UNIVERSITÉ NATIONALE DU VIETNAM À HANOÏ

INSTITUT FRANCOPHONE INTERNATIONAL

CONCEPTION ET ARCHITECTURE DES RÉSEAUX

RAPPORT DE PROJET APPLICATIF

THÈME : Messagerie instantanée interactive sur le réseau local

Juin 2019

Étudiants :

Cleg Peter OVIL
Odelet VALCIN
Mike Arley MORIN
Afi Elolo Gisèle DEKPE

Encadreur : **M. Nguyen Hong Quang**

Année Académique : 2019 - 2020

Table des matières

Table des figures

Introduction

Dans son programme de formation, l'Institut Francophone International (IFI) intègre au cours du cycle Master de Recherche, un module intitulé **Conception et Architecture des Réseaux**. Ce module est destiné à l'approfondissement des connaissances des étudiants sur le fonctionnement des réseaux informatiques. Grâce à ces derniers et l'évolution du monde de la technologie, la communication qui est le point important dans tout secteur organisationnel devient de plus en plus aisée. Ceci est possible grâce à l'existence des applications ou IHM (Interface Homme Machine) qui offrent des moyens plus ou moins efficaces ; selon les types des réseaux (local, étendu, etc...), les types d'organisations et cibles qui les utilisent. Nous remarquons une augmentation du nombre ces applications, au regard des avantages qu'elles offrent. Dans le but d'appliquer nos connaissances de développement au domaine des réseaux informatiques et de la communication, il nous a été demandé de concevoir et d'implémenter une messagerie instantanée interactive sur un réseau local. Le document présent fait office du rapport de notre projet. Dans un premier temps, nous allons décrire le cahier de charge, la phase de conception puis parler de comment le protocole applicatif qui a été implémenté. Il est également présenté dans ce rapport, les résultats des tests effectués l'application conçue.

1 Cahier de charges

Le cahier de charges de notre projet de messagerie instantanée se décline dans les lignes suivantes. D'un point de vue global, il s'agit de concevoir une application permettant à deux (02) ou plusieurs individus se trouvant sur leur poste de pouvoir s'échanger des messages et/ou des fichiers. Spécifiquement, il est question dans un premier temps de :

— concevoir notre propre protocole de messagerie,

— implémenter une application dont le fonctionnement sera basé sur notre protocole.

Les tâches liées à ces objectifs à atteindre sont les suivantes. D'un coté, nous avons :

— la rédaction de notre protocole de communication

— la description du format des messages

— le choix du protocole de transport des données,

d'autre part, nous aurons à implémenter l'application de messagerie de telle sorte que :

— les postes des utilisateurs se trouvent sur un même réseau local,

— les utilisateurs ne conservent pas leurs adresses afin de pouvoir se connecter à partir de postes différents,

— il n'existe pas de serveur central pour gérer les utilisateurs et les sessions d'échange ; et que tout se fasse de façon distribuée selon le protocole que nous allons concevoir,

— la communication soit faite par l'envoi de messages asynchrones sous forme de chaînes de caractères de longueurs différentes ; avec à la base, un seul groupe de discussion,

— chaque utilisateur puisse s'enregistrer sur le réseau lors de sa connexion en fournissant un identifiant (pseudonyme),

— chaque poste maintienne une liste des utilisateurs connectés au système de messagerie (gestion dynamique des connexions/déconnexions),

— la gestion des états de l'utilisateur (connecté, absent, occupé, etc.) soit possible.

Outre ces fonctionnalités, il pourrait être mis en supplément :

— le chat en conférence et par plusieurs groupes de discussions,

— le partage de fichier et l'envoie d'émoticones, etc...

2 Conception du projet

Dans cette section nous allons parler de la conception du projet, en définissant d'abord les cas d'utilisation de système. Nous allons ensuite décrire notre protocole de communication puis du protocole de transport que nous avons choisi pour la réalisation de notre projet. Nous parlerons aussi de l'architecture adoptée et du format des messages.

