Rapport de Projet COO & POO

SYSTEME DE CLAVARDAGE DISTRIBUE INTERACTIF MULTI-UTILISATEUR TEMPS REEL

DESPORTES Kilian
IMEKRAZ Yanis
2019 - 2020

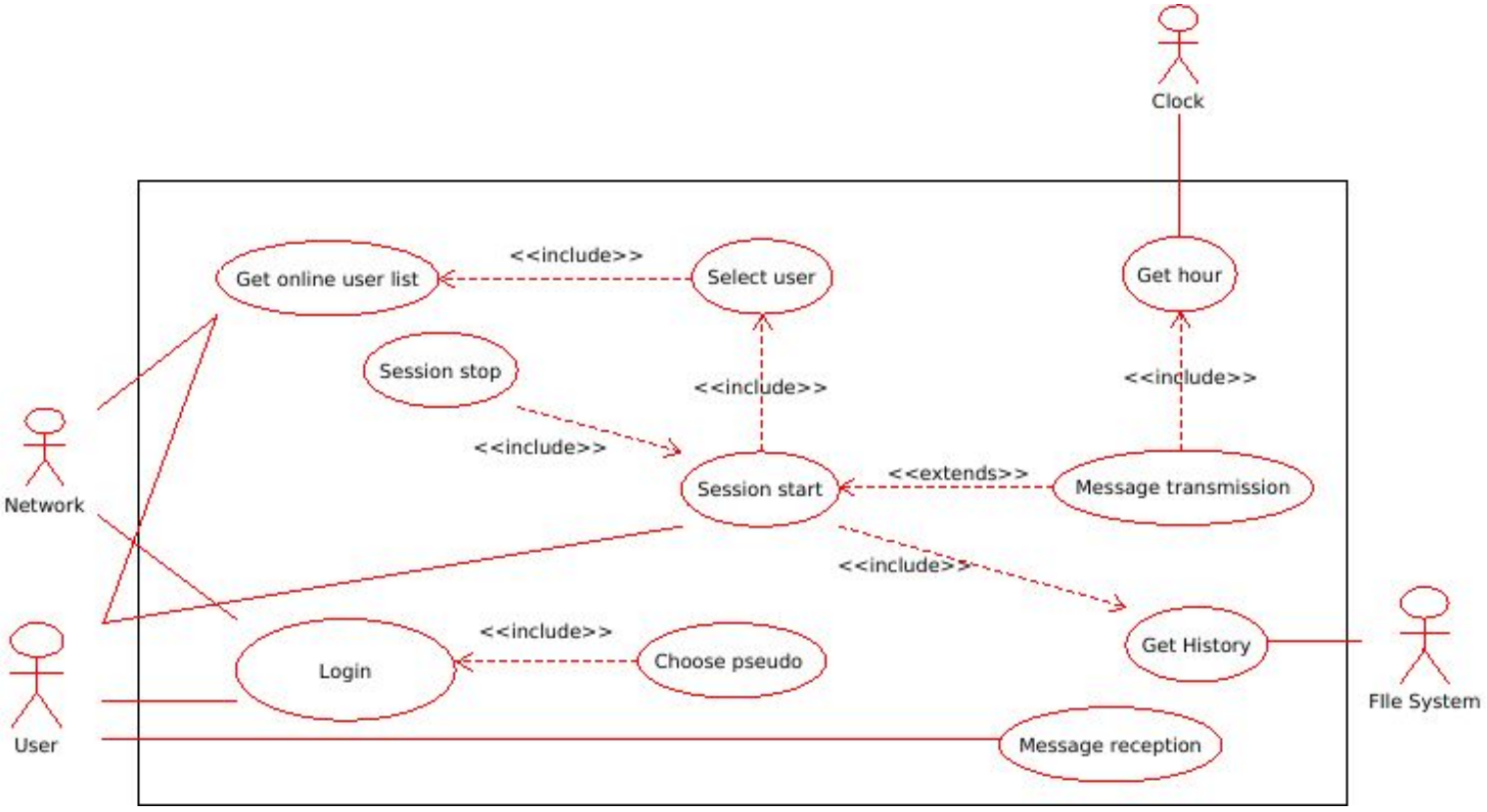
INSA Toulouse - 4IR I

Partie conception **2**
Diagramme des cas d'utilisations 2
Diagramme 2
Précision textuelle des différents cas d'utilisations 3
Diagrammes de séquence en boite noire 5
Diagramme de classes 8
Partie Programmation **9**
Choix pour l'envoi des messages 9
Choix pour l'historique 9
Fonctionnalités manquantes 9
Partie Installation - Déploiement / Utilisation **10**
Installation et Déploiement 10
Utilisation de l'application 11

