université de BORDEAUX

Master 1 Informatique

Projet de Programmation
Synchronisation audio-video pour expériences de perception

Auteurs :
Alexandre Kervadec
Thibaut Fabre
Guillaume Verdugo
Jeremy Arrestier

Client :
Jean-Luc Rouas
Chargé de TD :
Jeremy Frey

27 mars 2015

Remerciements

Nous tenons tout particulièrement à remercier Jeremy Frey, notre chargé de TD, qui n'a pas hésité à prendre sur son temps pour nous aider lorsque l'on avait des difficultés. Nous le remercions aussi de nous avoir suivi pendant toute la durée de ce projet. Nous remercions aussi Jean-Luc Rouas, qui nous a offert l'opportunité de participer à son preojet, et ainsi mettre nos connaissances au service de la recherche.

Résumé

"Dans le cadre de collaborations avec le laboratoire CLLE-ERRSàB de l'université Bordeaux Montaigne et l'université de Waseda à Tokyo (Japon), nous (le Labri ndlr) étudions les situations de communication intra et inter culturelles, en considérant les informations sonores et visuelles. Nous avons d'ores et déjà constitué un corpus audio-vidéo multilingue (français, japonais, anglais américain et portugais du brésil) pour un ensemble de situtations prédéfinies (admiration, séduction, arrogance, doute, irritation, évidence, politesse, surprise, ...)." — Jean-Luc Rouas

Ce projet vise à étudier comment un utilisateur peut construire à partir d'un son (exprimant par exemple l'admiration) et d'une vidéo (exprimant par exemple le mépris) un document audio-vidéo exprimant par exemple l'ironie. L'utilisateur peut être amené à utiliser les données enregistrées dans sa langue natale ou non, dans le but d'étudier les correspondances entre les langues.

Notre but est donc de réaliser une interface graphique permettant de réaliser des tests prosodiques [1] sur des cobayes. Ces sujets auront à faire un choix, d'une vidéo parmi plusieurs, à fusionner avec une bande son parmi une autre liste. Ce mixe de vidéo/son devra donner une réponse à une question du genre : "Réaliser une vidéo qui exprime l'ironie."

1. Prosodie : "La prosodie (ou la prosodologie) est une branche de la linguistique consacrée à la description (aspect phonétique) et à la représentation formelle (aspect phonologique) des éléments de l'expression orale tels que les accents, les tons, l'intonation et la quantité, dont la manifestation concrète dans la production de la parole, est associée aux variations de la fréquence fondamentale (F0), de la durée et de l'intensité (paramètres prosodiques physiques). Ces variations étant perçues par l'auditeur comme des changements de hauteur (ou de mélodie), de longueur et de sonie (paramètres prosodiques subjectifs)".[6]

Table des matières

Chapitre 1

Analyse de l'existant

1.1 Compréhension du sujet

Afin de comprendre au mieux la prosodie, l'article [4] nous permet de mettre une définition sur ce terme technique.

L'article [3] nous donne des notions sur les émotions que l'on peut passer avec la voix, alors que le livre [7] nous aide sur les expressions faciales.

Enfin, le document [8] nous offre une approche sociétale du problème avec une étude sur la prosodie attitudinale pour la langue japonaise.

1.2 Références sur la transformation de la voix

Le cours de Ricardo Gutierrez-Osuna [10] et l'article de l'IEEE Signal Processing Letters [11] sont des descriptions physiques de la modulation et modification de la voix.

L'extrait de la IEEE International Conference on Acoustique de 1998 [1] décrit deux possibles modifications prosodiques.

Le document de Véronique Aubergé [2] nous offre une approche par la méthode Gestalt de la prosodie.

Dans cet article du journal Voice transformation using PSOLA technique [20], un système de conversion de voix utilisant PSOLA et un module pour les transformations spectrales sont étudiés.

1.3 Références sur l'expression faciale

Le document [5] expose une approche sur la définition de règles de synchronisation des expressions faciales et particulièrement buccales par rapport à un certain discours.

