DAOU Ahmed Amine

M1-ILSEN CLASSIQUE

Rapport final

Middlewares

Sommaire

Introduction :

Le Travail demandé dans le cadre de l'UCE Middlewares est de réaliser une application distribuée en utilisant les techniques vus dans l'UCE, tels que ICE,et une application cliente Android.

Le but principal de cette application est de lire des morceaux de musique stockés dans un serveur distant, exécuter des fonctions distantes et exploiter leurs résultats du coté client Android.

I. Description générale de la solution :

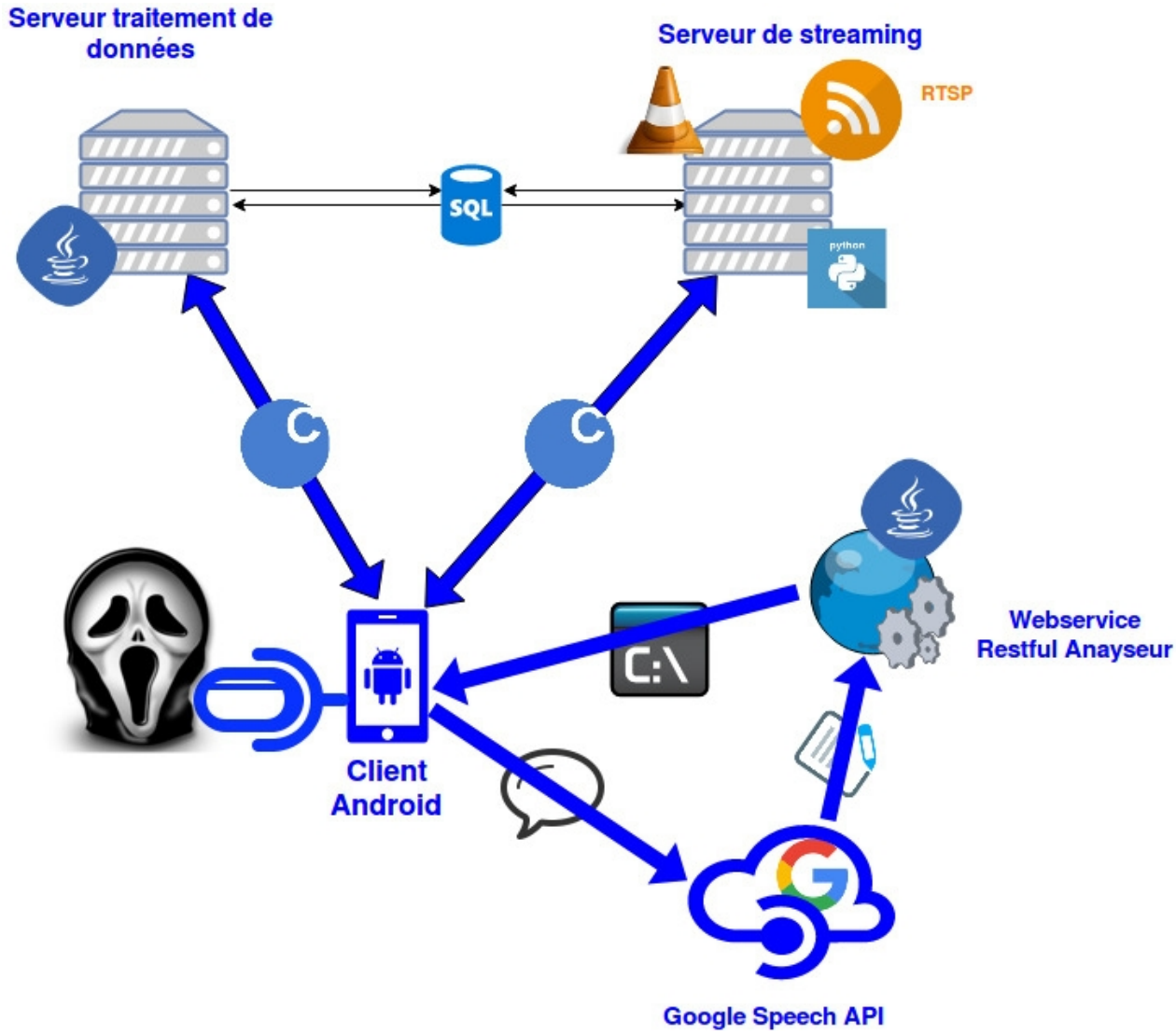

Illustration 1: Description de l'architecture de la solution

La solution est composée de quatre parties

I.1. Client Android :

Application Android qui permet à l'utilisateur de faire des recherches de musiques par nom, par Album ou par artiste en faisant appel au serveur de traitement de données, ou grâce au serveur de streaming, de lancer un flux RTSP de streaming, l'arrêter et le lire.
l'utilisateur peut choisir d'utiliser l'application avec les boutons ou avec les commandes vocales ou avec les deux ensemble.

