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

11 mars 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 Besoins détaillés

1.3.1 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.

1.3.2 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.

1.4 Contraintes

1.4.1 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.

1.4.2 Contraintes logicielles

Le client doit disposer du Java Runtime Environnement 8.

Chapitre 2

Conception préliminaire

2.1 Diagramme de classe

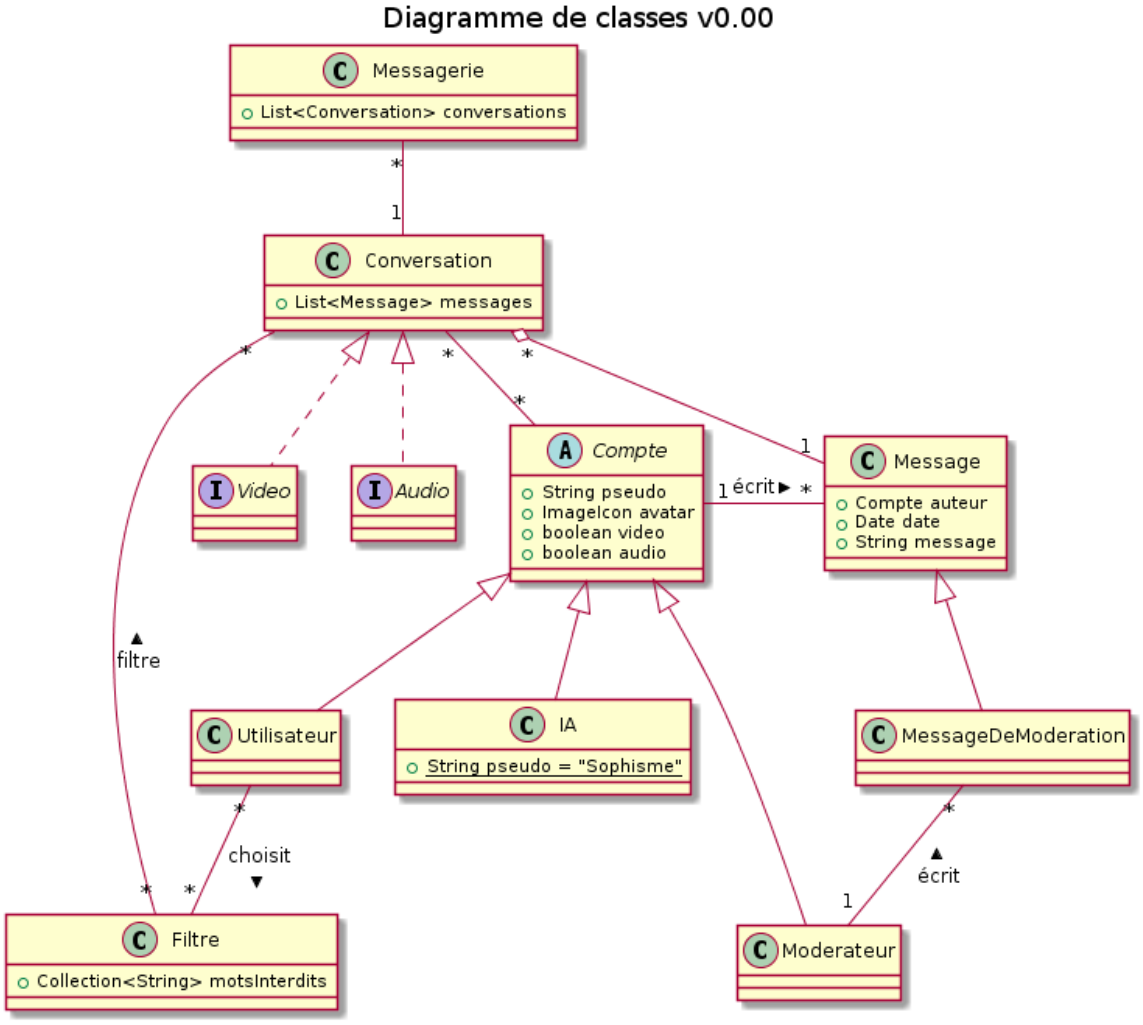

FIGURE 2.1 – Le diagramme des classes de conception préliminaire

2.2 Diagrammes de séquence système

2.3 Diagrammes d'activités

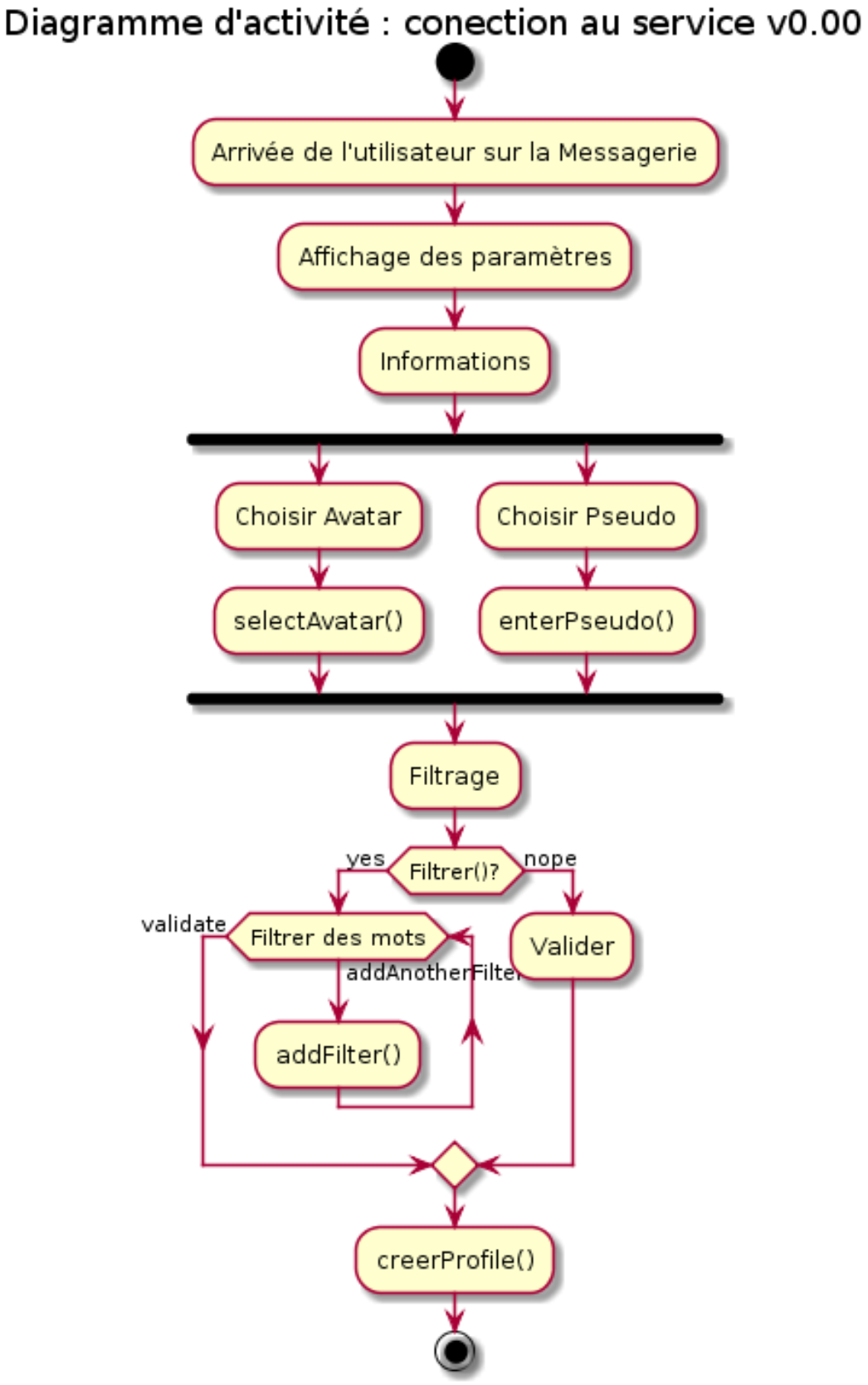

FIGURE 2.2 – Le diagramme de l'activité « se connecter »