Le document [16] est utile dans le sens qu'il porte sur l'étude de la prosodie faciale, et ce afin de pouvoir détecter l'ethnie du protagoniste, ce qui nous est utile car le projet porte sur trois langues : le français, l'anglais (américain) et le japonais.

1.4 État de l'art

L'article [17] nous donne une ligne directrice pour tout ce qui touche la reconnaissance des différents dialectes arabes. En d'autres termes, malgré une proximité géographique et linguistique, la prosodie permet la différenciation. L'article de la revue en ligne Alsic [15] nous apporte un point sur l'état actuel de l'art en matière de logiciel pour l'apprentissage de la prosodie, ce qui est proche de ce que l'on cherche à développer.

1.5 Programmation avec du contenu vidéo

Les articles [9] et [12] nous ont permis de mieux comprendre le concept d'encodage des vidéos et l'utilisation de *Codecs* pour pouvoir décoder ces vidéos afin de les lire.

1.6 Ergonomie

L'article [14] est une étude sur la "collaboration entre Ergonomie, Design et Ingénierie", qui nous offre une façon de procéder afin de réaliser une ergonomie des plus efficace.

1.7 Base de données

Durant ce projet, il a fallu trouver le système de base de données le plus adapté à nos besoins. Pour ce faire, nous nous sommes appuyés sur l'article [19], disséquant l'architecture *NoSQL* (Not Only SQL). Nous tournant finalement vers le *SGBDR* (Système de Gestion de Base de Données Relationnelles) *SQLite*, nous nous sommes informés sur l'utilisation et les finalités de ce logiciel avec le livre [13].

Chapitre 2

Analyse des besoins

2.1 Fonctionnement de l'application

La FIGURE 2.1 décrit le fonctionnement de l'application, *i.e.* les différents états dans lesquels il est possible de se trouver durant l'utilisation de l'application.

La FIGURE 2.2 est en rapport avec le diagramme précédent (FIGURE 2.1). Il décrit les relations entre les différents états de transition.

2.2 Classement par priorité des besoins

Les besoins sont classés par priorité dans leur ordre d'apparition. De plus, chaque besoin se voit attribué un niveau de priorité comme suit :

- Priorité critique
- Priorité moyenne
- Priorité basse
- Facultatif

2.3 Besoins fonctionnels

2.3.1 Utiliser l'application sur les systèmes d'exploitation principaux et récents

2.3.1.1 Description

Etant donné que notre client sera amené à transporter l'application, il faut que celle-ci puisse fonctionner sur différents environnements à savoir *Microsoft Windows 7*, *Ubuntu 12.04*, *Debian 6.0*, *MAC OS X 10.9* et les versions plus récentes de ces systèmes d'exploitation. L'application est susceptible de fonctionner sur d'autres systèmes d'exploitation (par compatibilité de noyau) sans toutefois de garantie de ce fonctionnement.

2.3.1.2 Faisabilité

Cette condition sera remplie en utilisant une application web, supportée par tous les environnements possédant un navigateur web. Or cette qualité est remplie nativement par tous les systèmes d'exploitation cités précédemment.

2.3.1.3 Contingence

Le risque principal est que le client travaille sur une machine ne possédant pas de navigateur web. Pour pallier à cela, un navigateur web portable, comme par exemple *Mozilla Firefox Portable Edition*, sera installé sur le périphérique utilisé.

2.3.1.4 Test

Lancer l'application dans chacun des systèmes d'exploitation cités précédemment.

