour les raisons suivantes :

- Diviser des fonctionnalités de l'application de façon indépendante permet une gestion optimale des données et de mieux s'adapter aux besoins de chaque service.
- Communication rapide (REST)
- Les pannes d'un service perturberont un service et pas l'ensemble des services.
- Les déploiements fréquents et indépendants des autres services est possible.

Twitch est une plateforme qui priorise avant tout la fluidité de l'interaction entre les utilisateurs, c'est pour cette raison que l'architecture microservices est préférable, elle garantit la scalabilité (intégrité), la fluidité de la communication et la tolérance envers les pannes.

Maintenant que nous connaissons l'architecture, nous pouvons mieux comprendre le fonctionnement de la plateforme :

Front end	Ember.js
Back end	Go
Communication	RTMP (Transfert des données) HLS (Transfert du stream pour les utilisateurs)
Recherche et Recommandation	API + Machine Learning

Applications mobiles	Java/Kotlin
Application sur consoles	Python, C++, Typescript

Conclusion :

L'architecture n'est une chose à choisir sans y avoir réfléchit, il faut d'abord savoir ce que l'on a l'intention de faire avec le logiciel.
Quelle est son objectif, quelles sont ses fonctionnalités, à quelle fréquence l'améliorer, quelles sont les caractéristiques à prioriser sont des questions à se poser avant de prendre une décision.

Avec Twitch, la priorité est avant tout l'expérience utilisateur qui nécessite la fluidité et la résilience aux pannes.

Ce n'est qu'un exemple parmi tant d'autre qui permet de comprendre l'importance de cette étape dans la réalisation d'un projet.