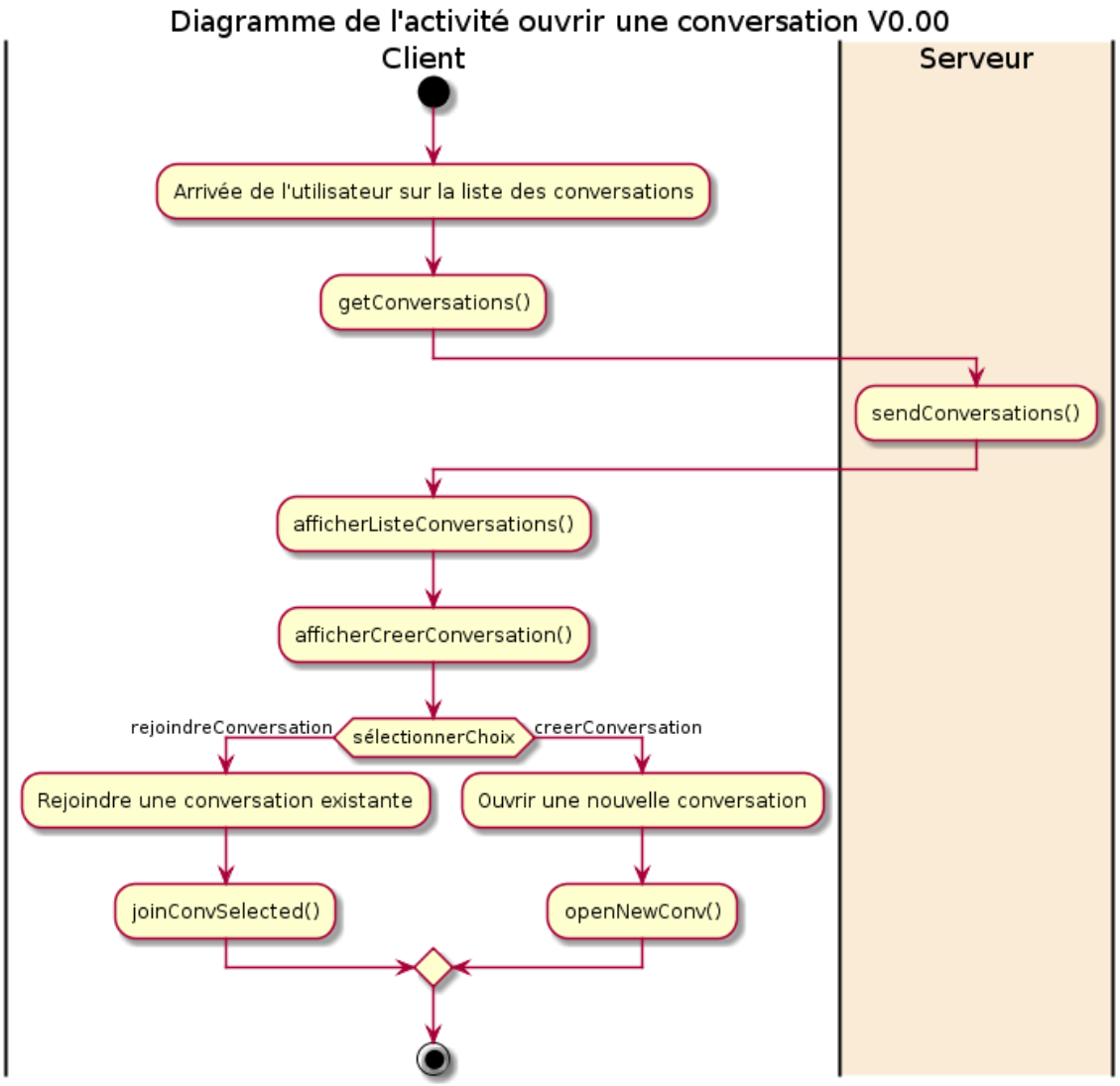

FIGURE 2.3 – Le diagramme de l'activité « choisir une conversation »

