RAPPORT COO/POO

SYSTÈME DE CLAVARDAGE DISTRIBUE INTERACTIF MULTI-UTILISATEUR TEMPS RÉEL

Année 2019-2020 - IR SI

Par Anaïs HONNET et Clémence CALAS

Introduction :

Nous avons une classe AppClavardage qui ne fait que lancer notre interface graphique principale MaFenetre dans laquelle nous pouvons choisir un pseudo, le changer et démarrer une session de clavardage avec un des utilisateur présent dans la liste des utilisateurs actifs.
Une nouvelle interface graphique se lance pour chaque session de clavardage à la création des serveurs TCP dans laquelle nous pouvons envoyer des messages, envoyer des photos et dans laquelle l'historique de toutes les anciennes conversations horodatées s'affiche.

Conception :

Diagrammes UML :

A travers l'ensemble de nos diagrammes UML nous pouvons avoir une vision globale du cahier des charges, du produit attendu et de notre démarche de réalisation du projet.
Chaque diagramme apporte une vision différente. Le comportement de notre application va plutôt être décrit dans le diagramme de cas d'utilisation (d'une manière plus globale), la machine à état ainsi que les diagrammes de séquence (plus précisément) alors que le diagramme de classe va nous montrer la structure générale du projet.

Nous n'avons pas inclu les diagrammes au rapport (par un souci de clarté et lisibilité) mais ils sont disponibles en annexe. Il y a cinq diagrammes de séquence différents (connexion, choix/changement de pseudo, démarrage d'une conversation, envoi d'un message et envoi d'une photo). Ils représentent assez précisément notre approche et permettent de comprendre plus facilement les interactions entre nos différentes classes.

Motivations des choix :

Un serveur permettant de communiquer en broadcast des messages de contrôle :

Dès notre connexion le serveur UDP se met en écoute (et se ferme à la déconnexion). Celui-ci permet de gérer les messages en broadcast et également les requêtes de lancement d'une conversation de la part d'un utilisateur.

En effet, nous avons mis en place une classe MessageControle qui crée une trame particulière, facilement reconnaissable en réception. Elle relaie des informations importantes : un changement de pseudo, l'arrivée ou le départ d'un utilisateur, ainsi que la demande de création d'une session. L'architecture de leur trame permet une création et un traitement plus rapide : en effet nous fonctionnons par mot-clés (connected, disconnected, pseudo, session) chacun engendrant un traitement différent en réception. Également, nous ajoutons à ce mot-clé le nouveau pseudo pour le transmettre aux autres utilisateurs dans le message en broadcast.

Nous avons décidé d'utiliser un format UDP pour tous nos messages de broadcast. En effet, l'absence de délai de retransmission et son efficacité reconnue pour les communications unidirectionnelles ont orientées notre choix.
Nous n'utilisons le broadcast que dans trois cas :

- Arrivée sur le réseau d'un nouvel utilisateur ➜ Il broadcast sa connection, les personnes présentes sur le réseau ajoutent ce nouvel utilisateur à leur liste de personnes actives et lui transmettent leur pseudo (tout cela par message de contrôle).
- Choix de pseudo pour un nouvel utilisateur ou changement de pseudo ➜ Lors de sa connection, l'utilisateur récupère le pseudo de tous les utilisateurs actifs. La liste des personnes connectées est mise à jour en temps réel pour chaque nouvelle connection. Cela permet donc d'assurer l'unicité des pseudos.
- Déconnection d'un utilisateur ➜ L'utilisateur envoie en broadcast qu'il se déconnecte afin que les autres utilisateurs suppriment son pseudo de leur liste de personnes actives.

Lorsque nous choisissons un pseudo avec qui communiquer, nous envoyons en UDP à l'utilisateur que nous souhaitons démarrer une session de clavardage et celui-ci ouvre un ServerTCP en écoute.