I.2. Serveur de traitement de données

Un serveur Java permettant de rependre aux requêtes d'affichage, d'insertion ou de mise à jour de la base de données SQL contenant toutes les informations des musiques dans le disque.

I.3. Serveur de streaming

Un serveur Python permettant de gérer le flux RTSP , le lancer ou l'arrêter après les demandes du client.

I.4. Service Web Restful Analyseur de requêtes

Un service web codé en java qui traite les commandes vocales du client et revoie un résultat correspondant à une commande qui sera exécutée par la suite.

II. Bibliothèques Utilisées :

Sqlite-jdbc-3.21.0 :

Utilisée pour la gestion de la base de données musique simple à utiliser pour créer la base, les tables, ajouter des éléments dans les tables, les mettre à jour et les récupérer.

mp3agic-0.8.1:

Librairie java codée par Michael Patricios parmi les fonctionnalités proposées j'ai utilisé celle qui récupère les photos(Cover & Artwork) qui sont stockées dans les fichiers Mp3 afin de pouvoir les afficher dans l'application cliente Android après le choix de la chanson.

ice-3.7.0:

Librairie publiée par ZeroC, permet d'assurer la communication client serveur dans une architecture distribuée.

JAX-RS :

Java API for RESTful Web Services est une interface de programmation Java permettant de créer des services Web avec une architecture REST.

LibVLC (python & Android) :

c’est un Framework multimédia intégré dans cette application pour obtenir des capacités multimédias. Elle fournit toutes les fonctionnalités qu’on peut trouver dans le logiciel VLC média Player.

III. Base de Données :

La base de données est composée d’une seule table MUSIC qui comporte 7 colonnes

ID : Identifiant unique de la chanson.

NAME : Nom de la chanson.

ALBUM: Nom de l’album duquel la musique fait partie.

GENRE :genre de la musique.

URL:chemin de la musique sur le disque.

MINUTES, **SECONDS** et un entier **FAV** qui défini si la musique est favorite(1) ou pas (0).

ID	NAME	AUTHOR	ALBUM	GENRE	URL	MINUTES	SECONDS	FAV
173YOJe_...	Treasure	Bruno Mars	Unorthodox Jukebox	Pop	/home/paolo/Bureau/client2/database/musics/Bruno Mars - Treasure....	3	12	0
1XVCPduN...	Cara Italia	Ghali	Album	Hip-hop/Rap	/home/paolo/Bureau/client2/database/musics/GHALI - Cara Italia.mp3	5	14	0
1sQerSDR...	Lacrime	Ghali	Album	Hip-hop/Rap	/home/paolo/Bureau/client2/database/musics/GHALI - Lacrime.mp3	3	48	1
1Gr-Jq-eVT...	Oggi No	Ghali	Album	Hip-hop/Rap	/home/paolo/Bureau/client2/database/musics/GHALI - Oggi No.mp3	2	25	0
1LACuMkR...	Wily Wily	Ghali	Album	Hip-hop/Rap	/home/paolo/Bureau/client2/database/musics/Ghali - Wily Wily.mp3	4	43	1
1Y1tsHSjH...	Vida	Ghali	Album	Hip-hop/Rap	/home/paolo/Bureau/client2/database/musics/GHALI - Vida (Prod. Ch...	3	12	0
18rTw7Y-7...	Bella	Maitre Gims	Subliminal	Hip-hop/Rap	/home/paolo/Bureau/client2/database/musics/Maitre Gims-Bella.mp3	4	43	1
1HrccK412...	Mon petit loup	Sofiane	Affranchis	Hip-hop/Rap	/home/paolo/Bureau/client2/database/musics/Sofiane - Mon petit loup....	3	15	0

Illustration 2: Aperçu de la base de données music.db sur sqliteonline.com

IV. fonctionnement du Système

IV.1. Coté Client :

La première activité permet de faire des recherches sur les chansons par titre, par artiste et par album, le résultat est affiché dans un tableau trié par critère de recherche.

