Institut National des Sciences Appliquées de Rouen

Département ASI

Architecture des Systèmes d'Information

EC Informatique Répartie

Document de Spécifications

Projet

Messagerie instantanée et visio/audio-conférence

Auteurs

Gautier Darchen
Alexandre Huat
Marie-Andrée Jolibois
Romain Judic
Alexandre Le Lain

Version

v0.00

10 mai 2017

Table des matières

Chapitre 1

Introduction

L'application à développer est une plateforme de messagerie instantanée entre deux interlocuteurs. Elle permettra à ces interlocuteurs de communiquer tout en étant connectés sur des machines distantes.

1.1 Fonctions principales

Les fonctions principales de cette application peuvent être scindées en trois catégories :
— l'échange de messages ;
— le filtrage des messages en fonction de leur contenu ;
— l'utilisation d'un avatar.

Concernant la première fonctionnalité, les interlocuteurs pourront s'envoyer des messages écrits de façon instantanée. Ils auront en plus la possibilité de communiquer avec d'autres formats via un système de visio- ou audio-conférence intégré à l'application.

Le filtrage des messages prend en charge le contrôle parental et la modération. Les utilisateurs auront la possibilité de choisir d'activer ou non ce système de filtrage. De plus, ce dernier pourra être personnalisé.

Concernant la dernière fonctionnalité, un système d'avatar pourra être utilisé par les utilisateurs pour les conversations audio et écrites. Cette image n'est pas animée.

1.2 Utilisateurs

Il existe plusieurs types d'utilisateurs dont les droits diffèrent.

Utilisateur *lambda* : il peut créer (ouvrir) une conversation, participer à une conversation qu'il sélectionne. Il peut paramétrer ses filtres. À la connexion, il choisit aussi un login et un avatar.

Modérateur : il peut émettre des messages de modération et supprimer des messages d'utilisateurs.

Plusieurs utilisateurs peuvent discuter en même temps sur une même conversation.

Les seuls prérequis à l'utilisation sont l'installation et ouverture d'une application.

1.3 Contraintes

1.3.1 Contraintes matérielles

L'utilisateur doit disposer d'une machine reliée à internet.

Pour profiter du service visio, l'utilisateur doit posséder un matériel de capture vidéo (webcam) et d'entrées/sorties audio.

Pour utiliser le service audio, l'utilisateur doit posséder une entrée et une sortie audio.

La majorité des calculs est réalisée côté serveur, ce qui n'implique pas un besoin de ressources conséquentes côté client.

1.3.2 Contraintes logicielles

Le client doit disposer du Java Runtime Environnement 8.

Chapitre 2

Spécifications

2.1 Spécifications fonctionnelles

L'IA est basique et à l'image d'un helpbot. L'IA est donc présente sur chaque conversation. L'utilisateur peut entrer des mots-clefs dans la conversation pour avoir des indications sur l'utilisation de l'application.

Mots-clefs :

« **bonjour_sophisme** » **:** répond à l'utilisateur « Bonjour `<login>` ».
« **help** » **:** affiche un menu d'aide avec l'ensemble des mots-clefs.
« **help_profil** » **:** indique à l'utilisateur comment paramétrer son profil.
« **help_filtre** » **:** indique à l'utilisateur comment paramétrer les filtres.
« **help_chat1** » **:** indique à l'utilisateur comment ouvrir un salon de chat.
« **help_chat2** » **:** indique à l'utilisateur comment rejoindre une conversation.
« **help_video** » **:** indique à l'utilisateur comment démarrer une conversation vidéo.
« **help_audio** » **:** indique à l'utilisateur comment démarrer une conversation audio.

Les utilisateurs ont des comptes temporaires qui n'existent que durant la durée de leur connexion au service. A l'ouverture de l'application ce dernier est redirigé vers la page « Paramètres » où il précise différents champs le caractérisant (avatar, pseudonyme, mots à filtrer...).

Par défaut, lorsqu'un utilisateur démarre une conversation, il est seul avec l'IA jusqu'à l'arrivée d'autres utilisateurs.

Cas d'utilisation

Cette section présente les cas d'utilisation suivants :
— discuter ;
— paramétrer le filtrage ;
— filtrer les messages ;
— se connecter.

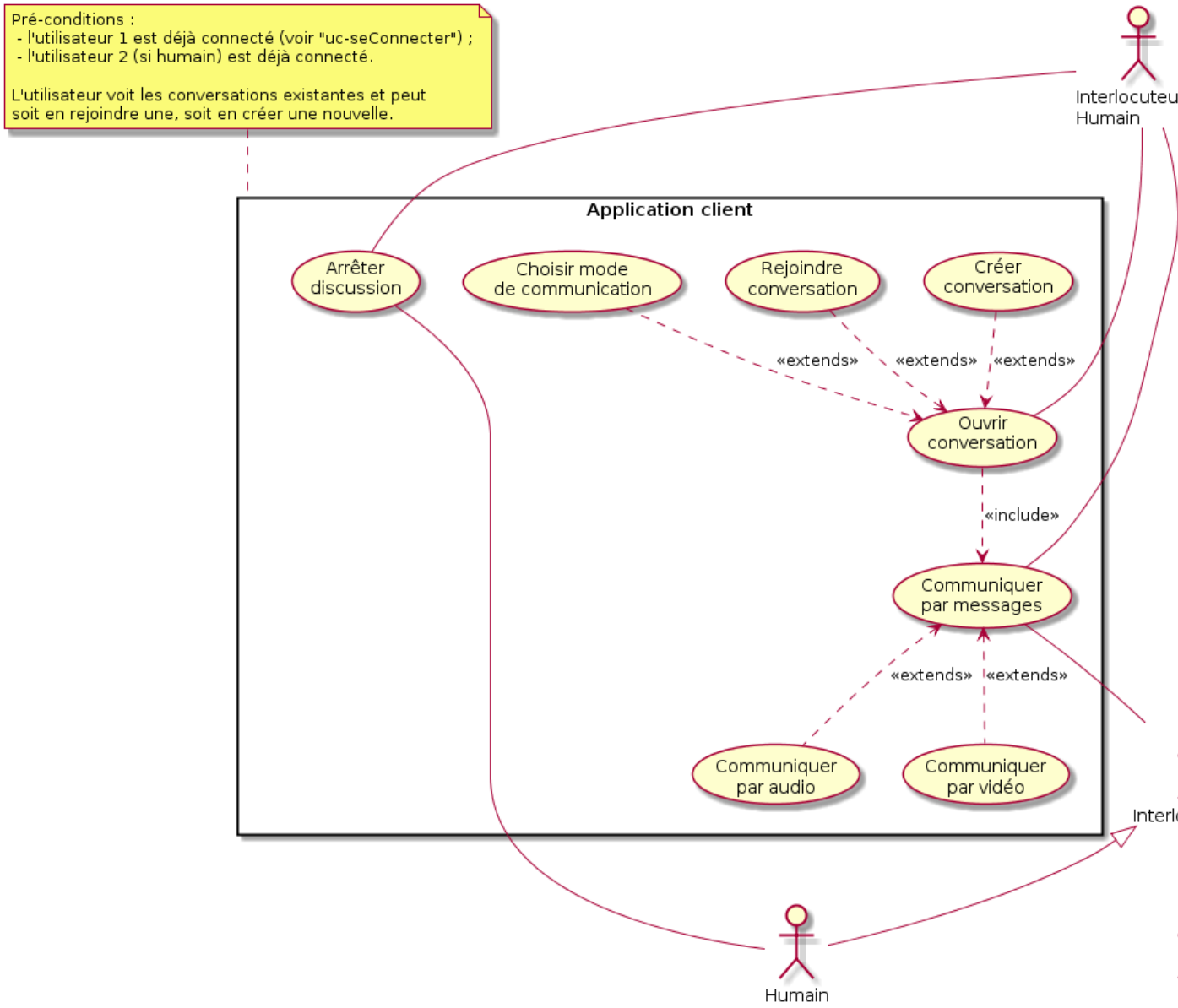

FIGURE 2.1 – Cas d'utilisation **discuter**

Le cas d'utilisation (figure 2.1) représente les interactions entre les différents interlocuteurs et les différents moyens mis à leur disposition pour communiquer.