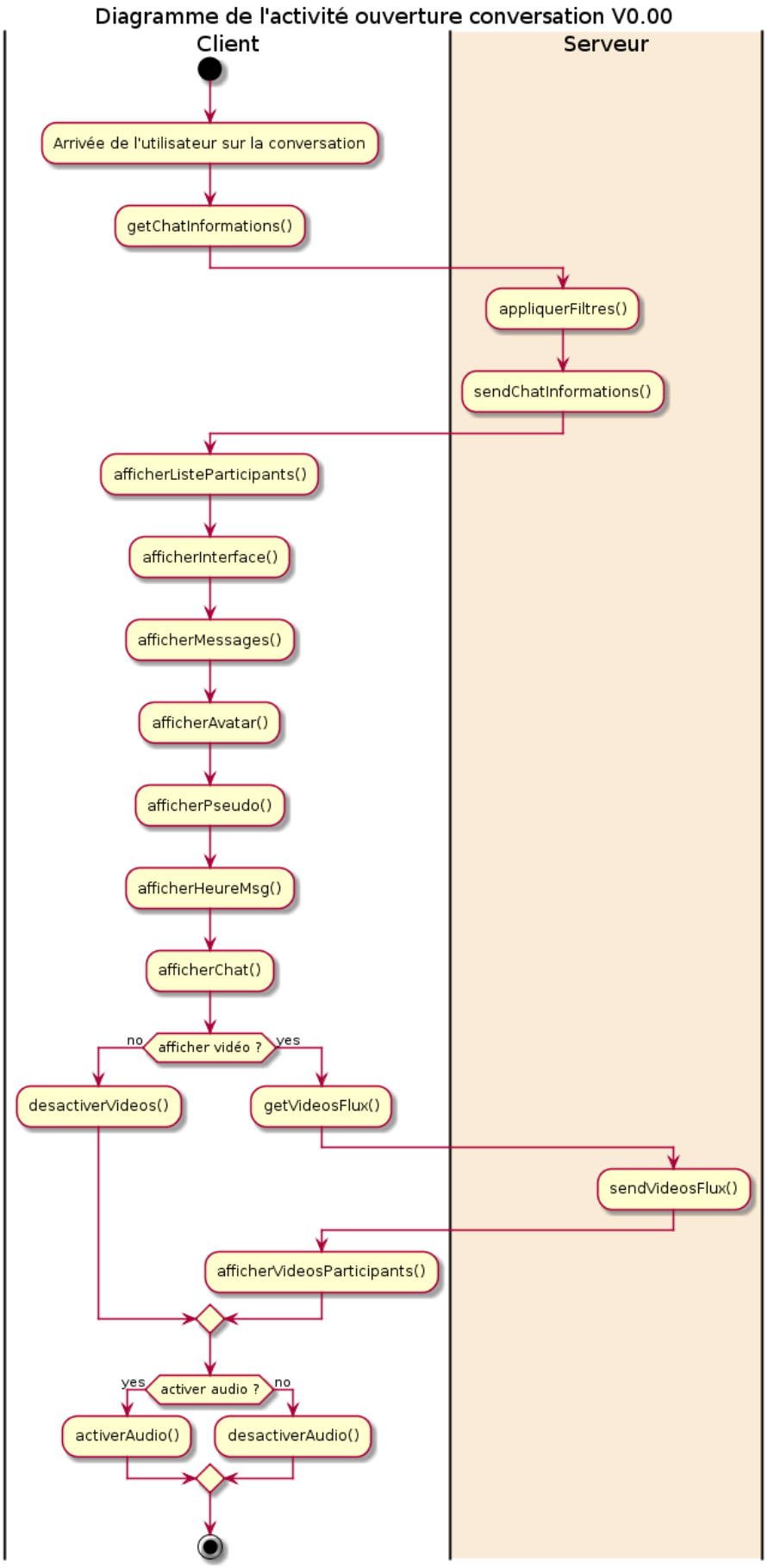

FIGURE 2.4 – Le diagramme de l'activité « ouvrir une conversation »