2.1 Les cas d'utilisation du système

Les cas d'utilisation de notre système sont la connexion, le chat en groupe et la déconnexion. Le diagramme de cas d'utilisation se décline comme suit :

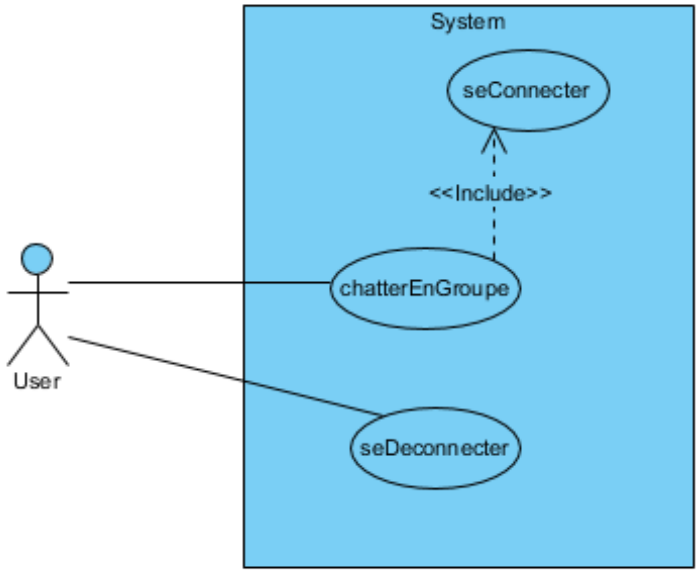

FIGURE 1 – Diagramme de cas d'utilisation du système

Un utilisateur peut se connecter, chatter et se déconnecter. La contrainte **"include"** veut dire que pour chatter en groupe, il faut impérativement que l'utilisateur se connecte.

2.2 Les protocoles

Nous ne pouvons pas parler de "réseau" et de "communication" sans parler de "protocole". Un protocole en informatique est un ensemble de règles qui régissent les échanges de données ou le comportement collectif en réseaux ou d'objets connectés. Il nous est demandé de concevoir notre protocole de messagerie.

2.2.1 Rédaction de notre protocole de communication

L'utilisateur se connecte au réseau local puis au système grâce à l'adresse multicast (224.0.0.1). Il renseigne les paramètres de connexion (dans notre cas, le login ou nom d'utilisateur est suffisant). Il est maintenant apte à échanger avec les autres utilisateurs connectés. Lorsqu'il n'y a qu'un seul utilisateur, cela devient une communication entre deux personnes comme si c'était en privé. Dans le cas contraire, d'autres utilisateurs rejoignent le système et on a une discussion en groupe. Les utilisateurs peuvent choisir de s'envoyer des fichiers, en plus des messages textuels. En outre, il nous faut choisir un protocole qui va assurer le transport des données d'un point à l'autre. Pour ce faire, nous avons choisit le protocole UDP.

Notons que l'adresse multicast est déjà paramétrée dans le code source de l'application et qu'il suffit d'être dans le même réseau local et de pouvoir exécuter l'exécutable de l'application (le fichier.java) pour avoir accès au chat.

2.2.2 Les raisons du choix du protocole de transport UDP

UDP qui signifie **User Datagram Protocol** est l'un des protocoles de la couche transport du modèle OSI. Il est rapide dans la transmission des data-grammes des utilisateurs, c'est ce qu'il donne comme grand avantage. Néanmoins, il ne favorise ni le handshaking ni le contrôle d'erreur ; pour cela il est qualifié de protocole peut fiable et "mode-non-connecté". Pour notre projet, le port 2244 du protocole UDP est celui utilisé. UDP sur le port 2244 pense que la vérification et la correction des erreurs ne sont ni nécessaires ni effectuées dans l'application, ce qui évite la surcharge d'un tel traitement au niveau de l'interface réseau.

2.3 Architecture utilisée

Il nous a été demandé de concevoir l'application de telle sorte qu'il n'existe pas de serveur de stockage de données. Du coup, nous avons opté pour une architecture **Peer-To-Peer** (P2P). Chaque poste client jour le rôle d'émetteur et de récepteur. Ci-dessous un schéma qui illustre de la plus simple des manières, l'architecture Peer-To-Peer.