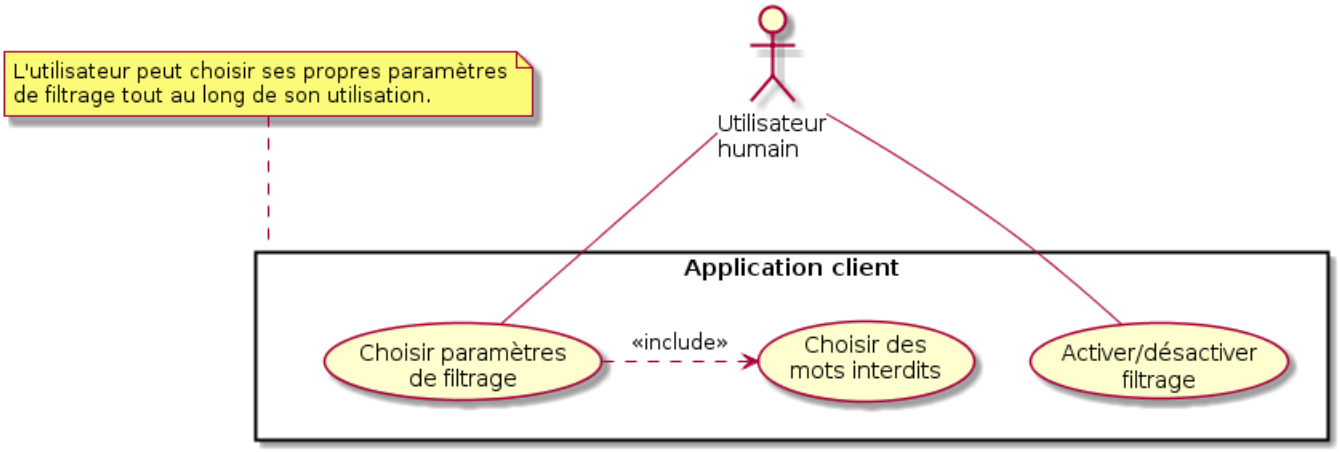

FIGURE 2.3 – Cas d'utilisation **paramétrer le filtrage**

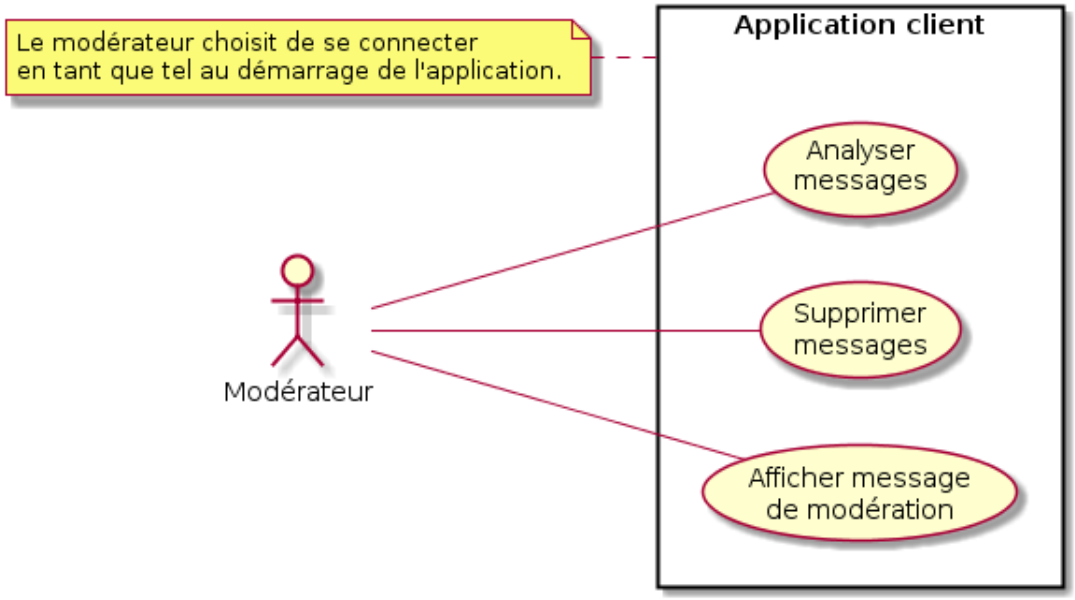

FIGURE 2.2 – Cas d'utilisation **filtrer les messages**

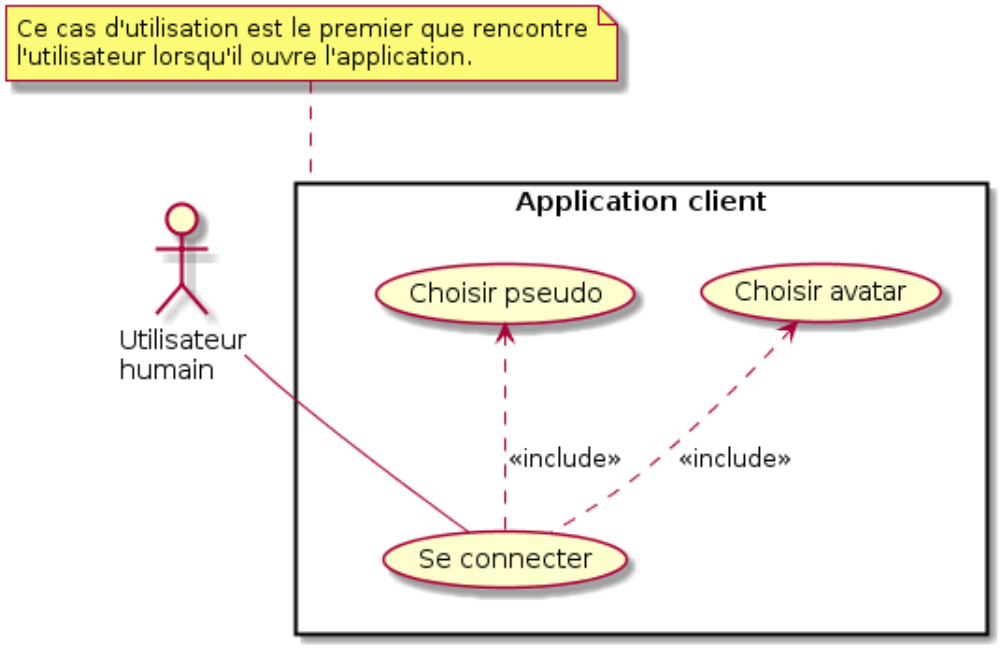

FIGURE 2.4 – Cas d'utilisation **se connecter**

2.2 Spécifications d'interfaces

Maquettes

Cette section présente les maquettes de l'application.