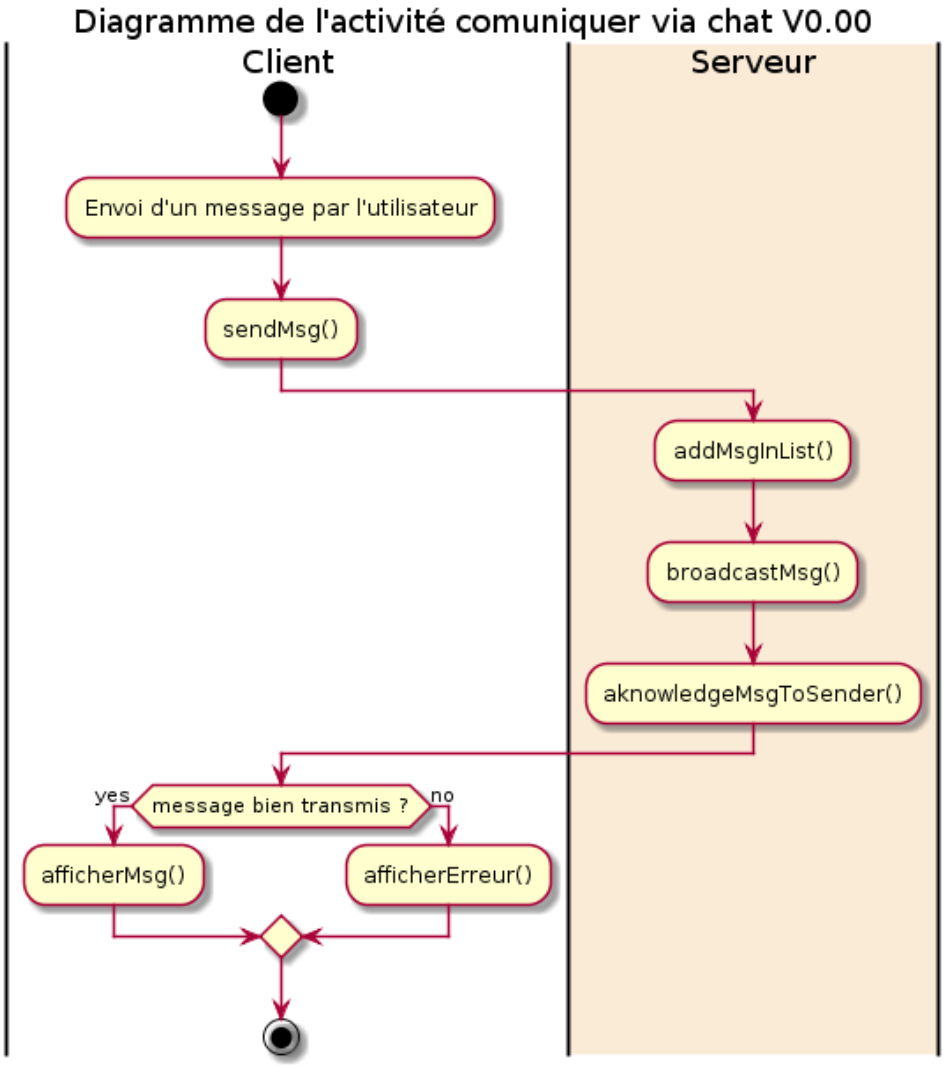

FIGURE 2.5 – Le diagramme de l'activité « communiquer »