Partie conception

Diagramme des cas d’utilisations

Diagramme



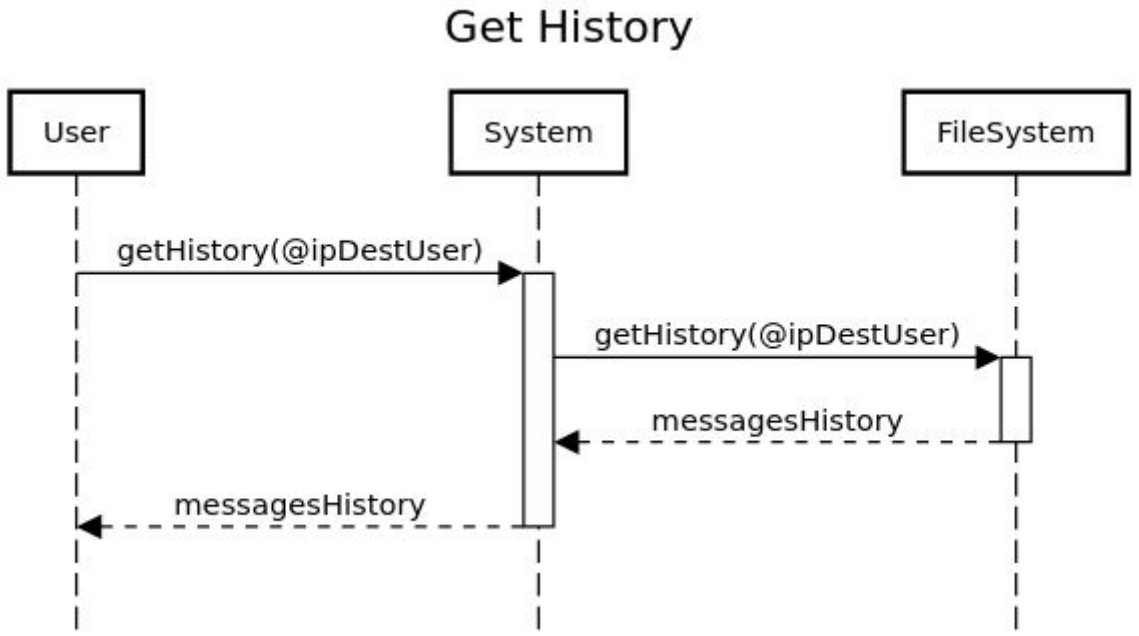
Précision textuelle des différents cas d'utilisations

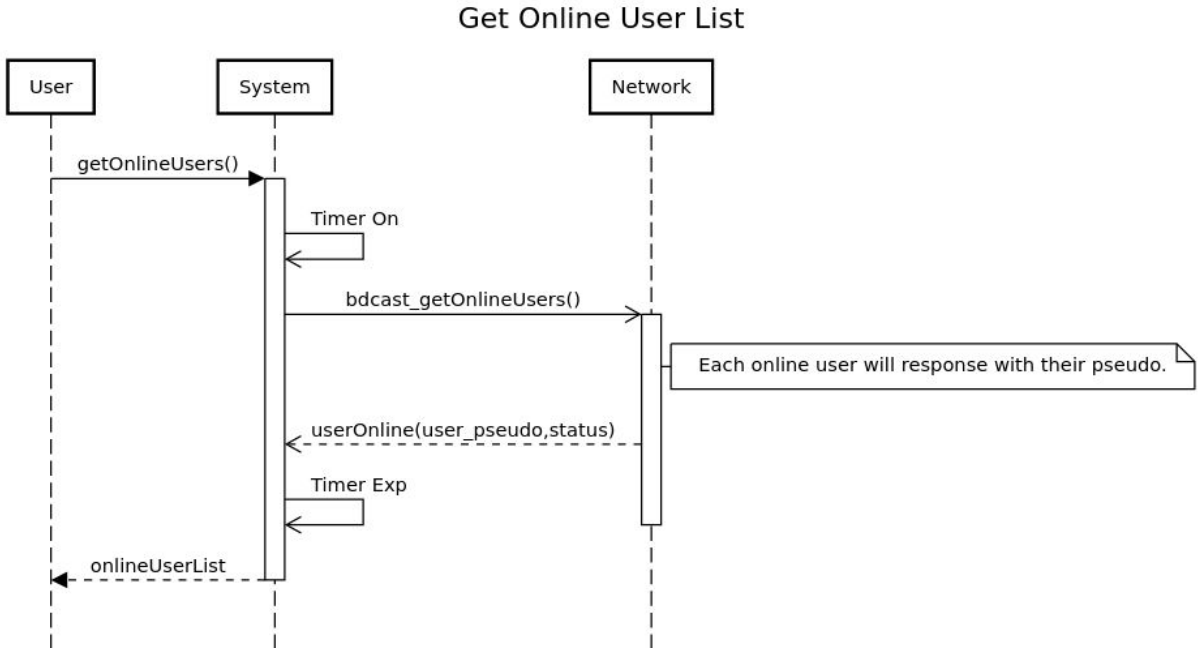
Use Case	Overview
	Pre-condition
	Post-condition