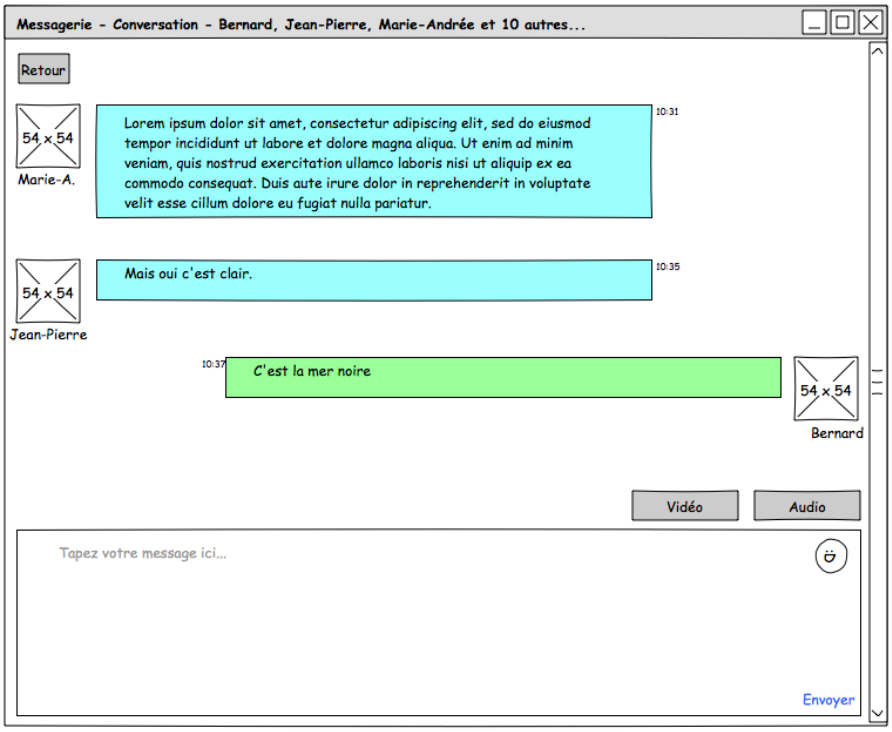

FIGURE 2.5 – Un exemple de conversation texte

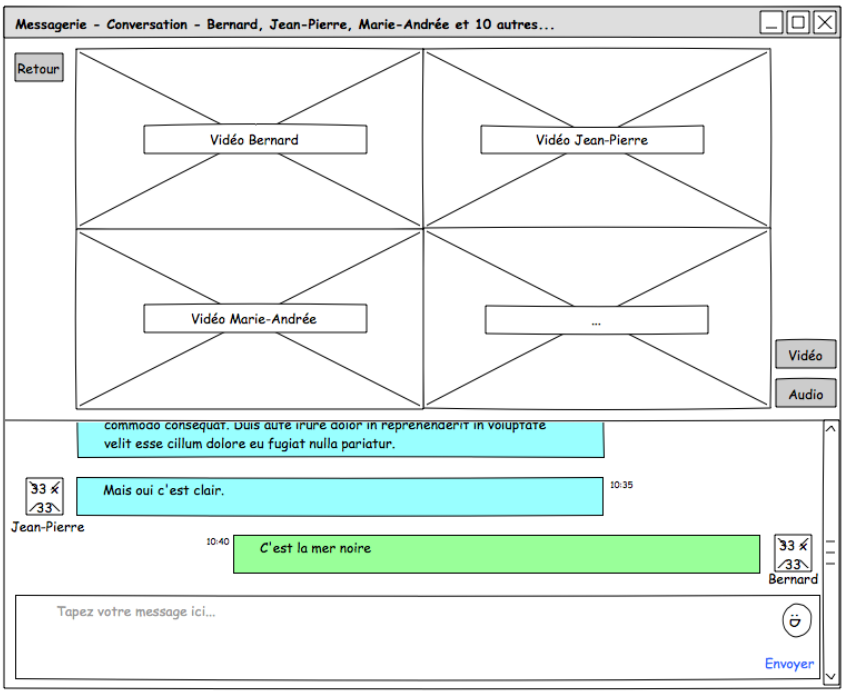

FIGURE 2.6 – Conversation vidéo

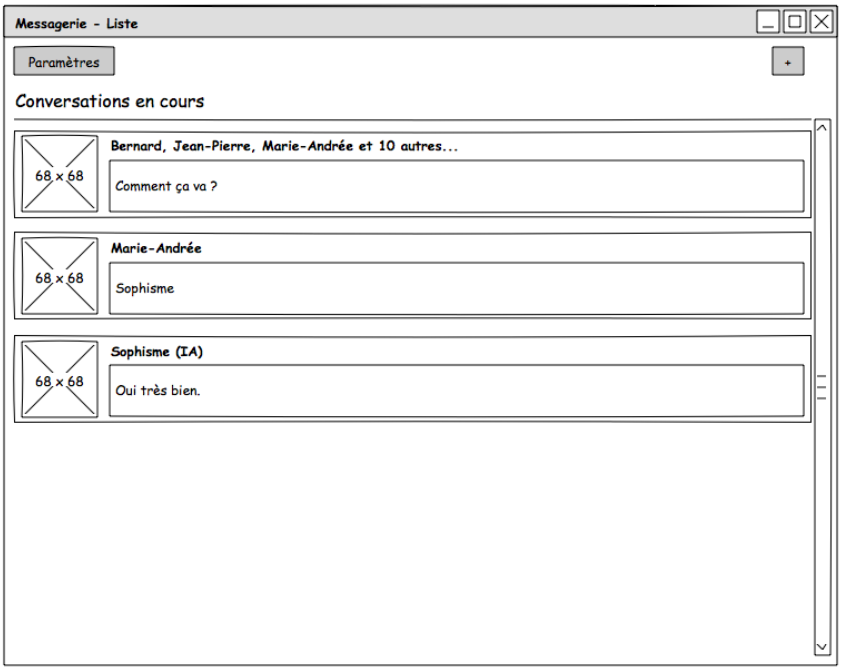

FIGURE 2.7 – Liste des conversations

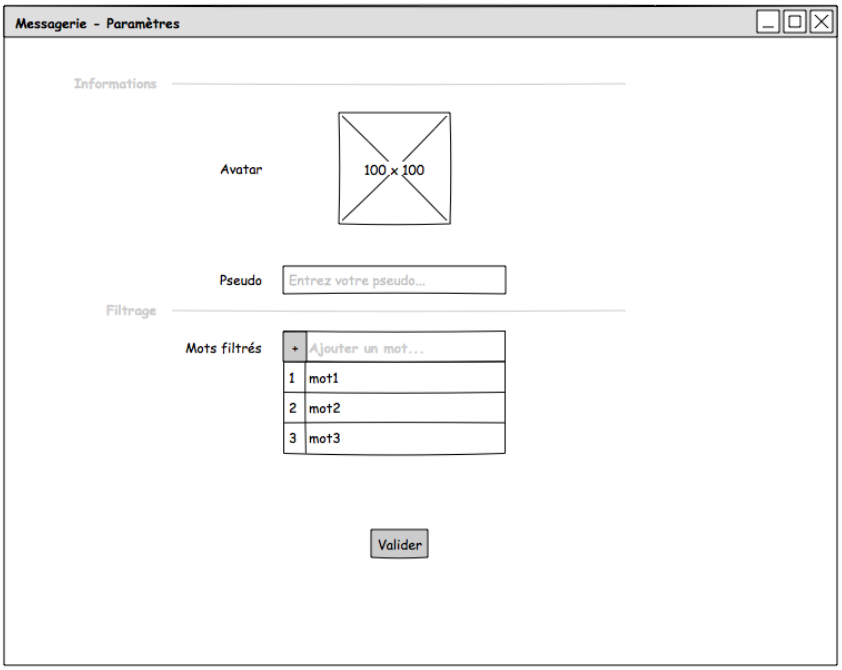

FIGURE 2.8 – Interface des paramètres

2.3 Spécifications opérationnelles

Le système de messagerie répond instantanément.

De plus, les messages sont sauvegardés tant qu'il subsiste un utilisateur connecté à la conversation.

Le serveur qui héberge l'application est disponible à chaque instant t pour permettre aux utilisateurs d'utiliser le service à n'importe quel moment.

Lors de l'utilisation de l'application, la reférence du compte est utilisée pour savoir qui participe à la conversation. Cette référence du compte transite donc entre l'application client et l'application serveur. Aucune mesure de sécurité n'est prévue pour éviter une transmission visible de cette référence.

