Boge
Julien

Cahier des charges

Lecteur de musique Android

Le but était de réaliser un lecteur de streaming sur Android, basé sur sur une architecture distribuée et commandé par la parole. Les différents modules sont :

- un client Android
- un module de reconnaissance vocale
- un analyseur de commandes
- un serveur de streaming

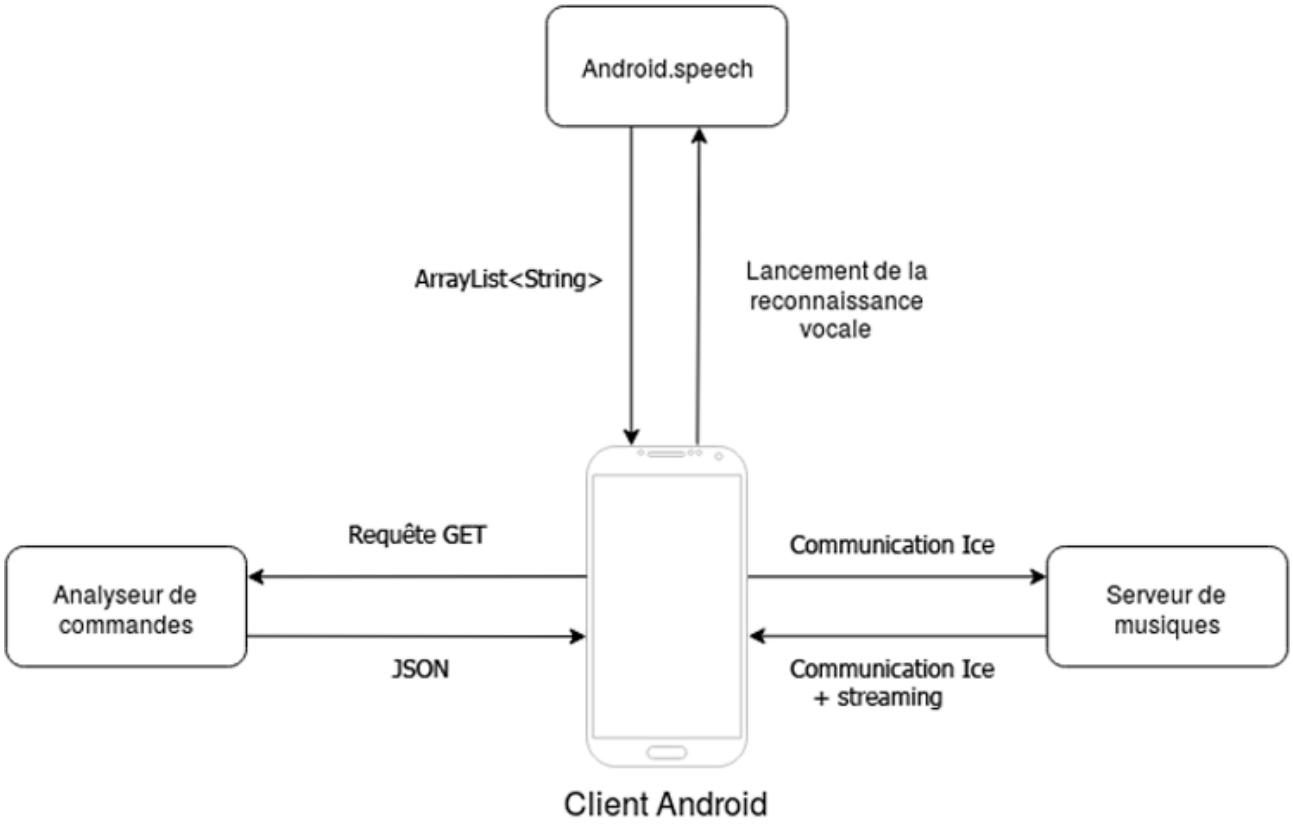

Illustration 1: Schéma de l'architecture de l'application

Client Android

L'application Android a pour rôle d'exécuter les actions de l'utilisateur et sert d'intermédiaire entre les différents modules. Il dispose d'un lecteur de musique pouvant lire un fichier audio diffusé en streaming sur un serveur distant.
L'ordre des actions à effectuer est :
- démarrer le service de reconnaissance vocale et récupérer la transcription
- envoyer cette transcription à l'analyseur, qui détermine la commande à exécuter
- exécuter cette commande, généralement en interaction avec le serveur de streaming et le lecteur.

Reconnaissance vocale

Il s'agit du module de reconnaissance vocale intégré à Android (Android.Speech). Lorsqu'il est appelé, il enregistre les paroles de l'utilisateur, en effectue la transcription et renvoie un tableau de chaînes de caractères correspondant à différentes transcriptions possibles.

Analyseur de commandes

Le client Android communique à distance avec le module d'analyse des commandes vocales.
Le module d'analyse repose sur Flask, une librairie Python permettant de créer un serveur.
Il prend en paramètre la chaîne de caractères correspondant à la commande vocale, et renvoie des données formatées en JSON indiquant la commande à exécuter ainsi que les paramètres qui l'accompagnent.
Le client Android quant à lui communique avec ce serveur à l'aide de la libraire Volley, qui permet d'exécuter des requêtes GET, POST, DELETE, PUT, … En l'occurence, on utilise seulement GET.
Le client envoie donc la transcription de la commande vocale à l'analyseur, et reçoit un objet JSON dont il extraie les informations pour déterminer la méthode à exécuter.

Exemple de commandes vocales prononcées par l'utilisateur :

- « Écouter X », « Jouer X »
- « Pause »
- « Resume », « Reprendre »
- « Stop »
- « Chercher X », « Search X »

Exemple d'objets JSON renvoyés par l'analyseur :

- {'command' : 'play', 'params' : 'X'}
- {'command' : 'pause', 'params' : ''}
- {'command' : 'resume', 'params' : ''}
- {'command' : 'stop', 'params' : ''}
- {'command' : 'search', 'params' : 'X'}
- Dans le cas où la commande n'a pas été reconnue, la commande est vide : {'command' : '', 'params' : ''}

Chacune de ces commandes correspond à une méthode de l'application cliente.

Serveur de streaming

Le client et le serveur communiquent à l’aide de Ice. Le client demande à réaliser des actions : démarrer/mettre en pause/reprendre/arrêter un stream, chercher un morceau de musique, récupérer la liste complète des musiques. Le serveur peut diffuser un stream à l’aide de libvlc.