Login	Un utilisateur peut se connecter avec un pseudo.
	- L'utilisateur n'est pas connecté - Le pseudo selectionné par l'utilisateur n'est pas deja utilisé
	- L'utilisateur est connecté
Choose Pseudo	L'utilisateur doit choisir son pseudonyme.
	- Personne ne possède le pseudo voulu. - Le pseudo voulu par l'utilisateur est valide (pas vide).
	- L'utilisateur a un pseudonyme.
Message reception	Un message est recu
	- L'utilisateur est connecté
	- L'utilisateur recoit un message
Get Online User List	Un utilisateur peut récupérer l'ensemble des utilisateurs connectés
	- L'utilisateur est connecté
	- L'utilisateur est informé sur les différents utilisateurs en ligne.
Session Start	Ouvre une session avec un autre utilisateur
	- L'utilisateur est connecté - L'utilisateur a selectionné un autre utilisateur
	- La session est ouverte (une fenêtre de conversation).

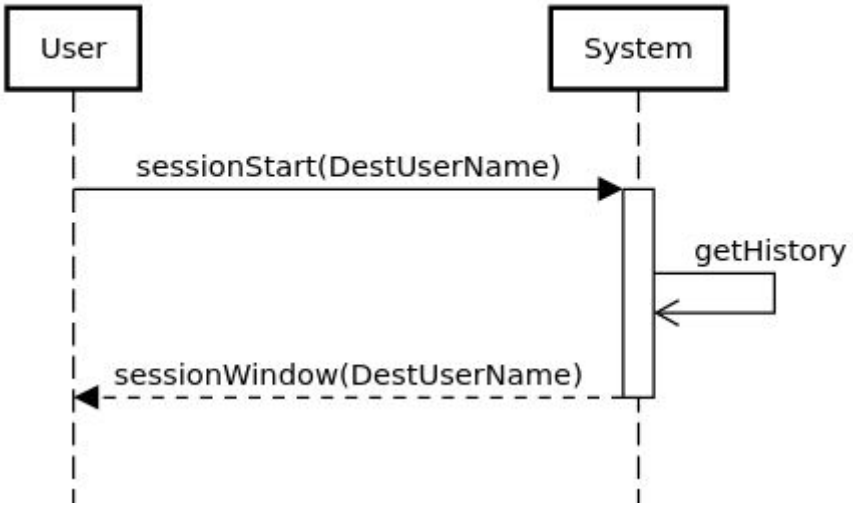
Get History	On récupère l'historique d'une ancienne session.
	- L'utilisateur est connecté. - L'utilisateur vient de demarrer une session. - L'utilisateur a deja utilisé le système de chat sur la même machine, avec le même login. - Une session entre les mêmes utilisateurs à deja existé (entre les deux mêmes adresses IP).
	- Les anciens messages apparaissent dans la session actuelle.
Session Stop	L'utilisateur peut mettre fin a la session.
	- L'utilisateur est connecté. - Une session a été ouverte.
	- La session est fermée.
Message Transmission	On peut envoyer un message
	- L'utilisateur doit etre connecté. - Une session doit être ouverte avec un autre utilisateur.
	- Un message est envoyé.
Get Hour	Les messages peuvent acceder à l'heure locale, via l'horloge systeme.
	- Un message est envoyé.
	- Le message est lié à l'heure locale.