Chapitre 3

Conception préliminaire

3.1 Modèle du domaine

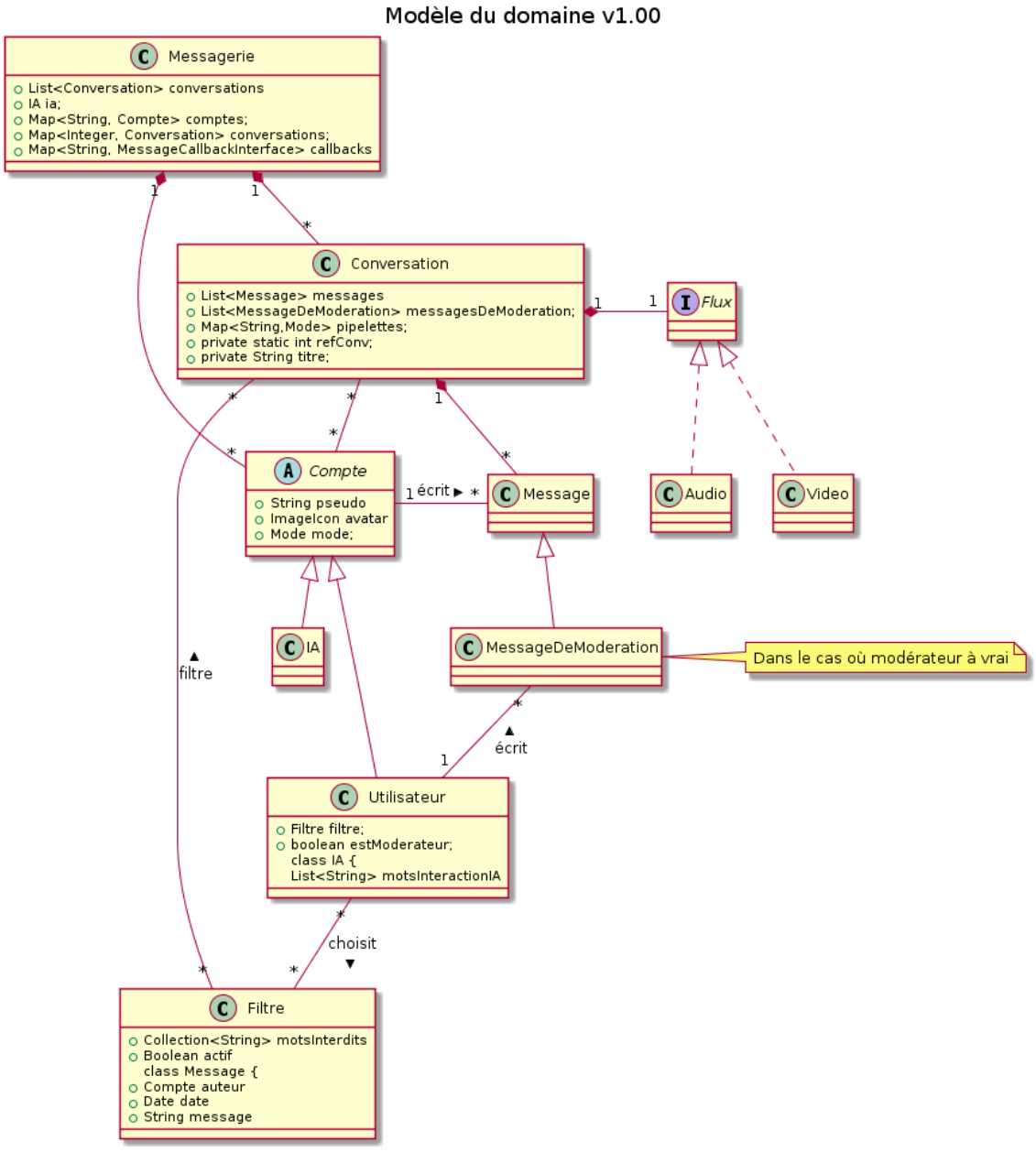

FIGURE 3.1 – Modèle du domaine

3.2 Diagrammes de séquence système

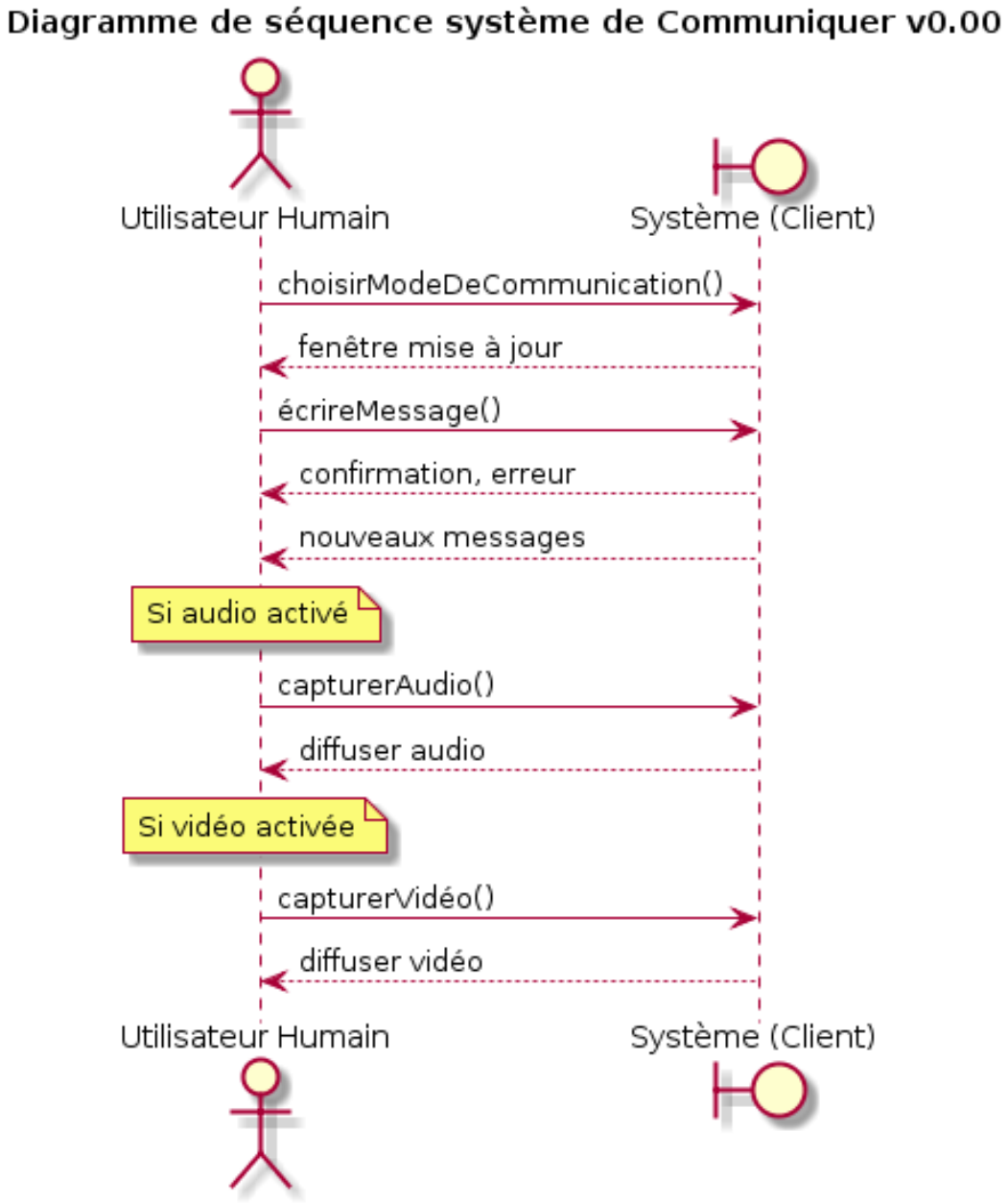

FIGURE 3.2 – Le diagramme de séquence système de « communiquer »

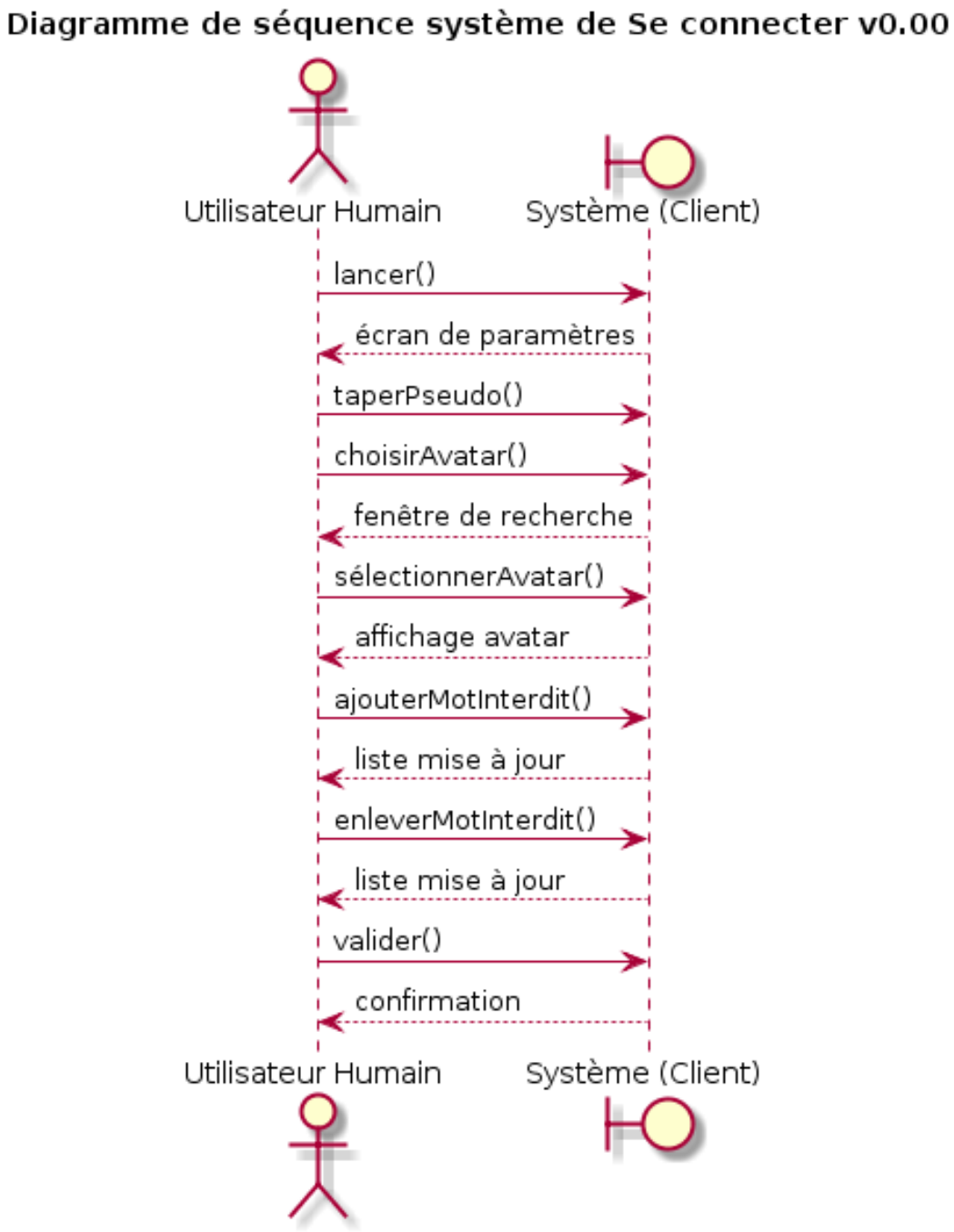

FIGURE 3.3 – Le diagramme de séquence système de « se connecter »