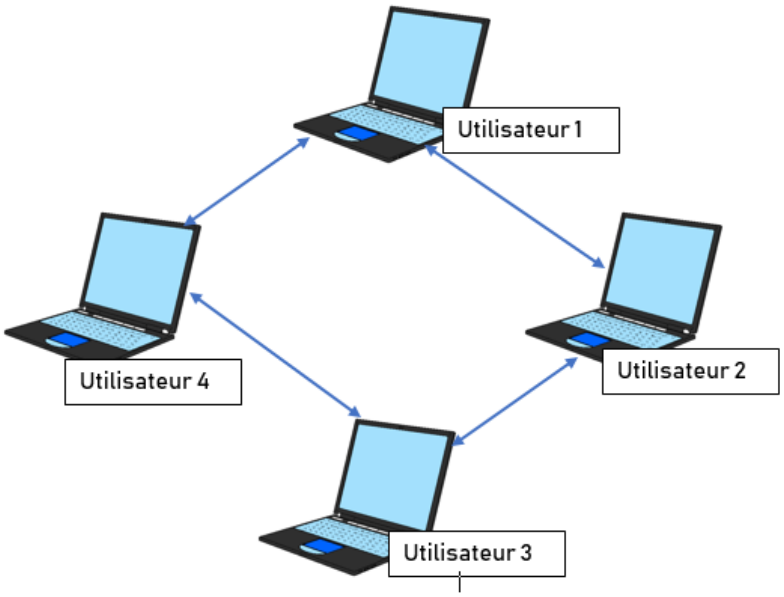

FIGURE 2 – Illustration d'une architecture Peer-To-Peer

2.4 Le format des messages

La figure ci-dessous représente le format sur lequel va se baser les messages envoyés.

Code	Name	Content	Extension

FIGURE 3 – Format d'un message

Description :

— **Code** : chaque événement qui se déroule au sein du système a un code particulier. Nous avons fixé la longueur de ce champ sur trois (03) caractères.

Evénements/ Messages	**Code**
chat	600
déconnexion	700
connexion	800
fichier	100

FIGURE 4 – Code des messages / évènements

— **Name** : c'est le nom de l'utilisateur qui envoie le message.

— **Content** : c'est le contenu du message.

— **Extension** : ceci désigne l'extension du message envoyé, important pour les fichiers.

3 Fonctionnalités du système

Dans cette section, il sera question de décrire les fonctionnalités du système mis en place.

3.1 Lancer le programme de messagerie instantanée

Pour exécuter le programme, il faut faire un **Run** du fichier **client1.java**.

3.2 Se connecter

Comme pour tout système, il faut pouvoir se connecter afin de pouvoir jouir des fonctionnalités offertes. Pour ce faire, l'utilisateur, dès qu'il a l'interface de l'application, il entre son nom ou login. Son statut passe à "online" et la liste des utilisateurs connectés est mise à jour ; puis renvoyée pour tenir informés les autres utilisateurs. Le diagramme de séquence ci-dessous pour mieux expliquer.

