****

**Conception**

**Orientée Objet**

**Chat System**

**Emma Baudoint / Matthieu Jacques**

4IR C1 | 2020/2021

Table des matières

[Introduction 3](#_Toc63926794)

[Diagramme de cas d’utilisation 3](#_Toc63926795)

[Diagramme de séquence: 6](#_Toc63926796)

[Diagramme de classe première version 8](#_Toc63926797)

[Diagramme de classe version finale 9](#_Toc63926798)

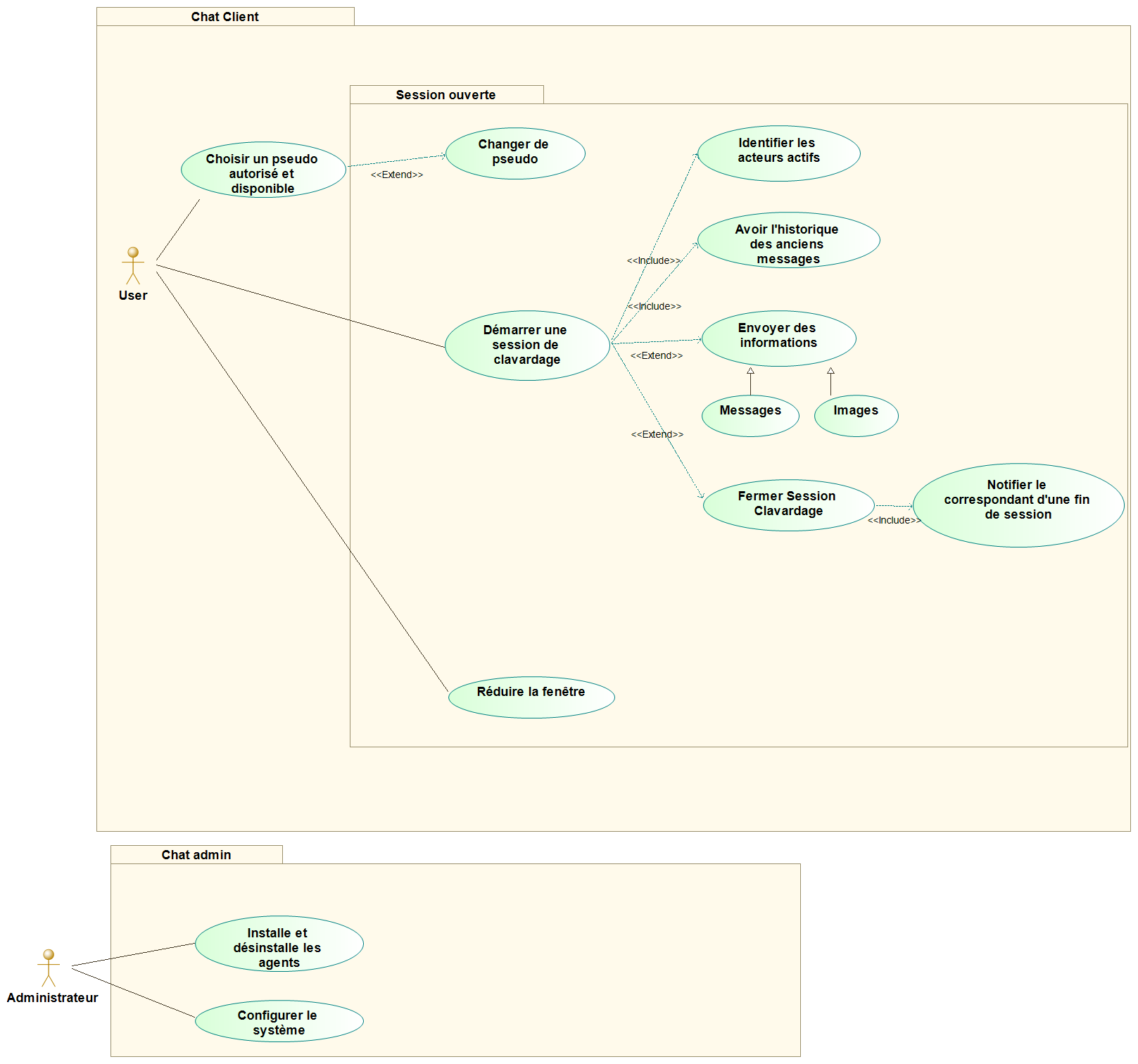
[Diagrammes d'état 10](#_Toc63926799)