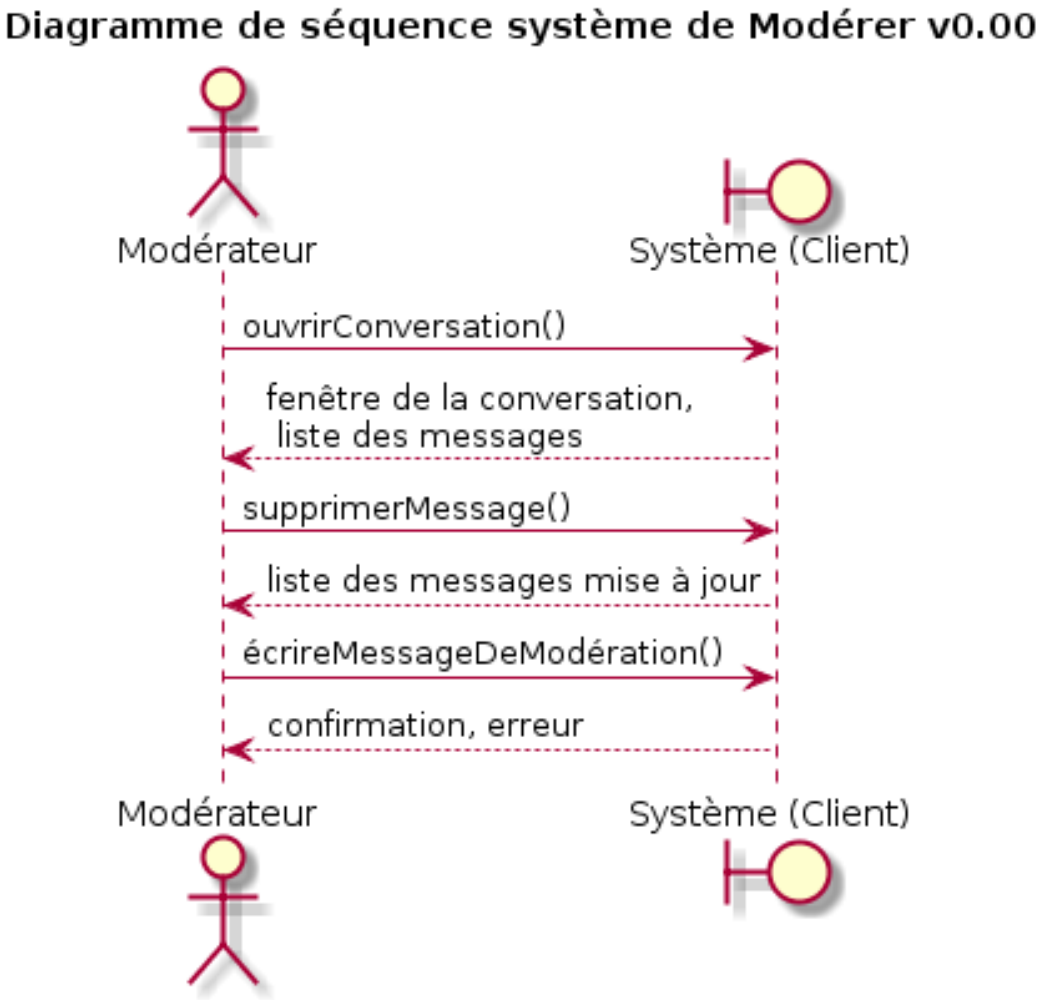

FIGURE 3.4 – Le diagramme de séquence système de « modérer »

3.2.1 Modification du modèle du domaine

La logique utilisée lors de la conception est restée la même. Nous avons cependant effectuée 2 changements principaux. Nous avons inclus beaucoup plus de champs dans la classe Messagerie. En effet nous avons tous centraliser via cette classe pour structurer les encapsulations. Par exemple, cette classe contient l'IA.
De plus, nous avions initialement une classe modérateur que nous avons supprimé et transformé en un booléen dans la classe utilisateur. Choix qui nous paraissait beaucoup plus pertinent.

3.3 Diagrammes d'activités

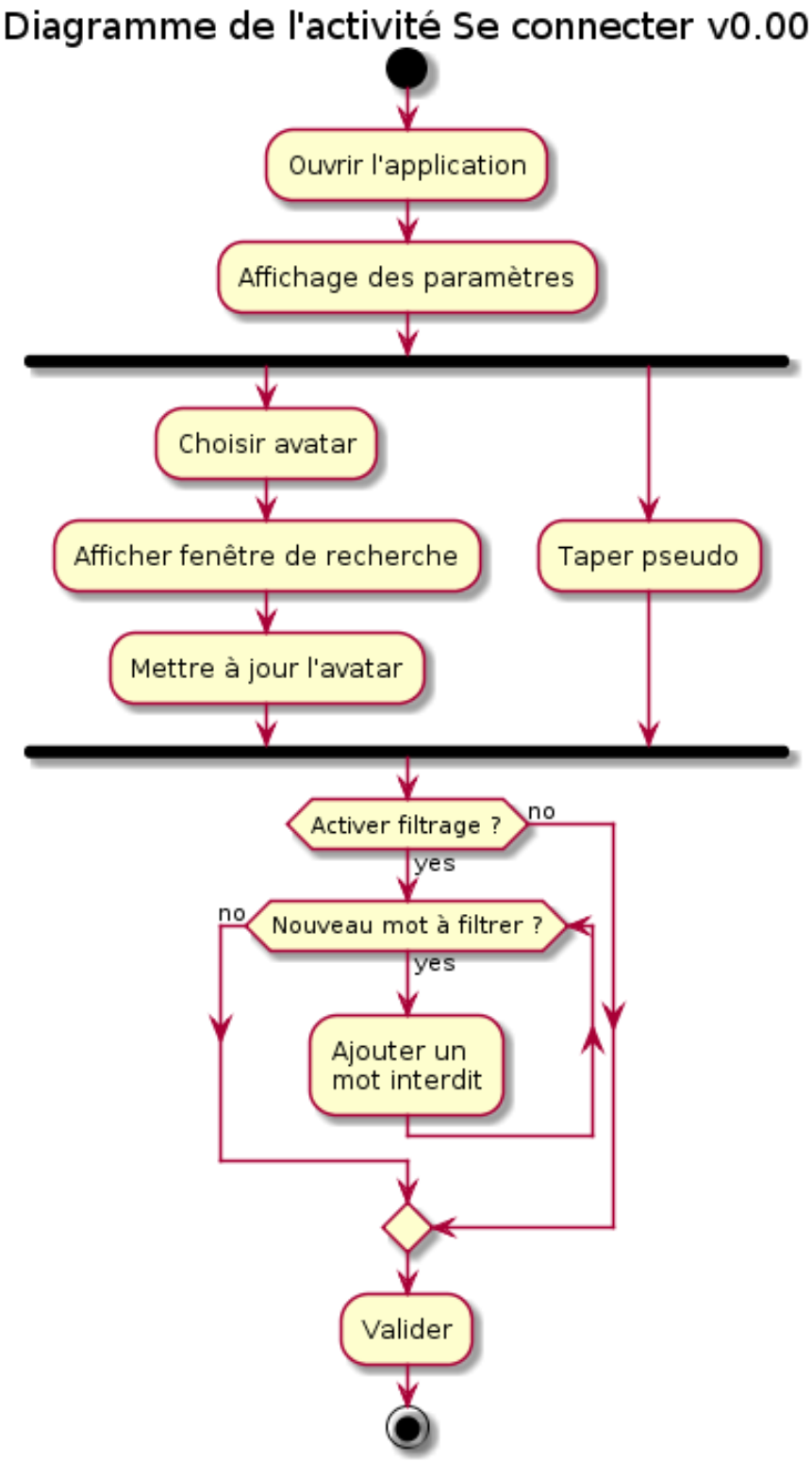

FIGURE 3.5 – Le diagramme de l'activité « se connecter »