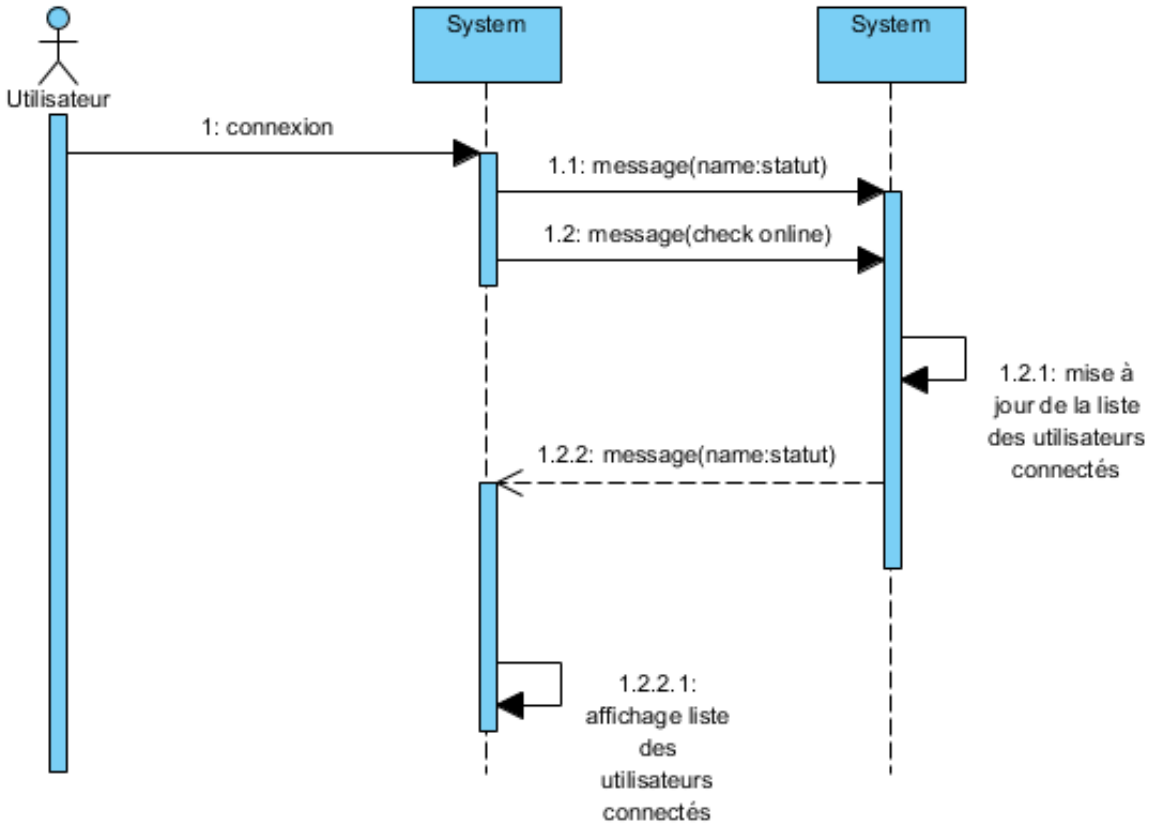

FIGURE 5 – Connexion au système de messagerie

3.3 Échanger des messages

L'échange des messages se fait selon le processus suivant : lorsqu'un message est envoyé, il comporte le code, le nom de l'utilisateur qui envoie, le contenu du message et l'extension (surtout pour les cas des fichiers). Le code est vérifié par le programme. Si le code est égale à 600, on a juste un message textuel, il est simplement affiché à qui de droit ; sinon (i.e. le code est égale à 100) on a un message comportant un fichier. Ce fichier est converti en base 64, la vérification de l'extension est faite ensuite puis le fichier est converti de nouveau de la base 64 en l'extension approprié. Le diagramme de séquence ci-dessous pour mieux expliquer.