[Diagrammes composites 12](#_Toc63926800)

# Introduction

Les diagrammes présentés ici ont été réalisés avant le projet. C’est donc une première version de la vision que nous avions de ce projet. Nous avons revu progressivement le diagramme de classe et avons reproduit la version finale du projet. Les autres diagrammes ont nécessité moins ou aucune modification par rapport à la première version du rapport. Si les images sont difficilement lisibles, elles sont disponibles dans le dossier COO du projet GitHub afin de faire un zoom dessus.

# Diagramme de cas d’utilisation



Un utilisateur doit être capable de choisir un pseudo et d’ouvrir une session de clavardage. A partir de là, il doit pouvoir voir les utilisateurs actifs, envoyer des informations à une personne, charger l’historique des messages avec cette personne et fermer la session avec cette personne.

Nous devrons aussi prévoir que l’administrateur devra être capable d’installer et de désinstaller cette application ainsi que de la configurer au besoin.

# Diagramme de séquence :

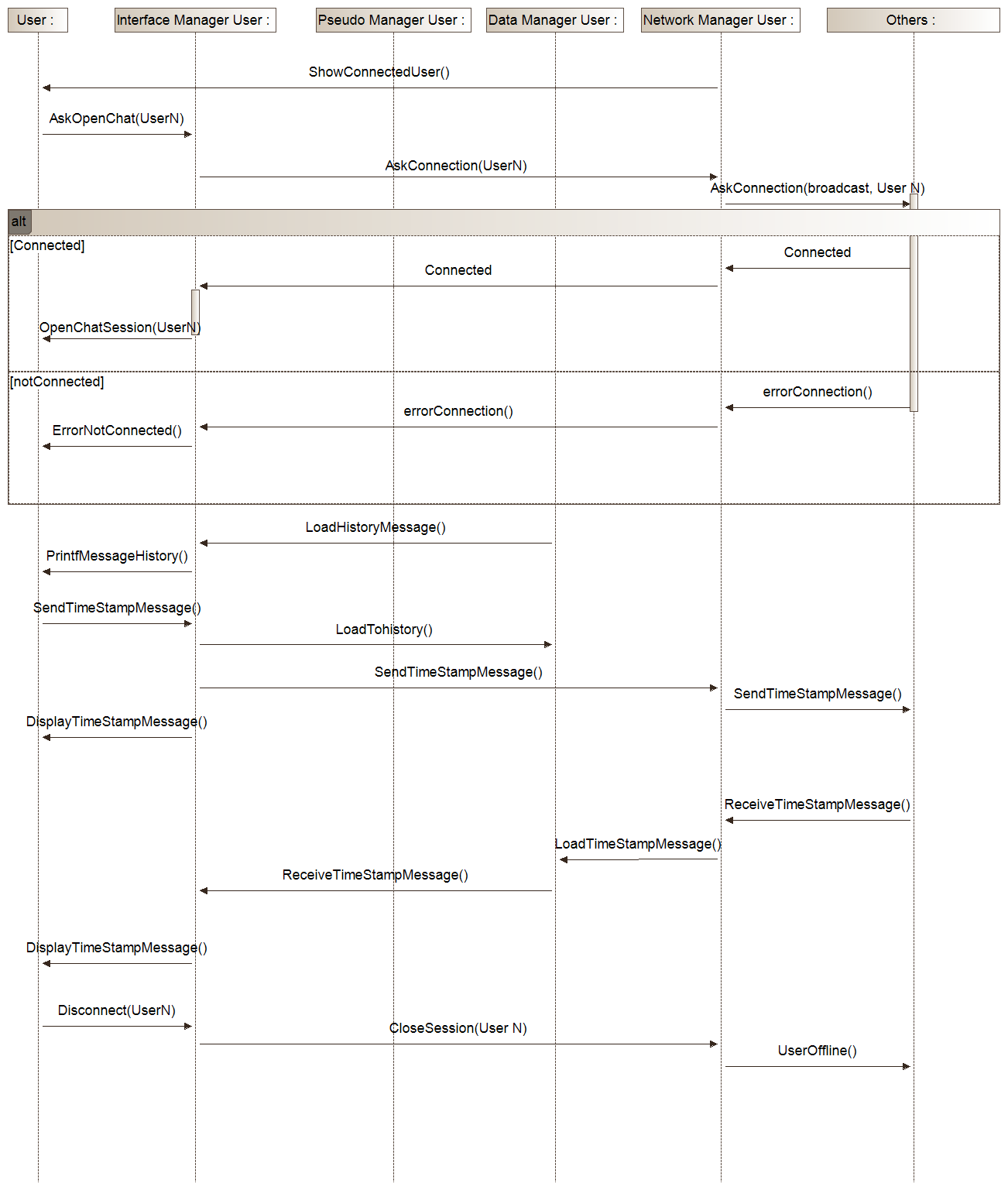


Diagramme de séquence de la connexion d’un utilisateur et de son échange de message avec d’autres personnes.