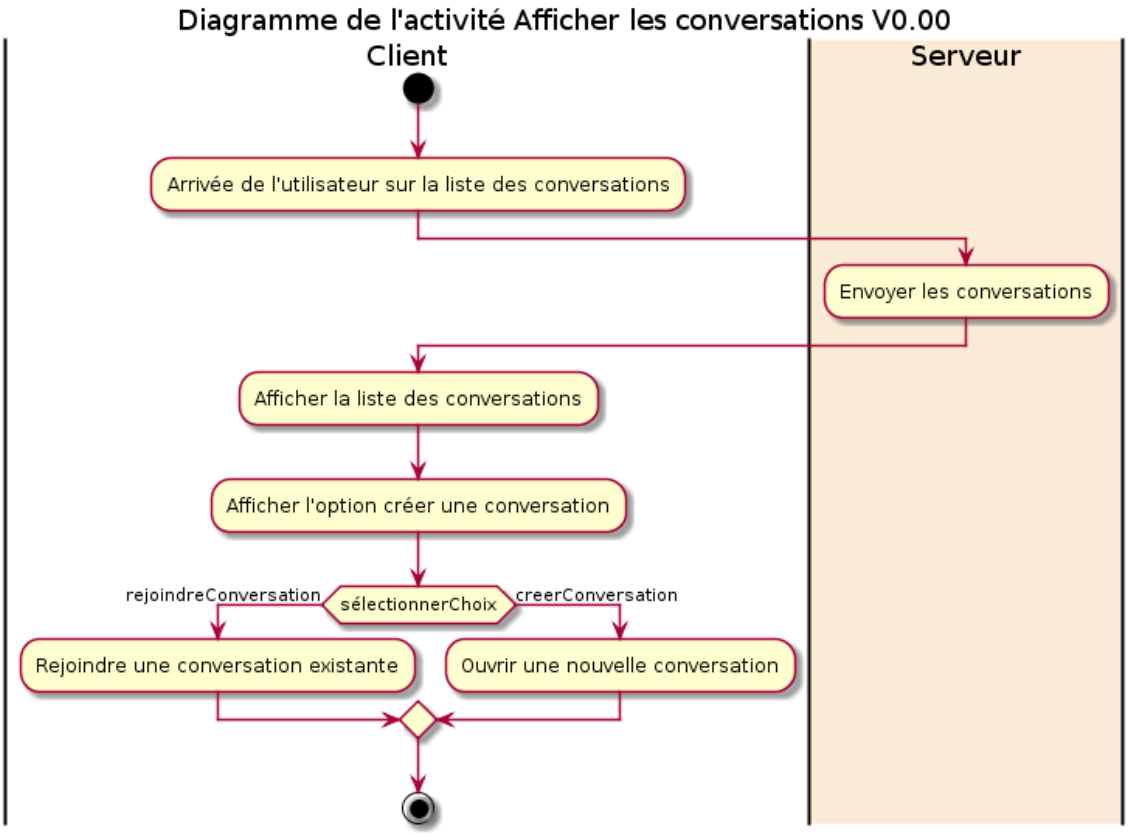

FIGURE 3.6 – Le diagramme de l'activité « choisir une conversation »

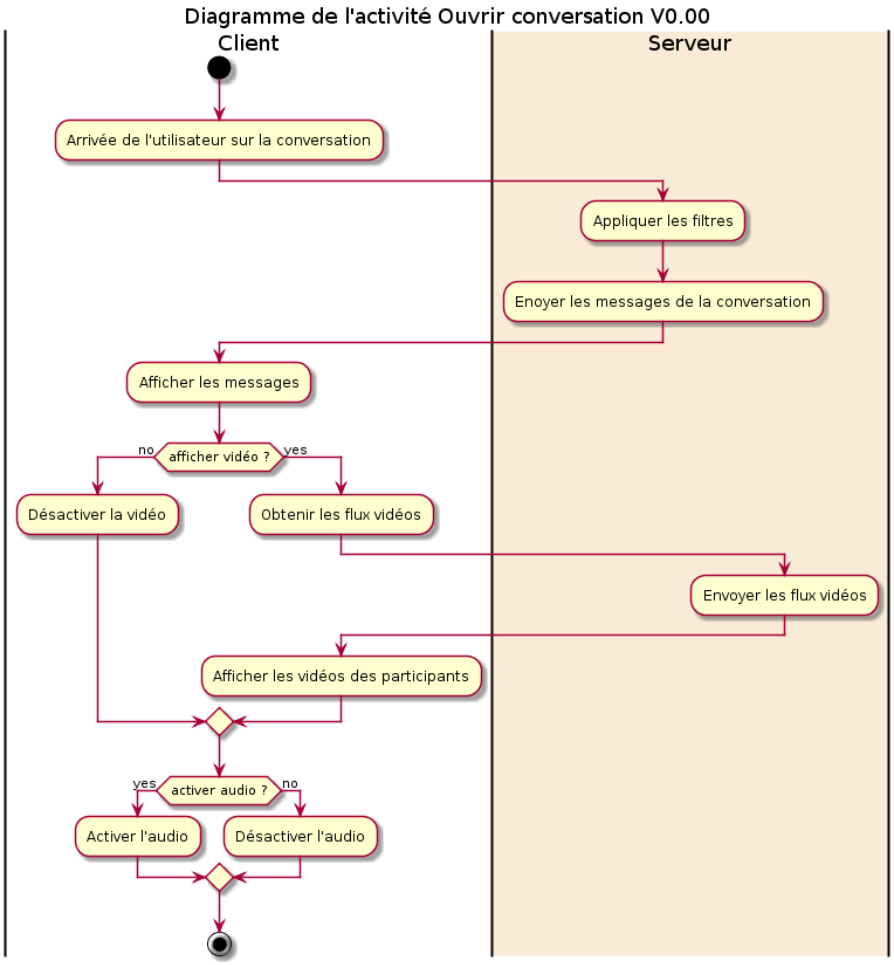

FIGURE 3.7 – Le diagramme de l'activité « ouvrir une conversation »

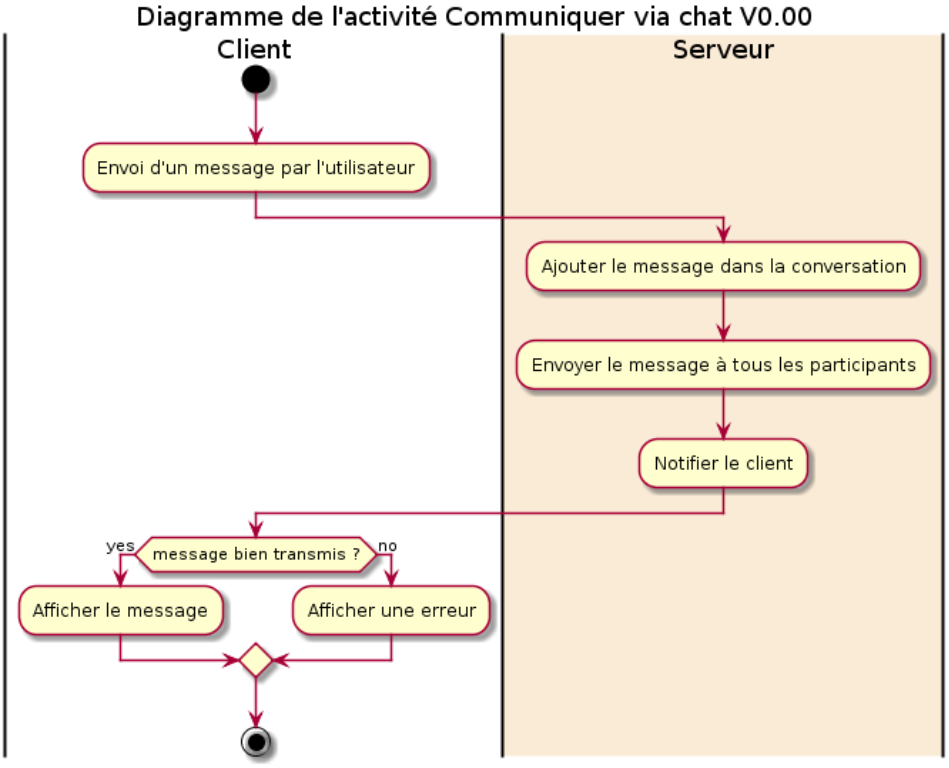

FIGURE 3.8 – Le diagramme de l'activité « communiquer »