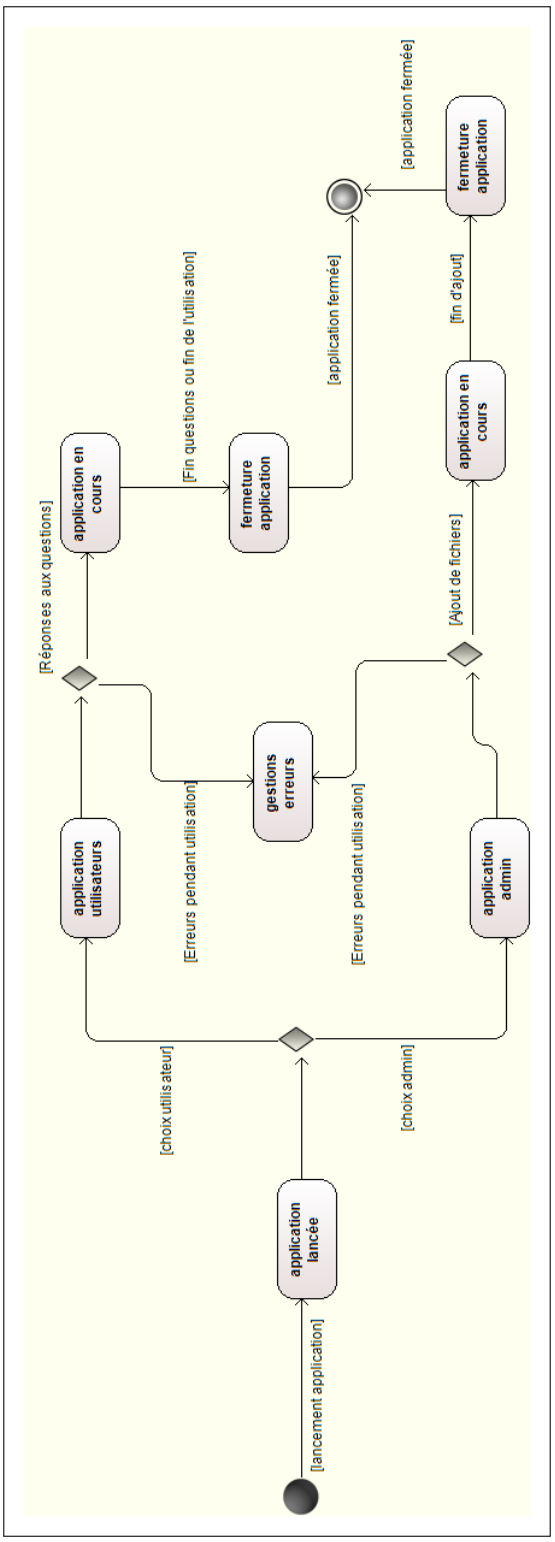

FIGURE 2.1 – Diagramme d'états - Fonctionnement de l'application

2.3.1.5 Priorité : *Critique*

2.3.2 Rendre l'application nomade

2.3.2.1 Description

Pour effectuer ses études, notre client ne souhaite pas transporter sa machine sur les lieux où elles se déroulent. Pour cela, il faut donc avoir une application transportable sur un périphérique externe de type clef USB.