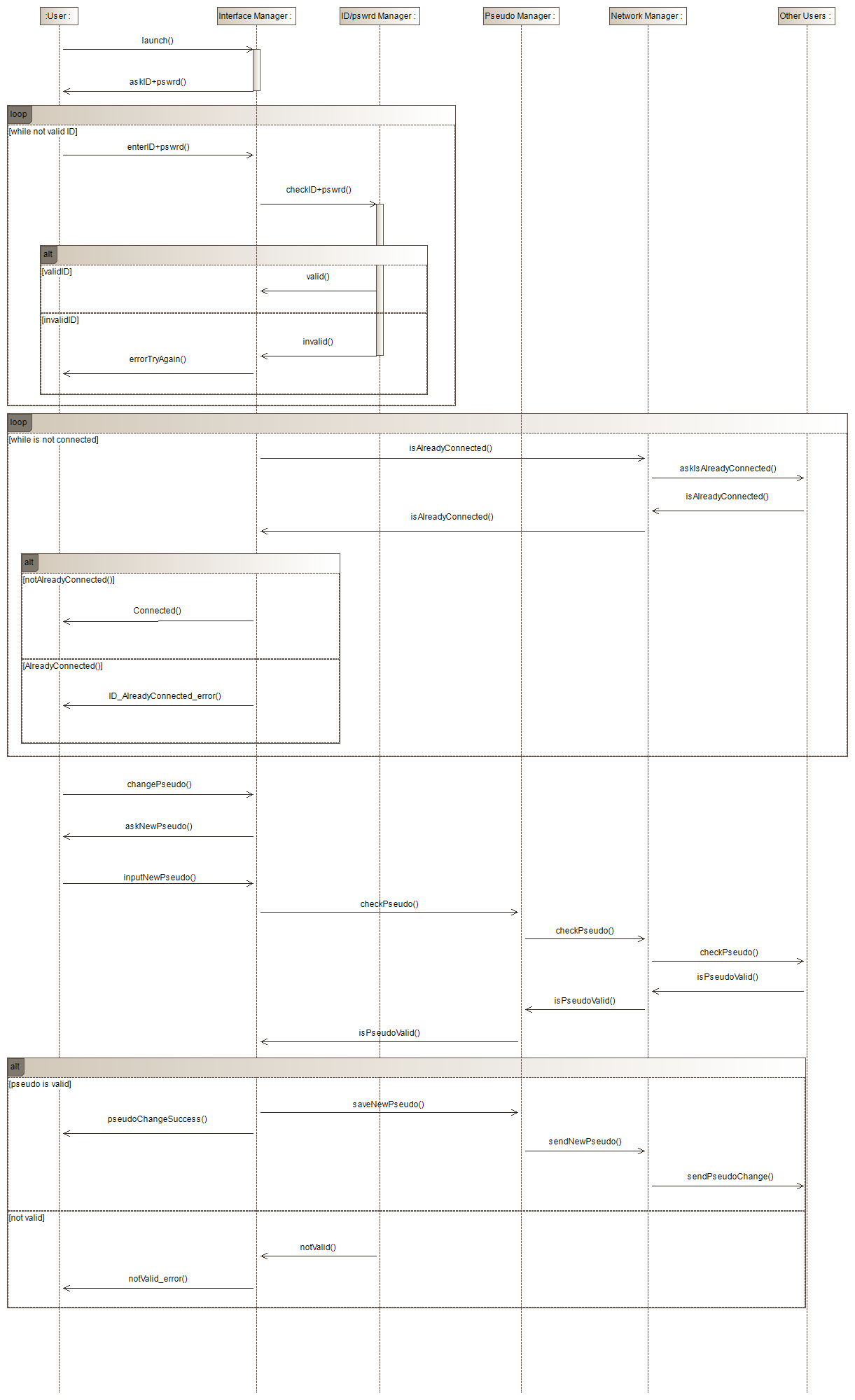
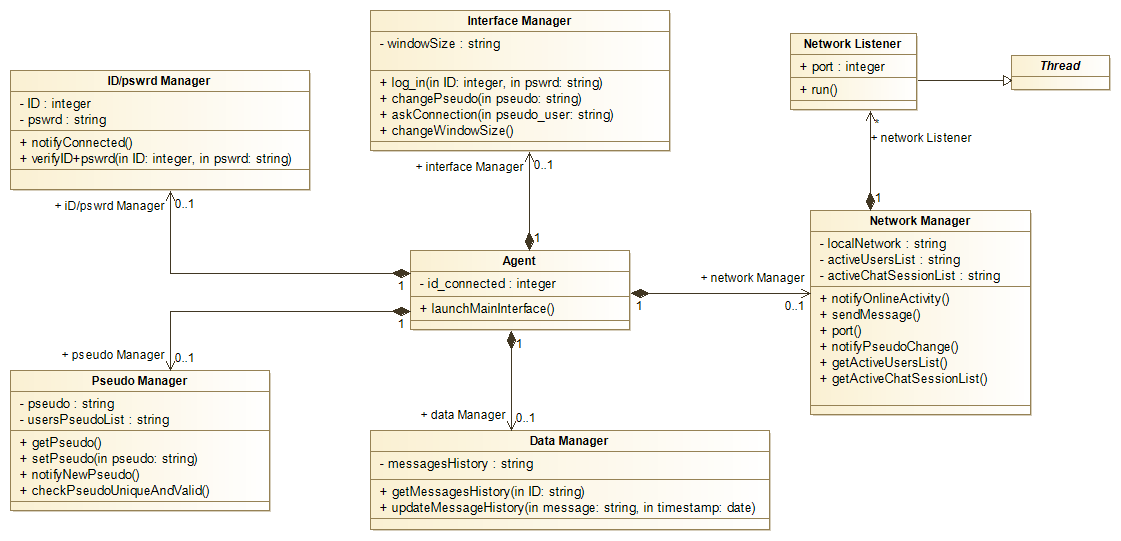


Diagramme de la demande de changement de pseudo d’un utilisateur.