Si le champ n'est pas rempli toutes les chansons seront affichées.

dès le lancement de la recherche l'exécution de la requête se fait côté serveur de données, le résultat est affiché dans un Table Layout, en cliquant deux fois sur le nom de la chanson on relance la première activité avec le couver de la chanson et c'est là qu'on peut ;

-jouer la musique/pause/stop/baisser ou augmenter le volume/ajouter ou retirer la chanson des favoris ou éffectuer d'autres recherches.

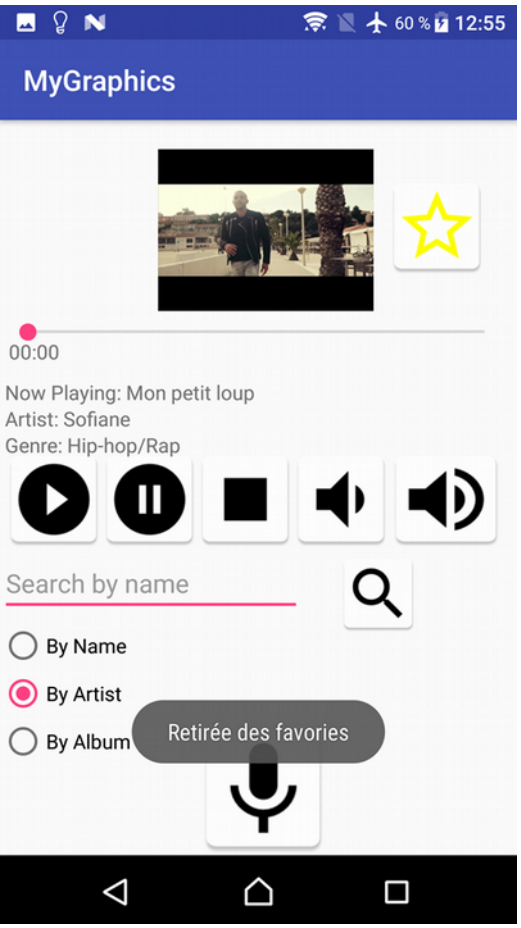

Illustration 3: Retirer une chanson des favoris

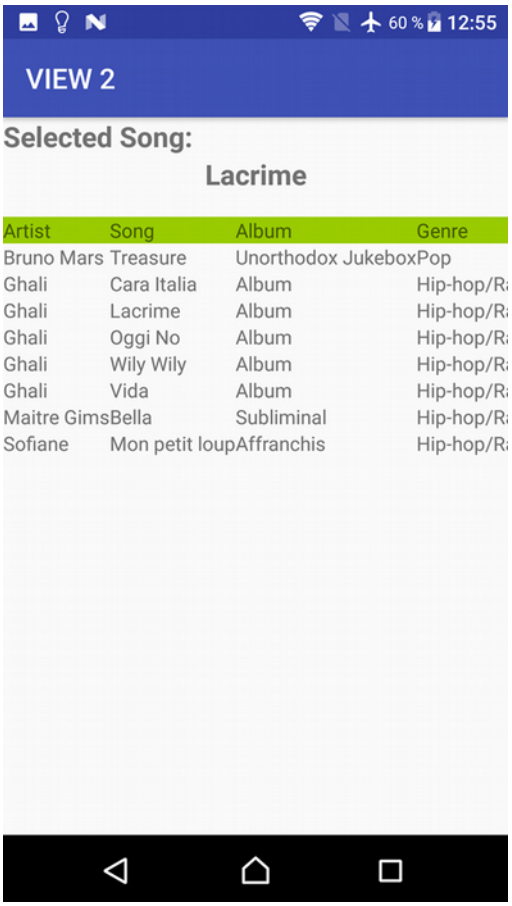

Illustration 4: Résultat d'une Recherche

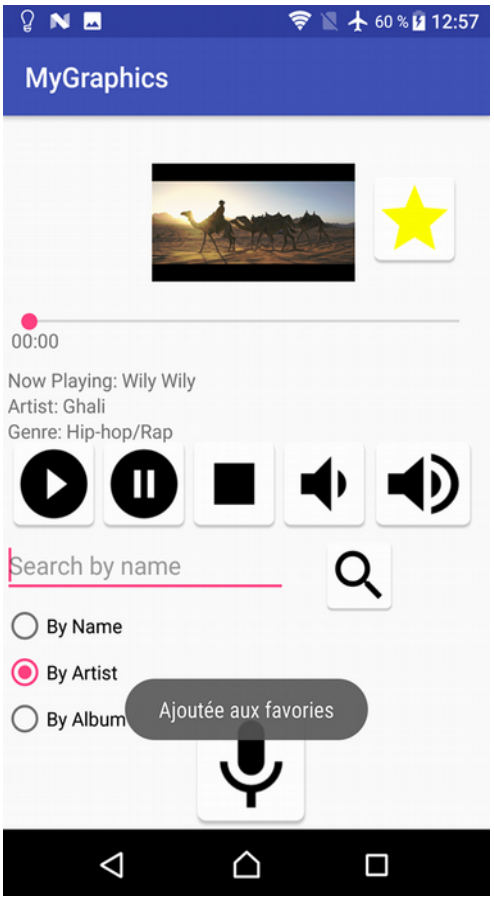

Illustration 5: Ajout d'une chanson aux favoris

IV.2. Coté Serveurs :

le serveur de données se lance sur le port 10000. Le résultat des requêtes sur la base de données est de type musique ou tableau de musique, ce serveur s'occupe aussi de récupérer le « couver » de la musique qui est envoyé au client sous forme de tableau d'octet (sequence<byte> dans l'interface ICE) qui sera converti à une bitmap avant de l'afficher.

Le serveur de streaming se lance sur le port 1234. il reçoit le chemin de la musique sur le disque avec lequel il crée un flux RTSP sur le port 8555, le passage à une autre musique nécessite d'arrêter le flux par une commande client pour libérer le port

IV.3. Coté Web Service :

Après la réception d'une commande vocale le web service est appelé d'une façon asynchrone, la commande doit Background de la classe AsyncTask d'Android crée une requête http du type GET en passant la parole en paramètre à l'adresse 'http://Ip adresse :8080/ws/analyse/{parole}, le résultat envoyé au client correspond à une commande qui sera directement exécutée

V. Commandes vocales :

Les commandes vocales que l'utilisateur peut utiliser :

- « Rechercher par nom $nom_de_la musique »
- « Rechercher par artiste $nom_de_l'artiste »
- « Recherche par album $nom_de_l'album »
- « Augmenter le volume/son »
- « Baisser le volume/son »
- « Pause »
- « Play/jouer/reprend la musique »
- « Stop/arrêter la musique »
- « Ajouter/Retirer-Supprimer des favoris »