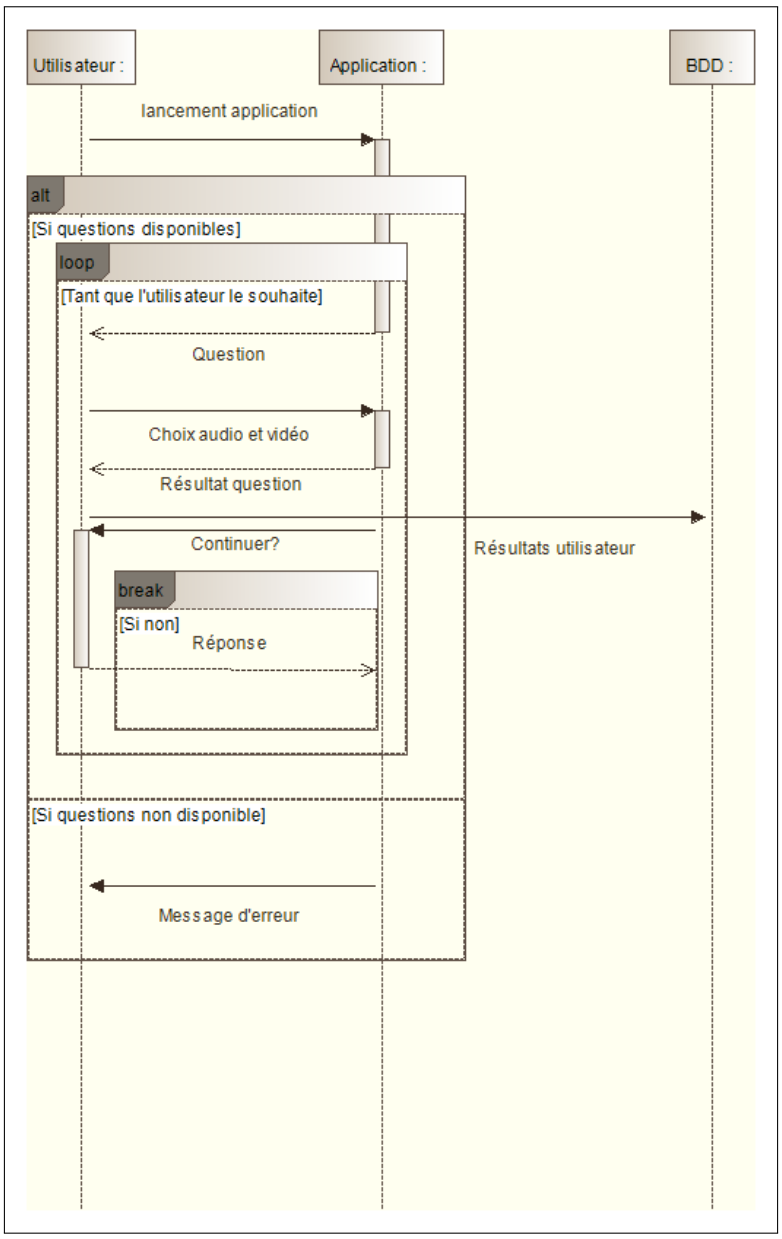

FIGURE 2.2 – Diagramme de séquences - Fonctionnement de l'application utilisateur mode Questions/Réponses

2.3.2.2 Faisabilité

Pour remplir cette condition, tous les composants de l'application seront stockés sur une clef USB ou un disque dur externe afin que le client puisse l'utiliser sur n'importe quel ordinateur.

2.3.2.3 Contingence

Le risque serait que le formatage de la partition de la clef USB ne soit pas pris en compte par le système d'exploitation de la machine (par exemple le formatage *NTFS* de *Windows* n'est pas reconnu par le système *Mac OS*, le formatage *EXT4* du système *Ubuntu* n'est pas reconnu par les systèmes d'exploitation *Windows*). Pour éviter cela, la clef devra être formatée en *FAT32* qui est un formatage de partition reconnu par les systèmes d'exploitation cités dans la section 2.3.1.

2.3.2.4 Test

Faire fonctionner l'application à partir d'une clef USB (ou disque dur externe selon le support qui sera choisi).

2.3.2.5 Priorité : *Critique*

2.3.3 Ajouter du contenu multimédia et des questions

2.3.3.1 Description

Le client doit pouvoir enregistrer dans la base de données, de nouvelles vidéos et de nouveaux sons afin d'augmenter l'efficience de ses tests. De plus, pour améliorer ses recherches, notre client pourra ajouter des questions avec leurs correspondances audios et vidéos.

2.3.3.2 Faisabilité

Pour faciliter cette gestion, nous utiliserons une base de données. Les données seront plus facilement accessibles lors de l'utilisation de l'application.

2.3.3.3 Test

Ajouter du contenu multimédia et une question puis lister tout le contenu de la base de données pour voir si l'ajout à été pris en compte.

2.3.3.4 Priorité : *Moyenne*

2.3.4 Exploiter différents formats de fichiers audio et vidéo

2.3.4.1 Description

Le contenu multimédia de notre client étant composé de différents formats vidéos et audios, l'application doit pouvoir assurer une lecture optimale.