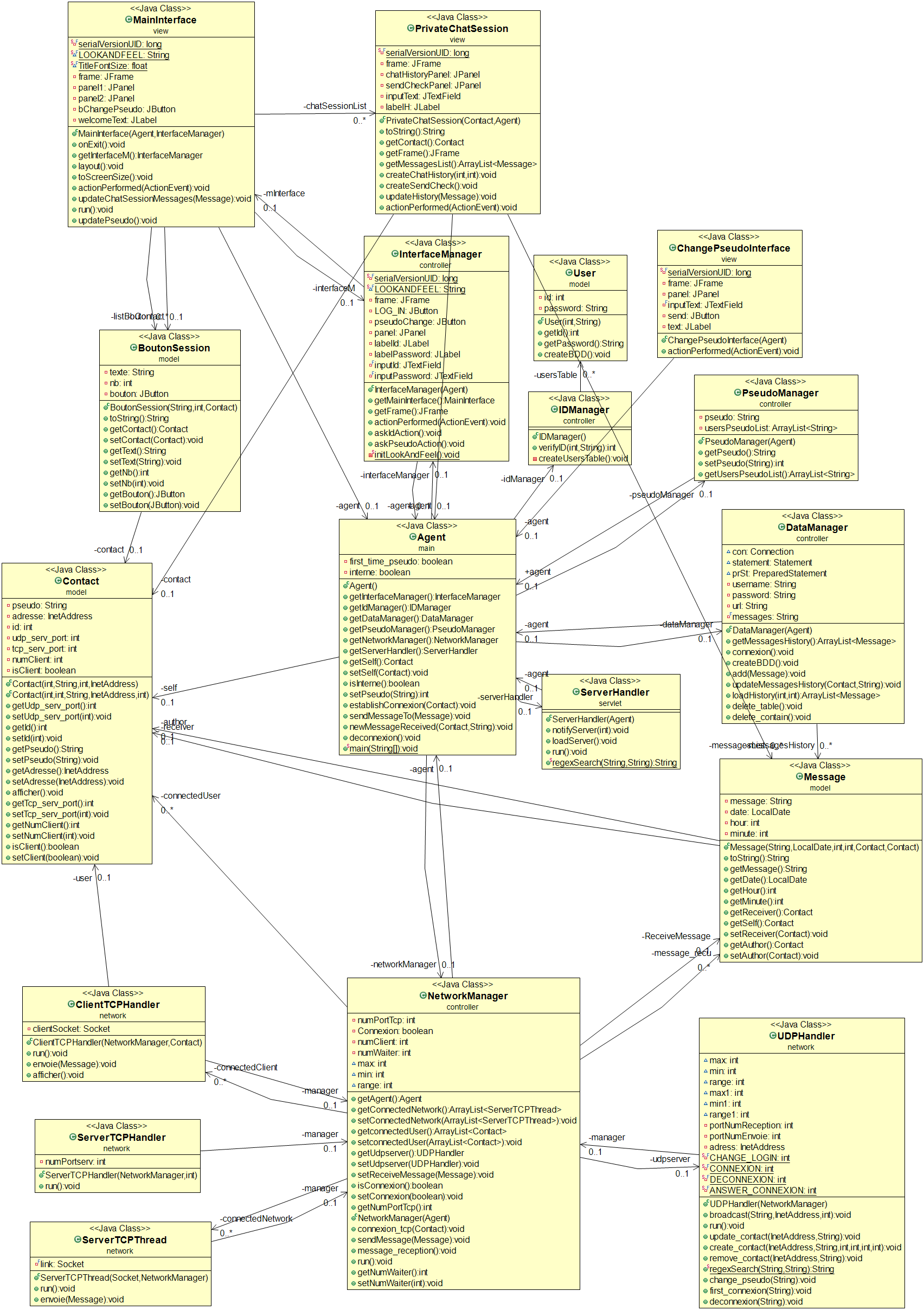
# Diagramme de classe première version



A cette version de notre diagramme de classe, nous avons dû rajouter la mise en place d’un servlet, et des classes annexes de types modèles afin de décomplexifier le code. Nous avons aussi dû détailler la partie Network qui s’est avéré bien plus complexe que prévue.

# Diagramme de classe version finale

Ce diagramme représente l’ensemble de notre projet au stade final. Nos choix d’implémentation sont détaillés dans la partie choix d’implémentation.