l'ajout de certaines expression tel que « ….s'il vous plaît » ou « merci de …. » est accepté tant que le minimum de termes existe pour distinguer quelle commande exécuter.

VI. Répertoires et lancement de l’application :

VI.1. Répertoires :

/: PrinterI.java l’implémentation des fonction du serveur de données

/server: Server.java

/generated:classes java générés a partir de l’ interface printer.ice du serveur de données

/classes: les .class java coté serveur de données

/interfaces : interfaces ICE printer.ice et streamer.ice contenant les définition des modules les objets a gérer et les fonctions implémentés cotés serveurs.

/lib : les jars utilisés coté serveur de données

/streamer:répertoire contenant le code source du serveur de streaming

/player:classes pyhton et classes java générées depuis l’interface streamer.ice et utilisés dans l’application cliente Android

/web:repertoire du webservice contenant la web archive , les classes et le descripteur du déploiement web.xml

/database:classes de test de la base de données ,création de la base , la table , les opération de l’insertion et l’Update .

/clientandroid:code source de l’application Android

VI.2. Lancement de l’application :

-Lancement du serveur de données :

commande : bash dataServer.sh , ce message s’affiche "Data Server running on :10000"

-Lancement du serveur de streaming :

commande : bash streamerServer.sh , ce message s’affiche "Streamer Running on 1234"

-Déploiement du service Web:

Apres avoir demmaré votre serveur d’application « Glassfish »

commande :bash webservice.sh.

-Lancement du Client:

Apres avoir mis l’adresse ip du serveur

installer l’apk sur le smartphone Android.

VII. Conclusion :

Mettre en pratique les techniques qu’on a vus au cours des Tp sont très intéressants surtout dans le contexte de cette application, ça permet d’avoir une idée sur l’importance des architectures distribuées et leurs utilités, ainsi que le principe de streaming via le protocole RTSP.

VIII.Sources

Comment créer une base de données avec SQLite – Java

Comment récuperer et lire un flux RTMP/RTSP en live sur android

mp3agic - la bibliothèque utilisée pour recupérer le cover d'un fichier mp3

Documentation de libVLC

Documentation ZeroC ICE

AsyncTask - Utilisée pour l'appel du service Web