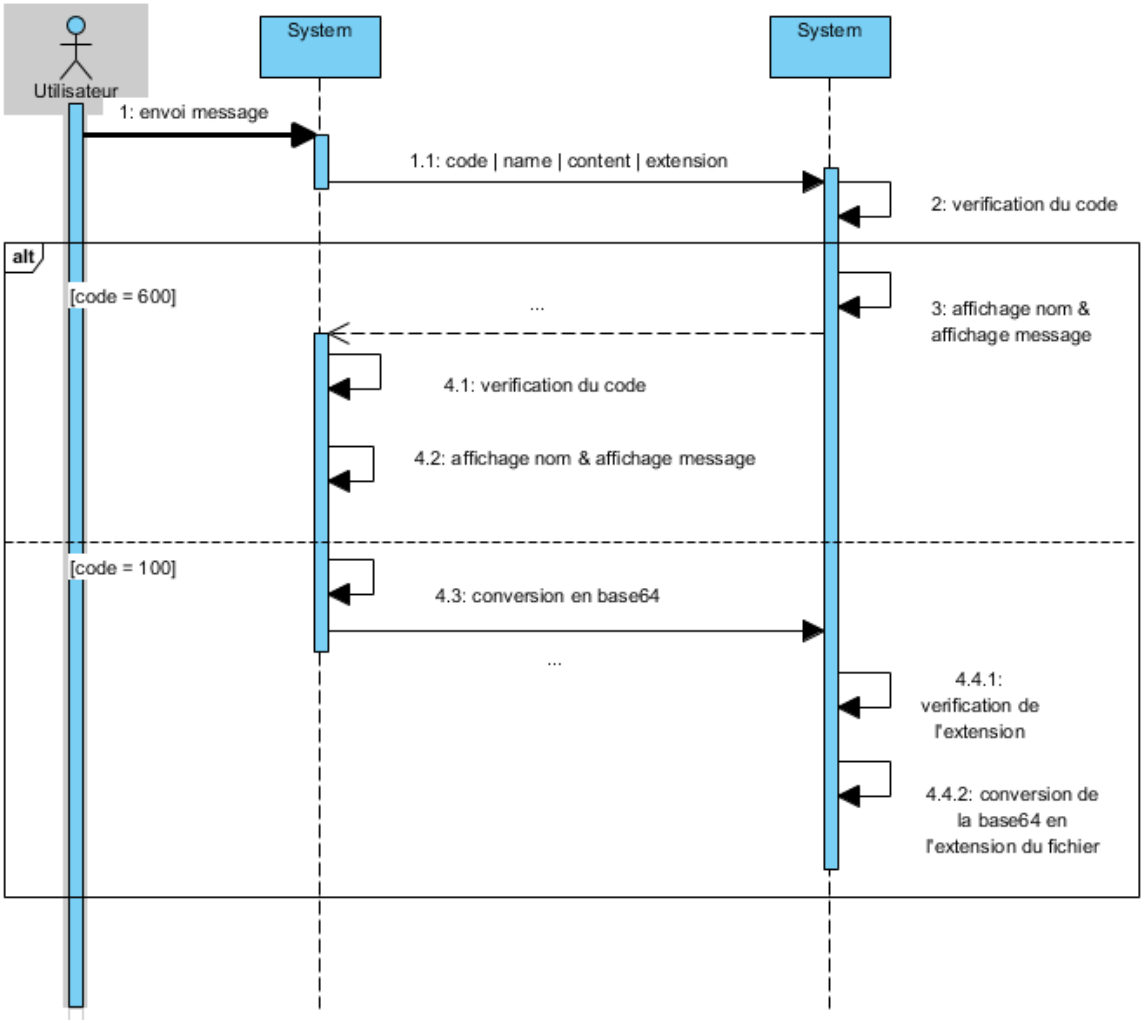

FIGURE 6 – Échange de messages

3.4 Se déconnecter

Pour se déconnecter, l'utilisateur clique sur "déconnexion" et la liste des utilisateurs connectés est mise à jour. Le diagramme de séquence ci-dessous pour mieux expliquer.

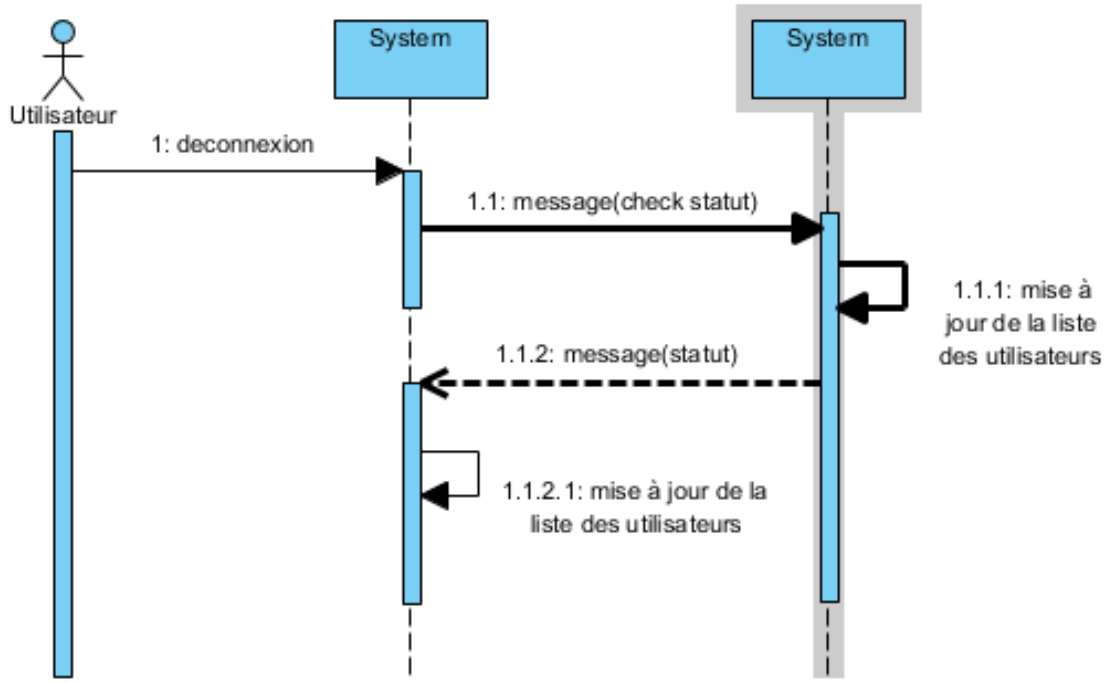

FIGURE 7 – Déconnexion

La prochaine étape est celle de l'implémentation, nous aurons à parler des outils utilisés pour l'élaboration de notre système de messagerie.