Diagrammes de séquence en boite noire

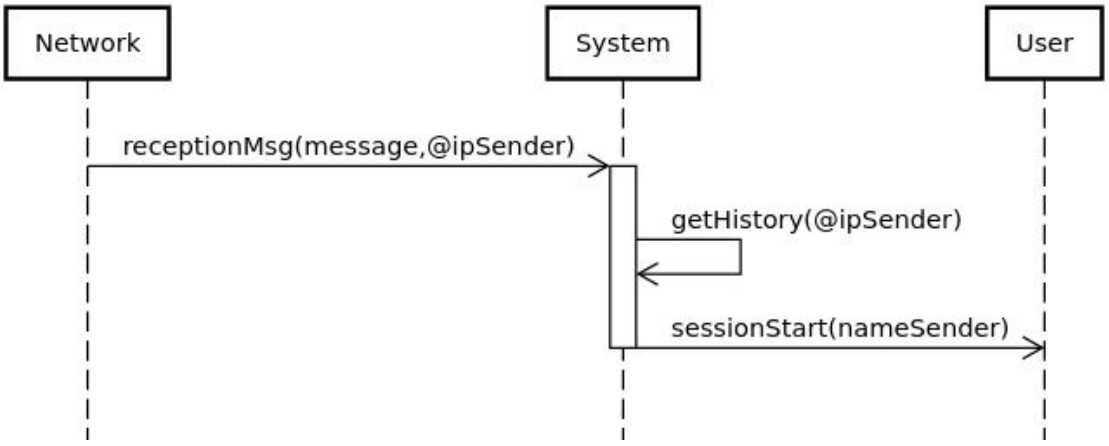




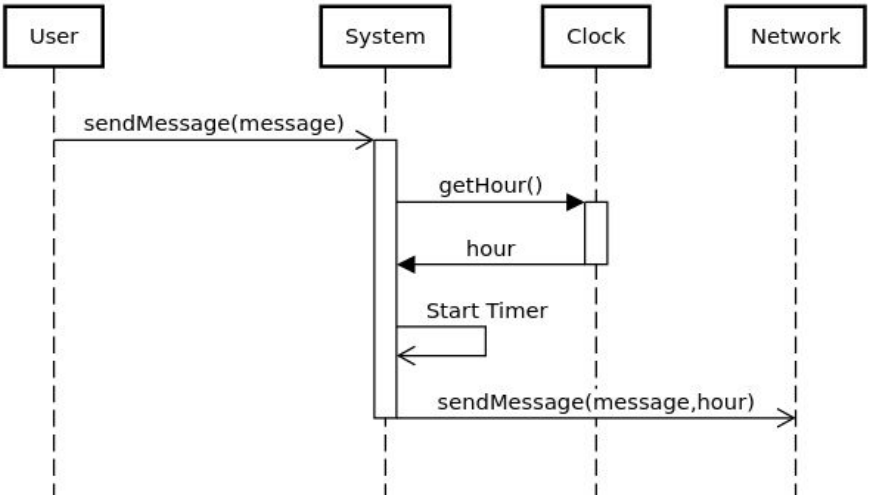
Session Start