Effectivement, c'est un obstacle que l'on rencontre dès que l'on commence à programmer dans le domaine de la vidéo et de l'audio (des références étudiant certaines de ces contraintes : [9] et [12]). Le fond du problème est l'utilisation de Codecs (vidéo et audio) pour lire les différents types de fichiers, qui sont encodés selon différentes normes.

Les formats que l'on doit pouvoir supporter sont les suivants :

Formats audio	**Formats vidéo**
mpeg2	*mp4 (H.264)*
aac	*mov*
wav	

2.3.4.2 Faisabilité

Une solution que nous avons trouvé est le lecteur de médias *VLC Media Player* [18].

Ce lecteur va nous permettre d'utiliser les formats vidéo et audio présentés précédemment. De plus, c'est un logiciel accessible (gratuit et sous licence open-source).

De surcroit, il existe une version portable de *VLC Media Player* permettant un transport optimal sur une clef USB ou un disque dur externe. Cette condition répond à notre besoin de transportabilité évoqué précédemment.

2.3.4.3 Contingence

Le risque d'utiliser ce genre de logiciel peut être l'impossibilité d'intégrer le lecteur dans une page web (solution choisie dans la section 2.3.1), si l'application doit s'ouvrir dans un navigateur. Si cette solution n'est pas possible, on pourra ouvrir un lecteur indépendamment de la page web (en *standalone*).

2.3.4.4 Test

Création d'un script qui ouvrira toutes les vidéos disponibles et vérifiera si une erreur est survenue.

2.3.4.5 Priorité : *Moyenne*

2.3.5 Récupérer les questions et résultats des tests

2.3.5.1 Description

L'application a pour but de répondre à une question donnée en fusionnant une vidéo et un audio.
Ces combinaisons sont importantes pour notre client, nous devons donc les récupérer.

2.3.5.2 Faisabilité

La solution à ce besoin serait une base de données, stockant l'ensemble des questions, sons, vidéos et réponses/résultats. L'application devra aussi exporter les données sous forme d'un fichier *txt*, fichier qu'exploitera le client.

2.3.5.3 Test

Faire faire le test à un faux sujet, et vérifier les données exportées dans le fichier *txt*.

2.3.5.4 Priorité : *Moyenne*

2.3.6 Rassembler des informations sur les sujets

2.3.6.1 Description

Afin de pouvoir exploiter les résultats des tests, le client veut récupérer des informations sur les sujets. Les informations que l'on doit récupérer sont les suivantes :

- nom
- prénom
- sexe
- date de naissance
- langue maternelle
- si la langue maternelle est différente de celle du test, nombre d'année d'études de cette langue

2.3.6.2 Faisabilité

Pour répondre à ce besoin, il faut implémenter un formulaire au lancement de l'application.

2.3.6.3 Test

Faire essayer l'application à une personne tierce, puis, vérifier que toutes les informations sur la personne ont bien été récupérées.

2.3.6.4 Priorité : *Faible*

2.4 Besoins non-fonctionnels

2.4.1 Permettre une maintenance du logiciel

2.4.1.1 Description

L'application que nous allons livrer ne sera pas une finalité mais seulement une étape dans un projet beaucoup plus vaste. Ainsi, d'autres personnes devront probablement modifier cette application afin de répondre à de nouveaux besoins. Ces nouveaux développeurs devront disposer de tous les éléments nécessaires pour comprendre notre travail.

2.4.1.2 Faisabilité

Il est possible de répondre à ce besoin en documentant notre code. Il existe par exemple en *Java*, la *Javadoc*, qui est générable avec des commentaires spéciaux. Elle est exportable en *PDF* et contient toutes les explications necessaires à la compréhension du code. Il existe aussi la *PHPDoc* qui est l'équivalent pour le *PHP*.

De plus, il faudra ajouter des commentaires quand une fonction sera trop complexe et que des indications intermédiaires seront nécessaires.

2.4.1.3 Test

Demander à un développeur externe au projet d'essayer de comprendre notre code.