****

# Diagrammes d'état

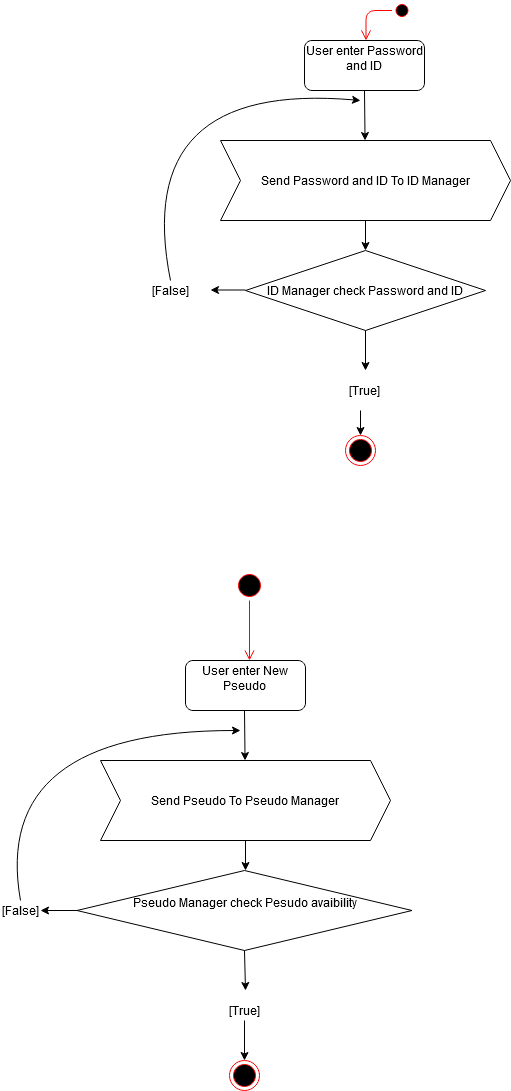


Diagramme d’état de l’identification et du choix du pseudo

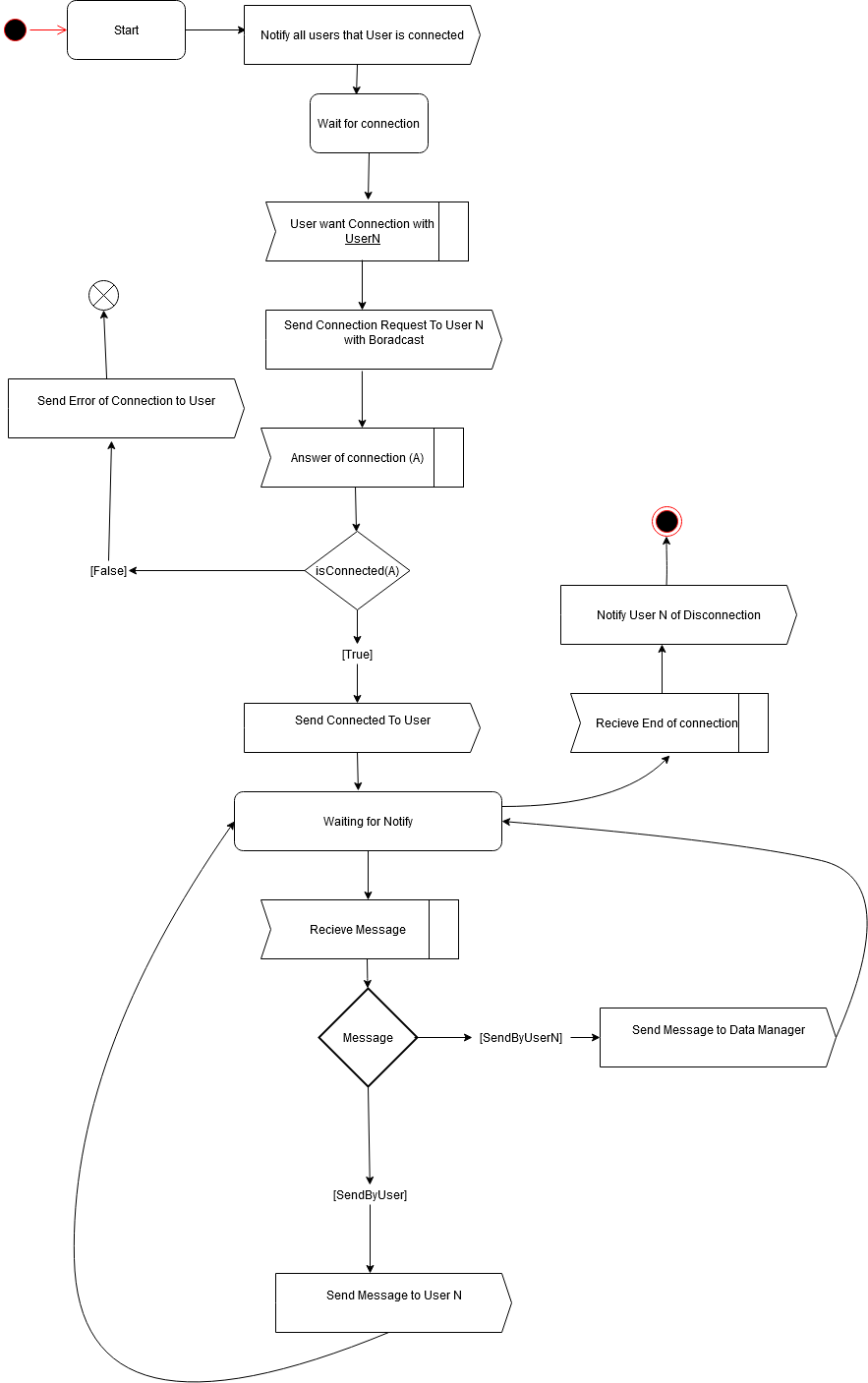
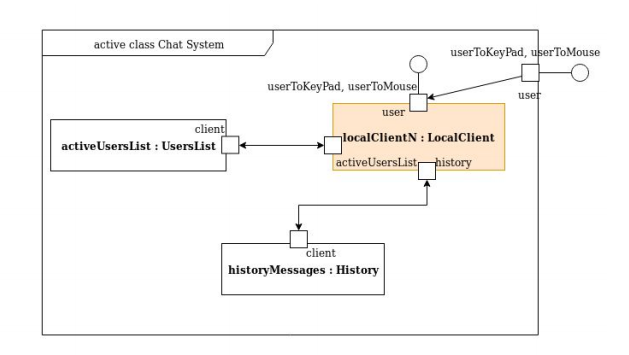


Diagramme d’état de l’envoie d’un message à un utilisateur n

# Diagrammes composites



Ce diagramme représente la connexion de notre utilisateur en local.

Active class Agent

# Choix d’implémentation :

## Classe Agent

La classe Agent est la classe centrale. Elle nous a permis de réunifier notre code après avoir travaillés chacun sur différentes parties.

## Doubles références

Les classes comportent des doubles références, notamment à la classe principale “Agent”. Nous avons favorisé cette conception plutôt que l’utilisation d’observer pour la simplicité d’appel de nos fonctions d’interfaces. Cela rend cependant le diagramme de classe qui en découle plus désordonné.

## Versions du projet

Le projet est divisé en deux programmes : chat\_externe et chat\_interne. Cela répond à la problématique de l’architecture 3-tiers faible proposée par le cahier des charges. Le premier programme est destiné à être installé sur une machine distante, en dehors du réseau, tandis que l’autre fonctionne à l’intérieur du réseau de l’entreprise. Le code est sensiblement le même à l’exception d’une variable booléenne « interne » qui change la façon d’accéder aux autres utilisateurs (broadcast ou servlet). Nous avons choisi de séparer le projet de cette manière plutôt que de demander à l’utilisateur de renseigner s’il se trouve sur le réseau ou non. De cette manière nous rendons l’utilisation du logiciel plus simple et seul l’administrateur a un choix à faire au moment de l’installation.

## Prévenir d’une connexion

Un agent interne au réseau prévient de sa connexion et sa déconnexion par broadcast udp. Les autres utilisateurs du réseau mettent alors à jour leur liste d’utilisateurs disponibles et