Une communication en TCP entre deux utilisateurs :

Lorsqu'un utilisateur va démarrer une session de clavardage avec un destinataire, les deux vont instancier la classe ServerTCP, un en expéditeur et l'autre en écoute.
Ils vont pouvoir donc échanger leurs messages de manière indépendante et la création de plusieurs sessions simultanées est possible grâce à une implémentation dynamique des numéros de port.

Les messages entrés directement sur l'interface graphique vont être transmis sur le réseau et enregistrés dans l'historique affiché en temps réel.

L'échange d'image est également possible via le bouton "Envoyer une image" qui va permettre, tout comme pour les messages, d'instancier des classes ServerTCPImage et ClientTCPImage qui vont traiter l'échange de l'image. Celle-ci est choisi directement par l'utilisateur dans ses fichiers et sera affichée sur l'écran du destinataire.

Un dossier de sauvegarde des historiques :

A chaque lancement de session de clavardage, un dossier Historique est créé dans la machine de l'utilisateur (s'il n'est pas déjà créé) sur son répertoire courant avec un fichier correspondant au destinataire. Nous avons choisi comme nom de fichier de mettre l'adresse IP (en nombre décimal pour éviter les confusions avec les points) afin de récupérer les conversations même si le destinataire a changé de pseudo entre temps. Les pseudos sont précisés à l'intérieur devant chaque message, ainsi que la date et l'heure du message.

Nous avons choisi d'implémenter cette solution car elle était plus rapide à effectuer et l'historique est accessible même lorsque l'application de clavardage n'est pas lancée.

Procédures de tests et de validation :

Afin de tester notre application de clavardage, nous le faisions par étapes : nous implémentions classe par classe les différentes parties du projet (par exemple envoi en broadcast, communication TCP, envoi d'images…). Cela nous permettait de débugger plus facilement et une fois sûres de leur bon fonctionnement, nous mettions en commun dans notre projet Clavardage.

Nous avons fait en sorte de répondre de façon précise au cahier des charges. En effet, nous pouvons choisir un pseudonyme qui est communiqué au système (accompagné de notre adresse IP). Ce pseudonyme pourra être changé à n'importe quel moment. La liste des utilisateurs actifs sur le réseau local est disponible pour tous les utilisateurs connectés et est actualisée continuellement (notamment lors des connexions, déconnexions et changement de pseudos). L'utilisateur peut démarrer une session de clavardage avec un destinataire qu'il choisira dans cette liste. Il est impossible que deux personnes choisissent le même pseudo. Tous les messages sont enregistrés dans l'historique avec la date et l'historique est affiché en temps réel pour les deux utilisateurs. L'un ou l'autre des utilisateur peut mettre fin à la session de clavardage de manière unilatérale sans consulter son correspondant. Dans ce cas-là, la fenêtre de conversation se ferme. Ils peuvent également réduire l'agent.

Le changement de pseudonyme, l'envoi/réception de messages, l'échec du choix de pseudonyme et le temps d'apparition des utilisateurs actifs sont instantanés (tous inférieurs à une seconde) et répondent donc au cahier des charges.

Exécution :

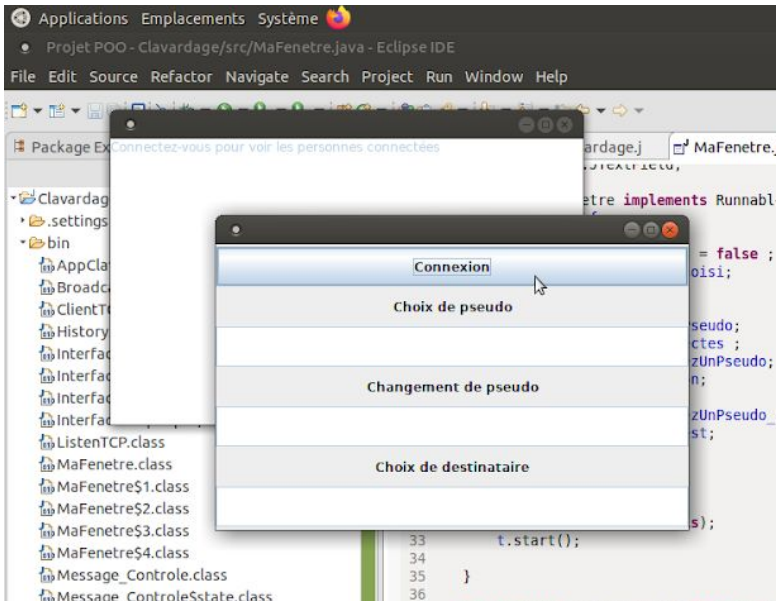
Guide d'administrateur :