2.4.1.4 Priorité : *moyenne*

2.4.2 Bénéficier d'une ergonomie efficiente

2.4.2.1 Description

L'application sera ergonomique pour permettre au sujet de se concentrer un maximum sur le test et pas sur le fonctionnement de ladite application. De plus, le sujet soumis au test sera possiblement novice en informatique, l'application devra ainsi être intuitive.

2.4.2.2 Faisabilité

Pour satisfaire ce besoin, nous devons mettre en place un graphisme épuré de tout accessoires détournant l'attention. De plus, nous utiliserons des couleurs de ton pastel pour éviter de fatiguer le regard de l'utilisateur.

Cette contrainte est satisfaisable en s'inspirant de nombreux designs qui ont déjà été développés et publiés sur le web. On s'appuiera notamment sur l'article [14] qui étudie la *"collaboration entre Ergonomie, Design et Ingénierie"*.

2.4.2.3 Test

Se servir d'un novice en informatique pour tester l'application.

2.4.2.4 Priorité : *basse*

2.5 Prototype d'application

En annexe, un ***prototype*** de l'application que nous allons développer est présenté. Le rendu final pourra différer de ce prototype.

2.6 Gestion du temps

Nous avons établi un calendrier de projet qui présente le déroulement général du projet.

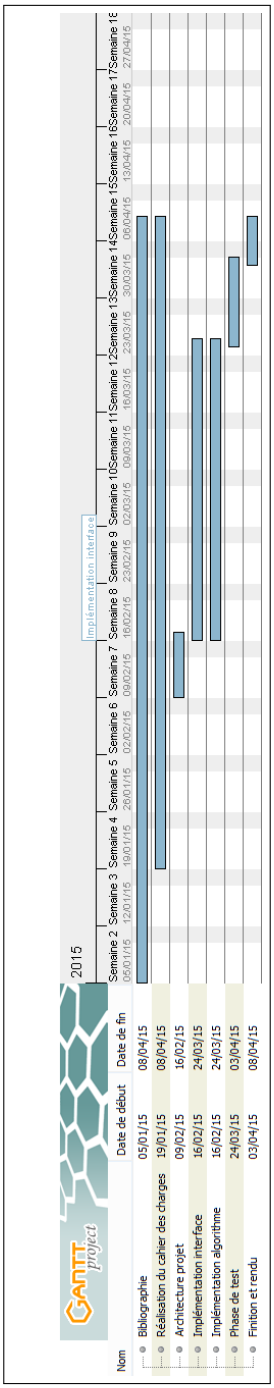

FIGURE 2.3 – Gantt - Calendrier du projet

Priorité	Nom	Raison
1	Tache 1	Doit être vérifié en premier car sinon [...]
2	Tache 2	On doit pouvoir [...]
3	Tache 3	Comme les principales fonctionnalités permettant de tester sont opérationnelles, nous pouvons passer à cette tâche.
4	Tache 4	Parce que [...]
5	Tache 5	La tache 5 fait partie des principales [...].
6	Tache 6	Dernière fonctionnalité essentielle à mettre en place.
7	Tache 7	Non-essentiel, mais apporterait un plus au projet.
8	Tache 8	Non-essentiel, mais apporterait un plus au projet.

FIGURE 2.4 – Tableau récapitulatif des tâches

Chapitre 3

Architecture

3.1 Explication détaillée des packages

Notre architecture est structuré en plusieurs grands groupes se composant de classes. Nous allons décrire ces groupes un à un.

3.1.1 BDD

Ce package contient l'ensemble des médias qui seront disponibles sur l'application (audio, vidéos, questions, langue de la question).

Cette organisation des données a été choisie pour optimiser l'ajout de médias et pour permettre l'ajout d'autres langues à l'application.

3.1.1.1 Classe DataBase

3.1.1.2 Classes Media, Audio et Video

3.1.1.3 Classe Question

3.1.1.4 Classe Language

3.1.2 Result

Ce package-ci est l'ensemble des informations concernant l'utilisateur (le sujet de l'expérience). Il est aussi composé de ses résultats aux réponses du test ainsi que d'une classe permettant d'extraire les données vers un fichier *CSV / txt / XML*.