2.4 Diagrammes d'interaction

2.5 Packages du projet et signatures externes

2.5.1 Découpage de l'application en *packages*

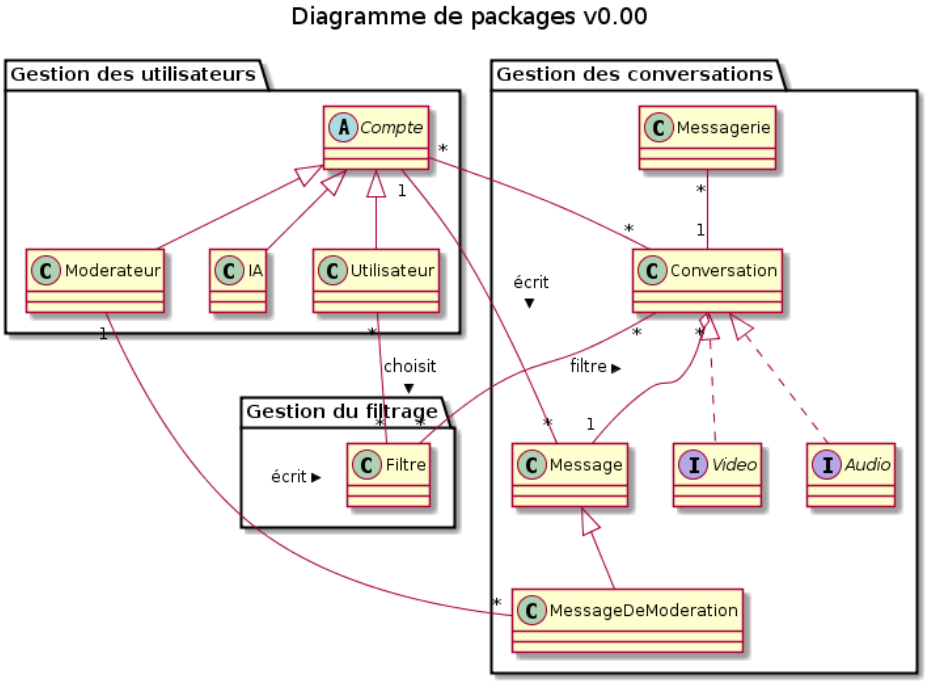

FIGURE 2.6 – Le diagramme de packages

2.5.2 Signatures externes de chaque *package*

2.5.2.1 « Gestion des conversations »

Ici sont listées les signatures externes des différents composants de chacun des *packages* de l'application.

Classe Messagerie

- `public List<Conversation> getConversations();`

Classe Conversation

- `public List<Message> getMessages();`
- `public List<Compte> getListeParticipants();`
- `public void afficherConversation();`

Interface Video

- `public void startVideo();`
- `public void endVideo();`

Interface Audio

- `public void startAudio();`
- `public void endAudio();`

Classe Message
- `public Compte getAuteur();`
- `public Date getDate();`
- `public String getMessage();`

Classe MessageDeModeration
- `public void sendMessageDeModeration(String message);`

2.5.2.2 « Gestion des utilisateurs »

Classe Abstraite Compte
- `public String getPseudo();`
- `public ImageIcon getAvatar();`
- `public boolean canVideo();`
- `public boolean canAudio();`
- `public void setVideo(boolean b);`
- `public void setAudio(boolean b);`

Classe Moderateur
Hérite de la classe abstraite `Compte`.

Classe IA
Hérite de la classe abstraite `Compte`.
- `public Collection<String> getMessagesPossiblesIA();`
- `public String answerMessage(String messageRecu);`

Classe Utilisateur
Hérite de la classe abstraite `Compte`.
- `public Filtre getFiltre();`

2.5.2.3 « Gestion du filtrage »

Classe Filtre
- `public Collection<String> getMotsInterdits();`
- `public void addMotInterdit(String mot);`

Chapitre 3

Conception détaillée