Nous avons défini les ports manuellement dans notre application.
Nous avons choisi pour le serveur UDP et le Broadcast de les mettre sur le port 3000. Le socket de la session de clavardage est mis sur le port 4500.
Pour les différents serveurs TCP, sachant que nous pouvons communiquer avec plusieurs personnes en même temps, nous choisissons un port (le premier libre) que nous communiquons à notre destinataire lors de la fonction generateConnection dans la classe SessionCla. De cette façon, nous sommes sûr que le port et libre et que notre destinataire va bien se connecter sur ce même port.
Afin d'envoyer les images, le serveur et client Images se lancent sur le port 6066.

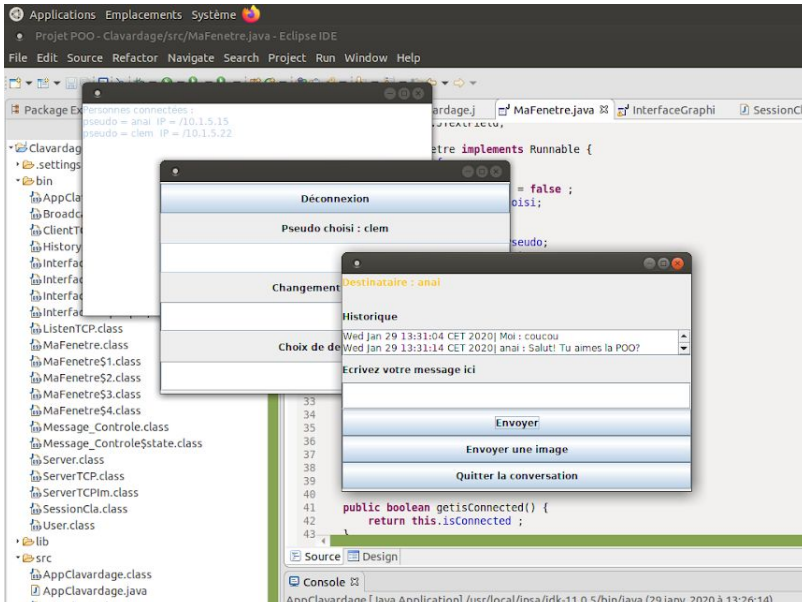
Afin de lancer l'application sous Ubuntu, le projet peut être lancé via eclipse et sinon, il suffit d'ouvrir un terminal et d'exécuter la commande "java -jar Projet_Cla.jar".
Sous Windows, il suffit d'effectuer un double-clic sur Projet_Cla.jar.

Manuel d'utilisation du produit :

Au lancement de l'application, une fenêtre principale s'ouvre, ainsi qu'une fenêtre où apparaît la liste des utilisateurs connectés.
Il faut alors appuyer sur le bouton "Connexion" et choisir un pseudo que nous pourrons changer à n'importe quel moment. On tape ensuite le pseudo du destinataire avec qui nous voulons démarrer une session que nous choisissons dans la liste.



Nous pouvons ensuite écrire un message et l'envoyer en cliquant sur "Envoyer".
Nous pouvons également envoyer des images que nous sélectionnons dans les fichiers en appuyant sur "Envoyer une image".
Lorsque la session de clavardage s'ouvre, l'historique contient tous les anciens messages envoyés avec leur date et il le met à jour et le réaffiche en temps réel lors de l'échange de nouveaux messages.



Il y a bien-sûr possibilité de communiquer avec plusieurs personnes simultanément et leurs pseudos s'affichent bien dans la fenêtre à chaque nouvelle connexion.