3.1.3 Tests

Ce package regroupe l'ensemble des tests qui seront necessaires pour minimiser les erreurs au niveau de la base de données (par exemple lors d'un upload de médias, on vérifie que le format soit adapté).

3.1.4 Graphic

Ici sont regroupés toutes les fonctionnalités liées à l'interface graphique.

3.1.4.1 Classe UserGUI

3.1.4.2 Classe TestGUI

3.1.4.3 Classe ChooseGUI

3.1.4.4 Classe MediaManager

3.1.5 Processes

Nous avons réuni ici, tout ce qui correspond au *contrôleur* de l'application.

La classe **GUI_select** est la sélection de l'ensemble des médias présentées lors d'une question. **GUI_Answer** correspond à la vidéo et à l'audio sélectionnés par l'utilisateur liés à une question.

Cmd est l'utilisation de l'application par l'administrateur lorsqu'il voudra ajouter ou retirer des médias et questions de la base de données.

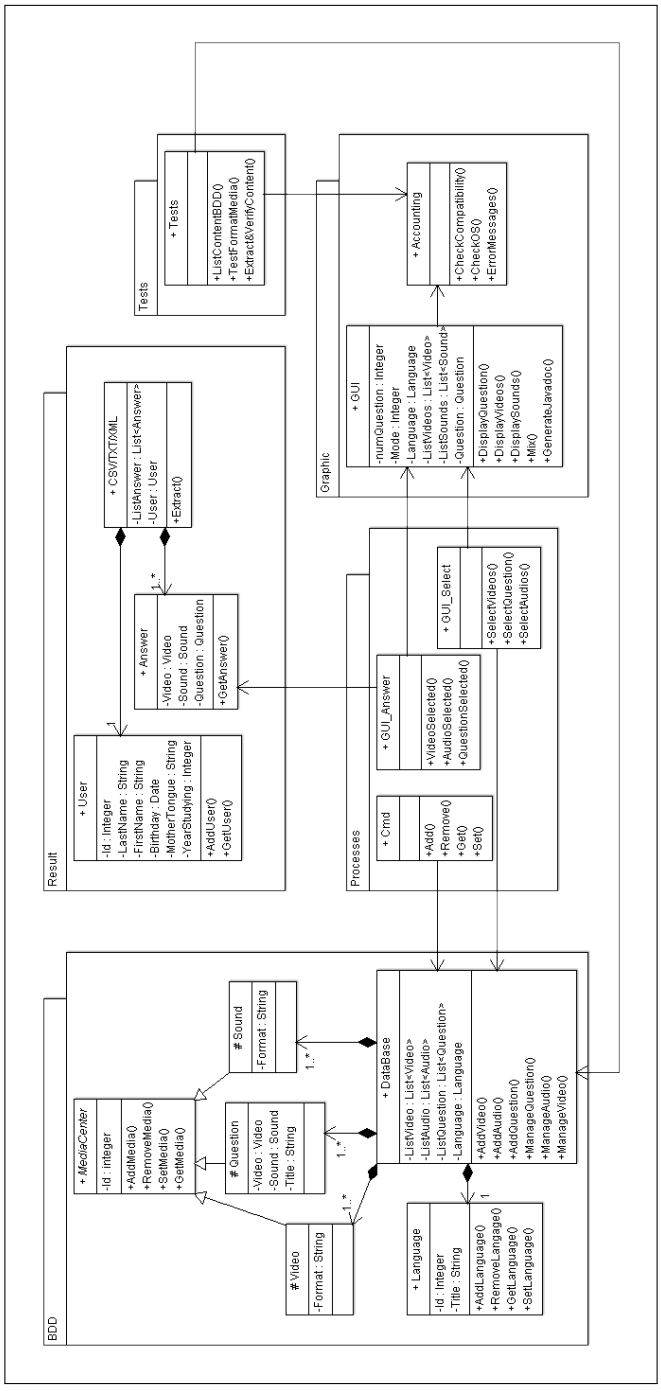

Figure 3.1 – UML - Architecture

Chapitre 4

Fonctionnement et tests

Dans cette partie nous cherchons à décrire dans un premier temps les fonctionnements, ou, le cas échéant, les non fonctionnements de notre application. Nous aborderons ensuite la politique de tests effectuée pour vérifier notre code.