Message Reception



Message Transmission



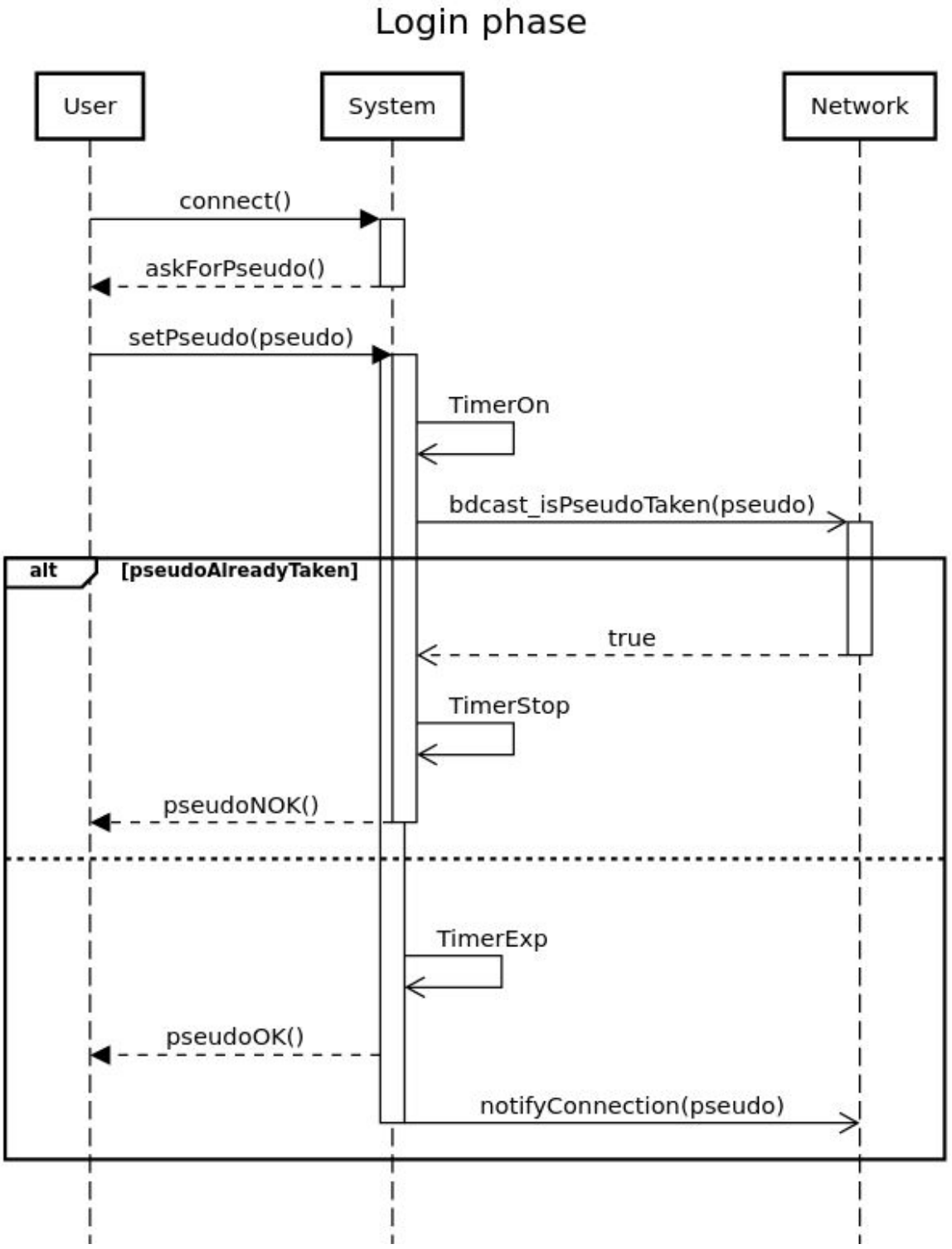
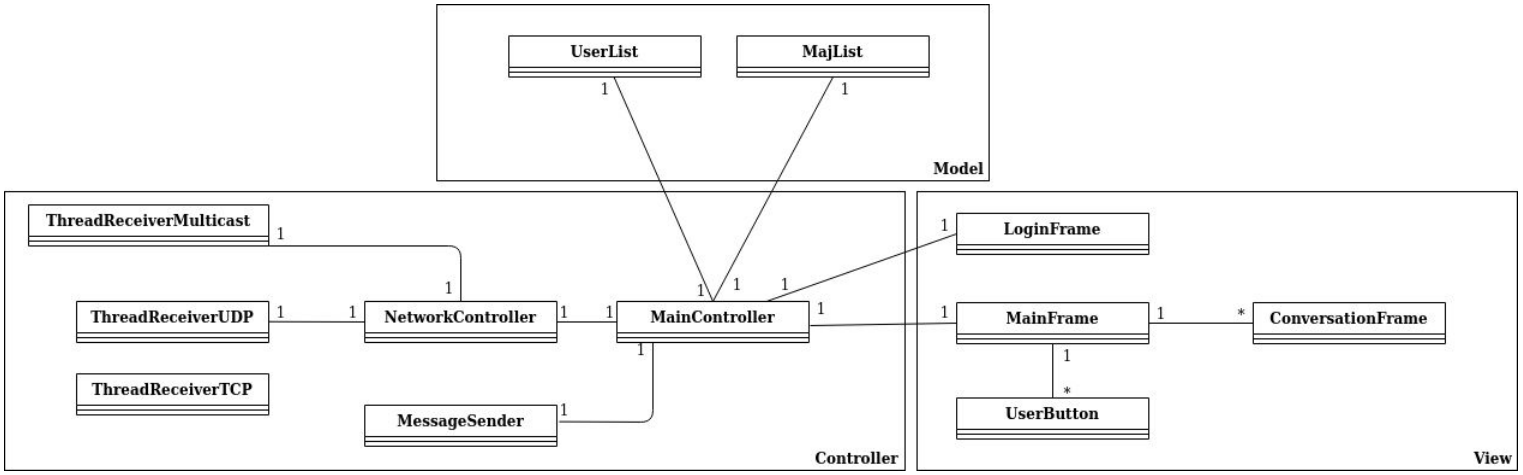