4 Implémentation

Pour la mise en place de notre application, nous avons utilisé des outils et langage de programmation tels que :

— **Le langage Java** pour la programmation : le choix de ce langage est du au fait qu'il est un langage système.

— **Le langage UML** pour l'analyse et la conception du système.

La conception a été faite sur des sytèmes tels que Mac OS et Linux (Ubuntu).

5 Tests

Dans cette partie, nous montrons les captures des tests que nous avons effectués et les résultats obtenus.

5.1 L'interface de connexion où un utilisateur met son login

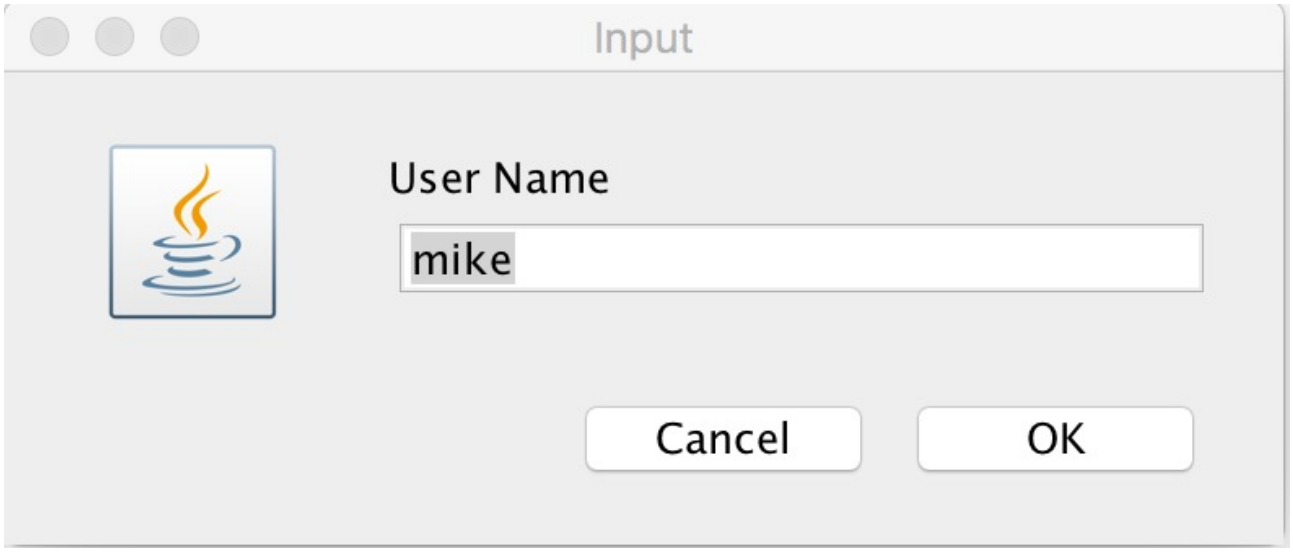

FIGURE 8 – Connexion au chat

5.2 L'interface d'échange de messages

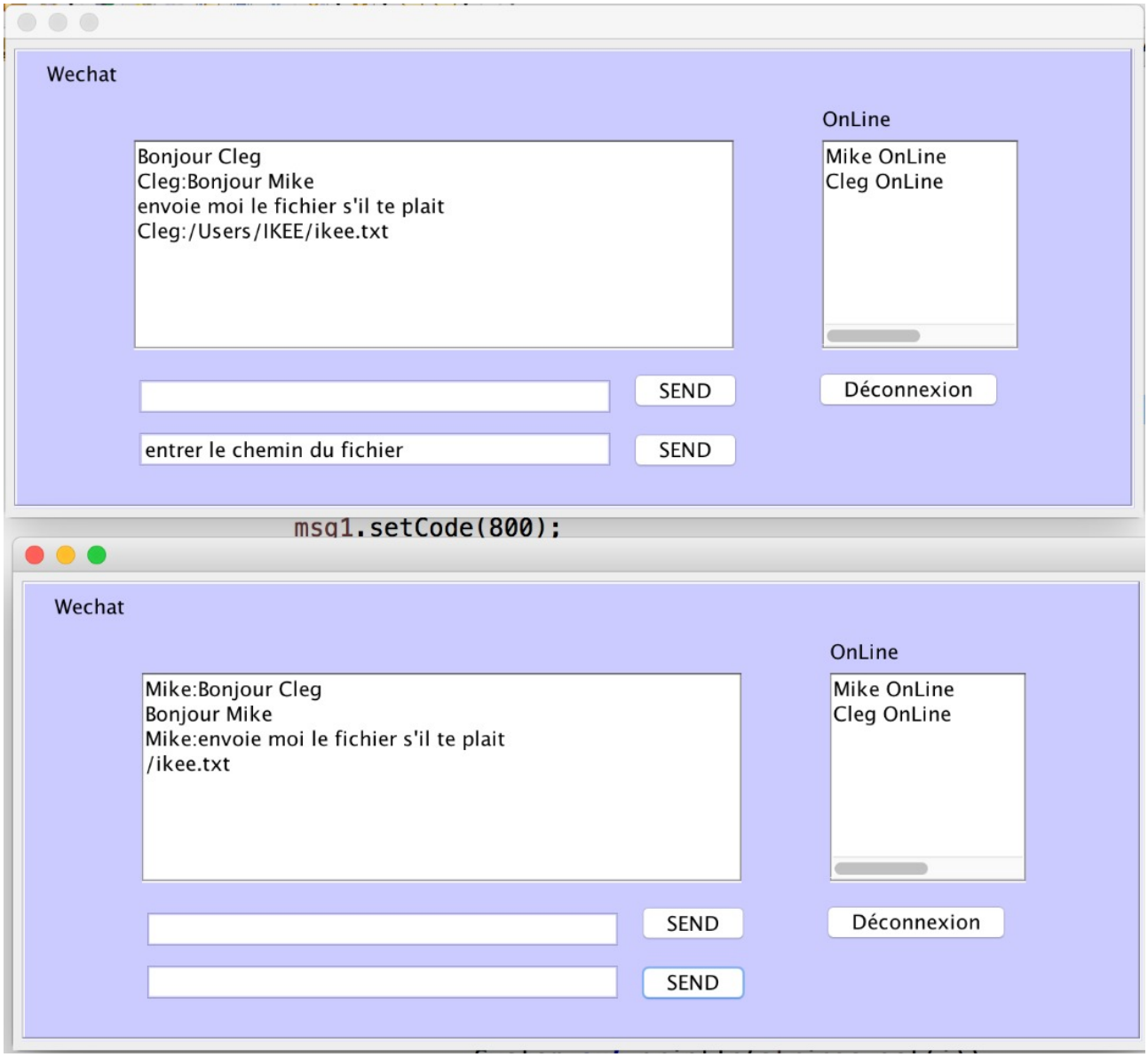

FIGURE 9 – Connexion au chat

5.3 L'interface montrant la déconnexion d'un utilisateur du chat

FIGURE 10 – Connexion au chat

Conclusion & Perspectives

Ce document recense globalement nos travaux sur le thème de conception d'un chat applicatif instantané. Du cahier de charge, nous avons pu déceler les aspects techniques et sécuritaires du travail qui nous est demandé, en d'autres termes la phase de conception. A partir des éléments définis dans cette phase, nous avons implémenté en Java, l'application de messagerie instantanée tout en tenant compte des spécifications du sujet. La phase de test nous a permis de vérifier l'efficacité de notre système. En général, ce projet nous a permis, non seulement de mettre en pratique nos connaissances en programmation réseaux mais aussi d'acquérir beaucoup plus de notions en ce qui concerne la conception des logiciels de messagerie instantanée sur un réseau local. Malgré les difficultés rencontrées, nous avons pu concevoir et implémenter un protocole de communication interactif permettant à des individus connectés depuis leurs ordinateurs (dans le même réseau local) de communiquer en toute simplicité. Néanmoins, nous comptons y apporter des améliorations telles que :

— l'utilisation du protocole de transport TCP surtout pour garantir le transfert effectif des fichiers envoyés,

— l'archivage des différents messages échangés par les utilisateurs ;

ceci afin que cette solution soit encore plus exploitable.

Références

— Cours de Conception et Architecture des Réseaux, M. Nguyen Hong Quang

— https ://openclassrooms.com/fr/courses/2654601-java-et-la-programmation-reseau/2668710-les-sockets-cote-client

— dspace.univ-tlemcen.dz/bitstream/112/6825/1/application-client-serveur.pdf