3.4 Diagrammes d'interaction

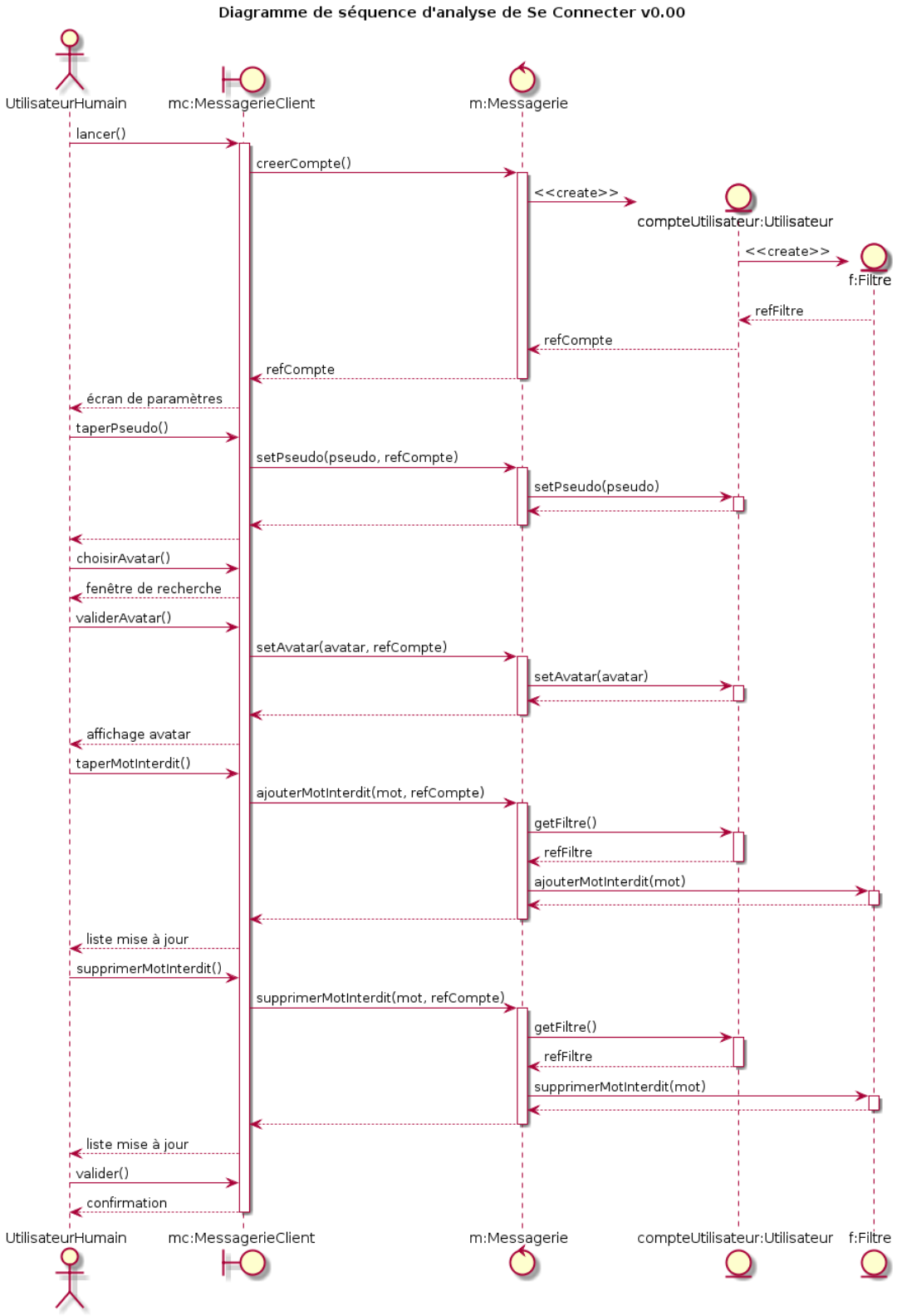

FIGURE 3.9 – Le diagramme de séquence d'analyse « se connecter »

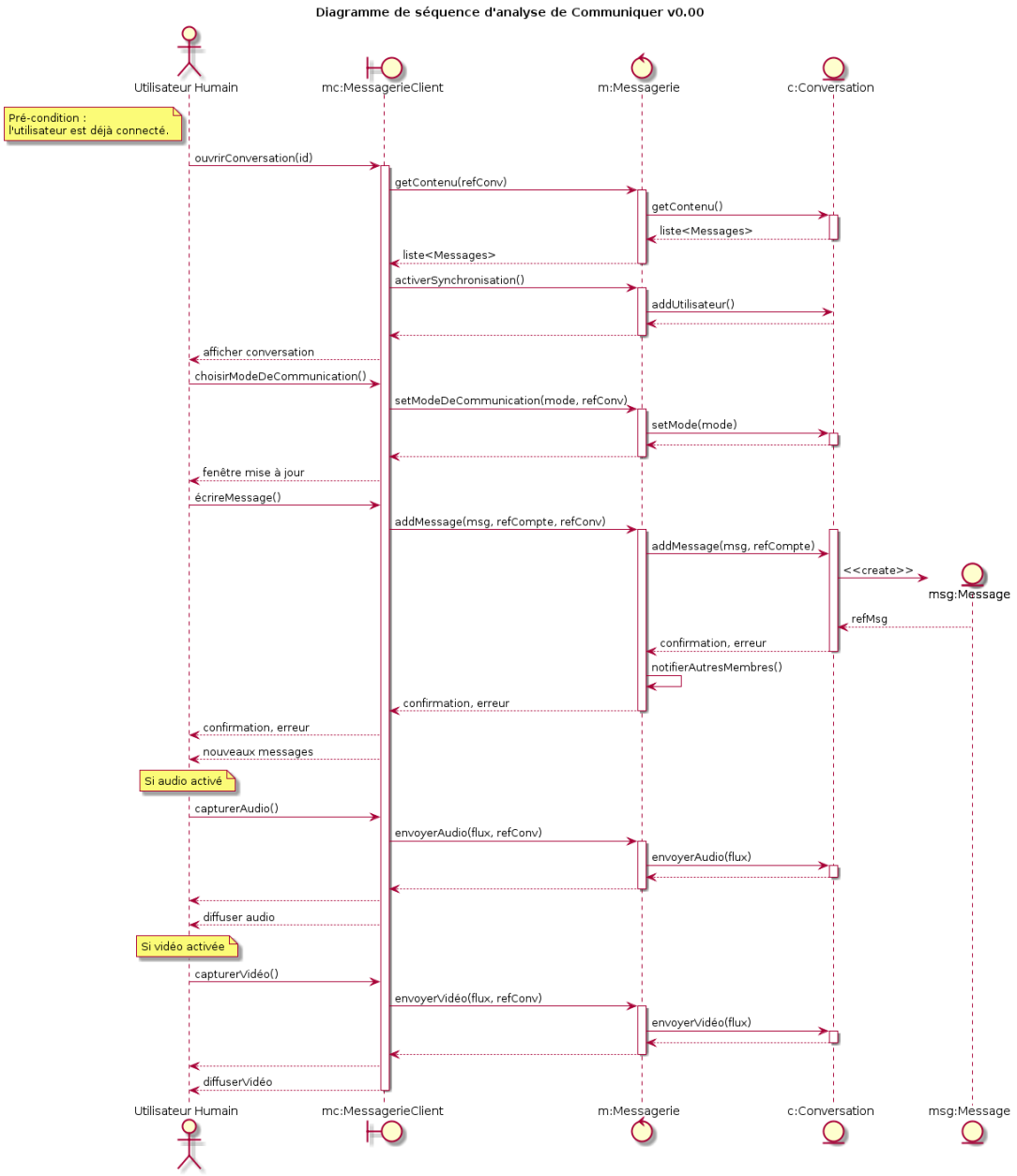

FIGURE 3.10 – Le diagramme de séquence d'analyse « Communiquer »

