Systèmes Distribués & Systèmes d'Exploitation

L3 Informatique & L3 MI-Info

Projet Terminal

Léa Brunschwig <lea.brunschwig@univ-pau.fr>

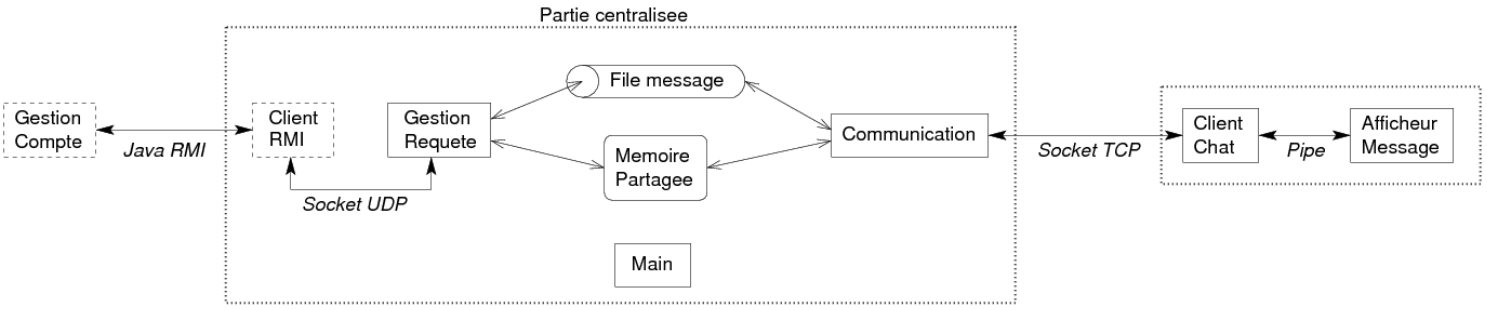
Ce document décrit le sujet du projet commun entre l'UE de systèmes d'exploitation et l'UE de systèmes distribués. Il est à faire en binôme. Il sera rendu un rapport comportant deux parties correspondant à chaque matière suivi d'une évaluation sur machine.

Description du sujet

Ce projet a pour but d'implémenter un outil de chat permettant à plusieurs utilisateurs de dialoguer.

Le principe général de l'application est qu'un utilisateur dispose d'un compte (*pseudo* et *mot de passe*) lui permettant de se connecter au système de chat. Une fois connecté, il peut envoyer un message à tous les utilisateurs connectés. Tout message envoyé est reçu par tous les utilisateurs connectés et affiché dans l'interface de l'utilisateur.

Architecture générale



La figure ci-dessus présente l'architecture générale de l'application à réaliser. Les éléments représentés dans des rectangles en tirets sont à implémenter en Java (`Gestion Compte` et `Client RMI`). Tous les autres le seront en C.

Client Chat et Afficheur Message

Ces deux éléments gèrent l'interaction avec l'utilisateur. Le *Client Chat* permet notamment à l'utilisateur d'envoyer des messages à tous les autres utilisateurs, une fois connecté au système. C'est le service par défaut qu'il offre à l'utilisateur. En plus de ce service, il offre également les services suivants :

1. Se connecter au système avec un pseudo et un mot de passe
2. Se déconnecter du système
3. Afficher la liste de tous les utilisateurs présents
4. Créer un nouveau compte
5. Supprimer un compte existant
6. Quitter le client

Comme l'utilisateur va recevoir régulièrement des messages et pour ne pas perturber l'interaction avec l'utilisateur, l'affichage de ces messages se fera dans un autre terminal. Ce terminal devra être lancé en même temps que le *Chat Client* et exécutera le processus *Afficheur Message*. Son seul but est d'afficher les messages qui seront reçus par le *Chat Client*. Il communiquera pour cela avec lui via un pipe (*nommé* ou *anonyme*).

Le *Chat Client* communiquera avec le processus *Communication* de la partie centralisée via des sockets TCP.

Gestion Compte

Cet élément gère la liste des comptes des utilisateurs. Un compte contient un pseudo et un mot de passe, tous deux sous forme de chaîne. Il ne peut exister deux comptes différents avec le même pseudo. La liste des comptes est persistante (enregistrée dans un fichier).

La communication avec cet élément se fera en Java RMI. L'élément implémentera l'interface ICompte.

Partie centralisée

Cette partie contient un ensemble de processus et de ressources associées pour assurer la communication et la coordination entre les clients et le gestionnaire de compte :

- **Processus Communication**
  Il assure la communication avec les clients. Il reçoit les différentes requêtes des clients, les messages envoyés et fait suivre les messages à tous les clients actuellement connectés
- **Processus Gestion Requête**
  Il assure le traitement des requêtes du client (demande de connexion/déconnexion, création/suppression de compte, liste des connectés)
- **Processus Client RMI**
  Il sert d'interface entre le processus de gestion des requêtes et le gestionnaire de compte, vu que l'un est écrit en C et l'autre en Java. La communication entre le gestionnaire de requêtes et le client RMI se fait via une socket UDP

- **File de messages**
  Elle est utilisée pour toutes les communications entre le processus de communication et le gestionnaire de requêtes
- **Mémoire partagée**
  Elle contiendra la liste des utilisateurs actuellement connectés
- **Main**
  Le processus principal. C'est celui qui sera exécuté pour lancer tous les éléments de la partie centralisée. Son rôle est de lancer les autres processus et de créer et d'initialiser la mémoire partagée et la file de messages. De plus, il devra permettre de fermer proprement tous les éléments de la partie centralisée.

Par principe, tous ces éléments seront donc exécutés sur la même machine.

Contraintes techniques

La partie centralisée, le gestionnaire de compte et les clients doivent pouvoir être lancés sur n'importe quelle machine.

En cas de besoin de parallélisme au sein d'un processus, vous utiliserez impérativement des threads.

Une attention particulière sera portée sur la gestion des problèmes et la terminaison des différents processus. Cette terminaison peut être volontaire (*décision de l'utilisateur*) ou non (*Ctrl^C ou plantage d'un processus*). Il faut que les différentes ressources soient fermées proprement en cas d'arrêt de la partie centralisée ou d'un processus. L'application doit aussi continuer à fonctionner du mieux possible si certains éléments ne sont pas présents et si on peut s'en passer en étant dans un mode dégradé.

Travail à réaliser

Implémenter le système comme décrit ci-dessus. Essayer de travailler de manière modulaire afin de toujours fournir un programme exécutable. **Si une partie de votre programme ne marche pas, assurez-vous de pouvoir tester le reste.**

Évaluation

Le travail est à faire en binôme. L'évaluation portera sur le fonctionnement nominal, la gestion des erreurs et la qualité du code. Vous rendrez votre projet contenant toutes les sources nécessaires ainsi qu'un rapport dans une archive ZIP avant le **lundi 29 avril** à **7h00** du matin (UTC+1).

Le zip devra être nommé en suivant le modèle suivant :
SD_SE_Prenom1Nom1_Prenom2Nom2.zip

Le travail sera évalué le **lundi 29 avril** sur machines, un planning sera prévu pour établir les ordres de passage. Il faudra se présenter au moins 15 minutes à l'avance.

Il faudra fournir un rapport écrit détaillant le fonctionnement de votre application, comment compiler et lancer les différents programmes, ainsi que la structure et les choix techniques de votre code pour la partie Systèmes Distribués ET Systèmes d'Exploitation.

Tous soupçons de tricherie pourront avoir un impact sur la note des élèves et sont à la discrétion de l'examinatrice, de même que les différences de notes si les étudiants ne semblent pas avoir fourni les mêmes efforts.