4.1 Fonctionnement

4.1.1 Base de données

4.1.1.1 Représentation des données

4.1.2 Interface Graphique (GUI)

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

4.1.3 Contrôleur

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

4.1.4 Exportation des données

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui.

Chapitre 5

Bilan

Intro / Rappel Contexte
Nous avons donc pu en tirer la problématique suivante :

Problématique du sujet

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Annexes

Annexe 1

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Annexe 2

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Bibliographie

[1] A. Acero. Source-Filter Models for Time-Scale Pitch-Scale Modification of Speech. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing*, Seattle, WA (USA), May 1998.

[2] Véronique Aubergé. A gestalt morphology of prosody directed by functions : the example of a step by step model developed at icp. In *Speech Prosody 2002, International Conference*, 2002.

[3] Véronique Aubergé. Prosodie et émotion. *Actes des deuxiemes assises nationales du GdR I*, 3 :263–273, 2002.

[4] Jo-Anne Bachorowski. Vocal expression and perception of emotion. *Current directions in psychological science*, 8(2) :53–57, 1999.

[5] Jonas Beskow. Rule-based visual speech synthesis. In *EUROSPEECH*, 1995.

[6] Albert Di Cristo. Di cristo 2000. page 3, 2000.

[7] Paul Ekman and Wallace V Friesen. *Unmasking the face : A guide to recognizing emotions from facial clues*. Ishk, 2003.

[8] Dominique Fourer, Marine Guerry, Takaaki Shochi, Jean-Luc Rouas, Jean-Julien Aucouturier, and Albert Rilliard. Analyse prosodique des affects sociaux dans l'interaction face à face en japonais. In *XXXèmes Journées d'études sur la parole*, Le Mans, France, June 2014.

[9] Mohammed Ghanbari. Video coding : an introduction to standard codecs. Institution of Electrical Engineers, 1999.

[10] Ricardo Gutierrez-Osuna. Prosodic modification of speech. Introduction to Speech Processing.

[11] Jesper Haagen. Transformation and decomposition of the speech signal for coding. *IEEE signal processing letters*, 1(9) :136, 1994.

[12] Yun He, Jörn Ostermann, Marek Domanski, Oscar C Au, and Nam Ling. Introduction to the issue on video coding : Hevc and beyond. *Selected Topics in Signal Processing, IEEE Journal of*, 7(6) :931–933, 2013.

[13] Jay Kreibich. *Using SQLite*. " O'Reilly Media, Inc.", 2010.

[14] Chloé Lenté, Soizick Berthelot, and Stéphanie Buisine. Scénariser l'usage pour améliorer la collaboration entre ergonomie, design et ingénierie. 2014.

[15] Philippe Martin. Winpitch ltl, un logiciel multimédia d'enseignement de la prosodie. *Alsic [en ligne]*, 8(2), 2005.

[16] David Matsumoto. American-japanese cultural differences in the recognition of universal facial expressions. *Journal of cross-cultural psychology*, 23(1) :72–84, 1992.

[17] Jean-Luc Rouas, Melissa Barkat-Defradas, François Pellegrino, Rym Hamdi, et al. Identification automatique des parlers arabes par la prosodie. *Journées d'Etude sur la Parole*, 2006.

[18] VideoLan Streaming Solutions. Vlc media player, 2006.

[19] Christof Strauch, Ultra-Large Scale Sites, and Walter Kriha. Nosql databases.

[20] Hélène Valbret, Eric Moulines, and Jean-Pierre Tubach. Voice transformation using psola technique. *Speech Communication*, 11(2) :175–187, 1992.