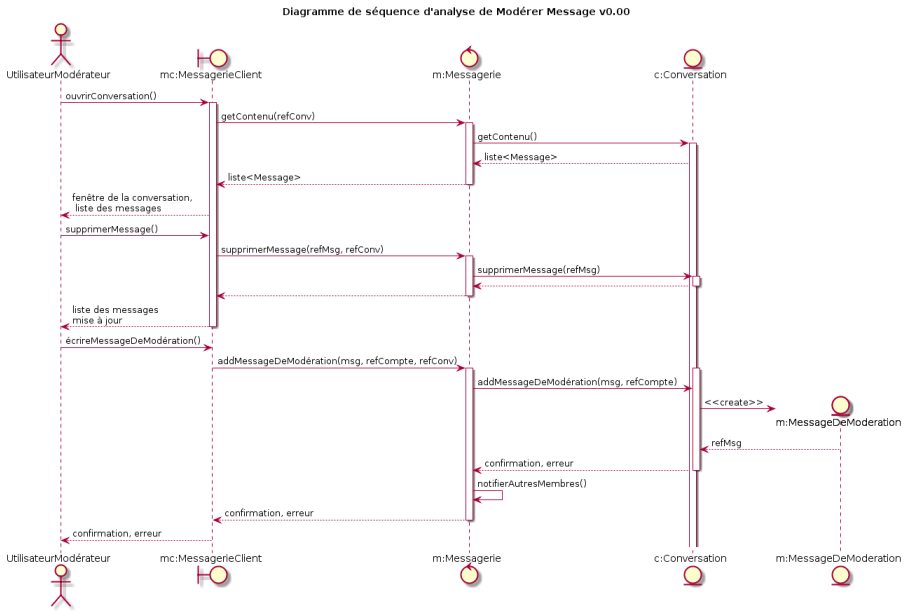

FIGURE 3.11 – Le diagramme de séquence d'analyse « Modérer Message »

3.5 Diagramme de communication client serveur

3.6 Découpage de l'application en *packages*

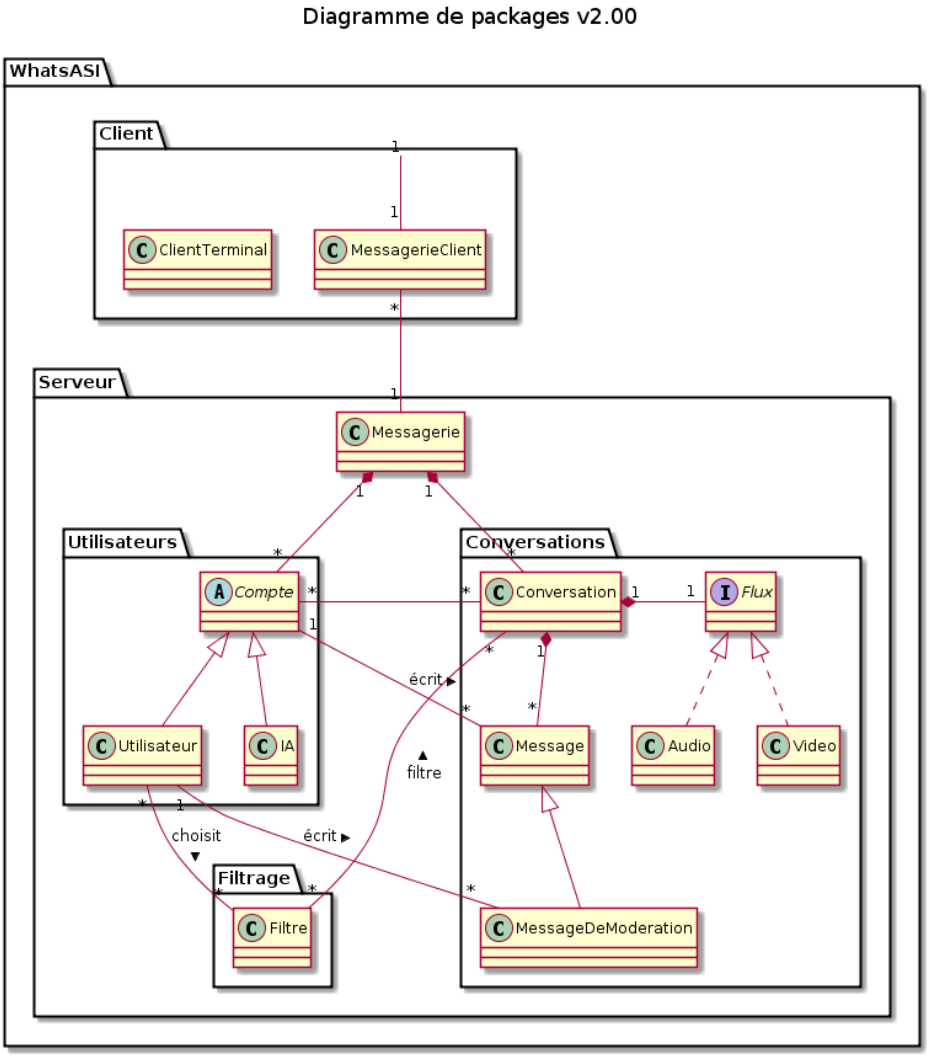

FIGURE 3.12 – Le diagramme de packages

Nous avons choisi de travailler avec des threads et callback. Comme indiqué sur la figure ci dessous. Nous sommes en multiclient : il y a donc un callback par client.

Chapitre 4

Conception détaillée

Chapitre 5

Implémentations et tests

L'objectif de cette partie est de préciser nos différents choix techniques. De même que de justifier le bon fonctionnement de nos fonctionnalités.

5.1 Choix techniques

5.1.1 Langage de programmation, librairie, technologie pour la notion d'informatique répartie

5.1.1.1 Langage de programmation

Nous avons choisi d'utiliser un langage orienté objet : JAVA. L'utilisation d'un tel langage nous paraissait pertinente et adaptée à notre projet que nous voulions orienté logiciel.
Premièrement, il était assez simple de passer d'une conception UML à une programmation JAVA.
Deuxièmement JAVA permettait la gestion des exceptions.
De plus ce langage est celui le plus maîtrisé par l'ensemble du groupe. Enfin il s'adaptait parfaitement à l'utilisation de RMI.

5.1.1.2 Librairie

Nous avons choisi d'utiliser java-fx, qui offre un certain nombre de fonctionnalités graphiques. Cela nous permettait de découvrir une nouvelle librairie sachant que nous connaissions déjà swing.

5.1.1.3 Technologie pour la notion d'informatique répartie

Nous avons fait le choix d'utiliser un système de communication RMI avec de l'appel de méthode sur des objets distants. Nous avons choisi cette technologie car elle n'est pas orientée WEB de façon pure (moins que REST) et nous souhaitions réaliser un travail basé sur un logiciel et non un site WEB.
De plus il était nécessaire d'utiliser des threads pour avoir plusieurs clients.

5.1.2 Guide d'utilisation

Voici les différentes étapes à suivre pour utiliser notre application :

5.1.2.1 Concernant l'installation

— Compiler le projet via le fichier compile.shs
— Lancer le RmiRegistry ./launchRmiregistry.sh
— Lancer le serveur ./launchServer.sh (terminal si version terminale)
— Lancer le client dans un autre terminal ./launchClient.sh (terminal si version terminale)

5.1.2.2 Concernant l'utilisation

— Suivre les instructions présentes sur l'IHM
— Pour lancer l'IA mettre \bonjour, toutes les instructions apparaîtront.
— compléter certains trucs

5.2 Tests de validation

A COMPLETER

5.2.1 Tests des spécifications fonctionnelles

5.2.2 Tests des spécifications d'interface

5.2.3 Tests des spécifications opérationnelles

Chapitre 6

Perspectives

6.1 Développement

6.1.1 Ce que nous avons développé

A compléter

6.1.2 Ce que nous avons abandonné

Nous n'avons pas pris le temps de réaliser l'échange via le flux audio et le flux vidéo par manque de temps.

6.2 Ce qui aurait pu être ajouté

Nous aurions pu ajouter une fonctionnalité qui permet de sécuriser le passage de la référence du compte lors de messages. En effet, dans notre cas un utilisateur malveillant peut récupérer cette référence. Le fait de la crypter ne changerait rien, ce qu'il faudrait c'est utiliser un cryptage asymétrique. Cependant en raison des objectifs pédagogiques de ce projet, de la pertinence de sécuriser cette référence, et du temps que cela impliquait nous n'avons pas réalisé cette fonctionnalité. La conséquence est minime car nos données ne sont pas stockées via une BD et non persistantes. De plus les informations qui transitent sont non sensibles.