Login phase
User
System
Network
connect()
askForPseudo()
setPseudo(pseudo)
TimerOn
bdcast_isPseudoTaken(pseudo)
alt
[pseudoAlreadyTaken]
true
TimerStop
pseudoNOK()
TimerExp
pseudoOK()
notifyConnection(pseudo)

Diagramme de classes



Partie Programmation

Choix pour l'envoi des messages

L'envoi des messages entre deux utilisateurs se fait via le protocole UDP.

Pour les messages à envoyer à tous les utilisateurs (messages qui sont gérés par l'application et non par l'utilisateur lui-même, comme les messages d'identification), nous avons choisi, pour ne pas surcharger le réseau, d'utiliser le multicast. Lors du démarrage de l'application, celle-ci va s'abonner à un groupe multicast lié à l'application pour pouvoir envoyer et recevoir des messages liés à ce groupe (et donc aux autres instances de l'application, qui correspondent à d'autres utilisateurs).

Choix pour l'historique

Pour le chargement de l'historique, nous avons choisi de sauvegarder les conversations dans un fichier local. Nous supposons que chaque utilisateur a un ordinateur fixe (qu'il ne change pas d'ordinateur entre chaque session) et que celui-ci garde une adresse IP fixe également. Pour chaque message, qu'il soit reçu ou envoyé, nous créons (s'il n'existe pas) un document texte lié l'adresse IP de la personne distante, où nous enregistrons le contenu du message, la date et l'heure. Lors du lancement d'une session, nous recherchons dans un dossier spécifique s'il existe un fichier texte dont le nom est celui de l'adresse IP distante (Adresse IP de la personne que l'on souhaite contacter). Si cela est le cas, nous chargeons l'ensemble des messages ainsi que leur date et l'heure avec un code couleur permettant de différencier les messages reçu des messages émis.

Fonctionnalités manquantes

Notre application ne dispose pas de certaines fonctionnalités comme :

- L'envoi et la réception de documents (fichiers).
- L'envoi et la réception d'images.
- Les statuts utilisateurs (un utilisateur est forcement en ligne lorsque l'application est ouverte et hors-ligne lorsqu'elle est eteinte).
- La connexion avec des utilisateurs distants (via servlets).
- La possibilité de changer de pseudonyme (pour celle, il faut ré-ouvrir l'application).

Partie Installation - Déploiement / Utilisation

Installation et Déploiement

Pour déployer notre installation, il faut cloner le dépôt git du projet :
https://github.com/KilianDesportes/OO_distributedChatSystem
via la commande :

```
$ git clone https://github.com/KilianDesportes/OO_distributedChatSystem
```

Une fois le repertoire cloné, il faut se déplacer depuis celui-ci dans les sources du projet pour pouvoir compiler celui-ci :

```
$ cd java_project/src/
```

La compilation se base ensuite sur **ant**, qui doit être installé sur la machine :

```
$ ant build
```

Une fois le build réalisé, les fichiers exécutables sont disponibles et l'application se lance avec la commande :

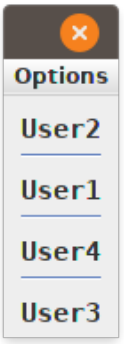
```
$ java Main
```

Pour effacer les fichiers liés à la compilation, vous pouvez exécuter la commande :

```
$ ant clean
```

Utilisation de l’application

L’application est simpliste, une étape de connexion est nécessaire au lancement de l’application, qui va vous permettre de choisir un pseudonyme.
Si le pseudonyme voulu est déjà utilisé, vous en serrez notifié et vous ne pourrez pas utiliser celui-ci.
Une fois la connexion réussie, une liste des utilisateurs connectés actuellement apparaîtra.



Un clic sur un des noms déclenchera l’ouverte d’une fenêtre de chat qui sera, s'il y a déjà eu une discussion avec cet utilisateur, remplie des anciens messages.

On peut ensuite, via cette fenêtre, envoyer un message en le saisissant à l’emplacement prévu pour, puis appuyer sur envoyer.

Lorsque l’on reçoit un message, une fenêtre de conversation va également s’ouvrir directement avec l’utilisateur ayant envoyé un message.