Projet ING2

Ce projet a pour but d'implémenter une application de chat de type clients/serveur 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.

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 avec un tube nommé..

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 enregistrée dans un fichier. La communication avec le gestionnaire de requêtes peut se faire via un socket UDP.

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).
- **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

Le code devra être rédigé en langage C avec les primitives systèmes vues en cours. Pour ceux qui le souhaite, il est possible de créer des interface graphique avec la SDL :

(https://zestedesavoir.com/tutoriels/1014/utiliser-la-sdl-en-langage-c/).

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

Vous pouvez vous aider de scripts shell si cela est utile.

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, des signaux et des handlers appropriés doivent être utilisés dans ce cas.

Travail attendu

- Le projet est à réaliser en groupes de plusieurs étudiants suivant les directives du responsable du cours..

- Un rapport écrit détaillant la modélisation et le fonctionnement de votre application, ainsi que le code de votre application devra être envoyé au responsable du cours.

La date limite pour rendre le travail est le dimanche 17 avril 2022 à 23:59.