2024_2025 Projet IN363 : Réseau.

Project : IMMON

I make **m**y **o**wn **N**etwork

jean-baptiste.caignaert@esisar.grenoble-inp.fr

L'objectif de ce projet est de réaliser une simulation d'échange de trames réseau afin de comprendre les problématiques intrinsèques des trames, dysfonctionnement, routage, relation client/serveur …

Il est demandé de concevoir, de développer 3 programmes différents qui devront communiquer ensemble via le réseau. Il faudra prévoir la communication avec les programmes de vos camarades. Le choix du langage est Java.

Premier programme : "Client" :

- Ce programme pourra envoyer des datas utilisateurs (DU) qui seront des caractères alphanumériques, ils seront envoyés sur le réseau uniquement avec des "0" ou des "1" (langage binaire). Ces DU et le client destinataire seront envoyés par l'utilisateur
- Réception de trame réseau : le programme sera capable d'extraire les DU à partir du code binaire.
- Capacité à répondre au serveur qu'il est toujours en fonctionnement
- Proposition à l'utilisateur des différents clients et du texte à envoyer

Deuxième programme : "Serveur" :

- Ce programme recevra des trames des programmes clients et les enverra au programme client correspondant si il y a accès directement. Sinon il enverra au serveur correspondant sur sa table de routage . Il ne modifiera pas la trame.
- En cas de destinataire inconnu ou injoignable, le serveur enverra un message spécifique au client.
- Le serveur est chargé de vérifier régulièrement si les clients/serveurs sont "vivants" en cas de non-réponse, le serveur ne redirige plus les trames vers ce client/serveur et prévient les autres clients/serveurs.
- En cas de nouveau client le serveur lui alloue les caractéristiques nécessaires et prévient l'ensemble du réseau
- Les serveurs doivent pouvoir gérer les modifications de lien entre serveurs

Troisième programme : "Admin":

- Lancement du programme serveur.
- Connexion du serveur aux serveurs données avec les table de routage correspondant.
- Création des nouveaux clients (pouvant être appelé pendant le fonctionnement).
- Arrêt de certains clients (pouvant être appelé pendant le fonctionnement).
- Modification des liens possibles entre serveur (pouvant être appelé pendant le fonctionnement).

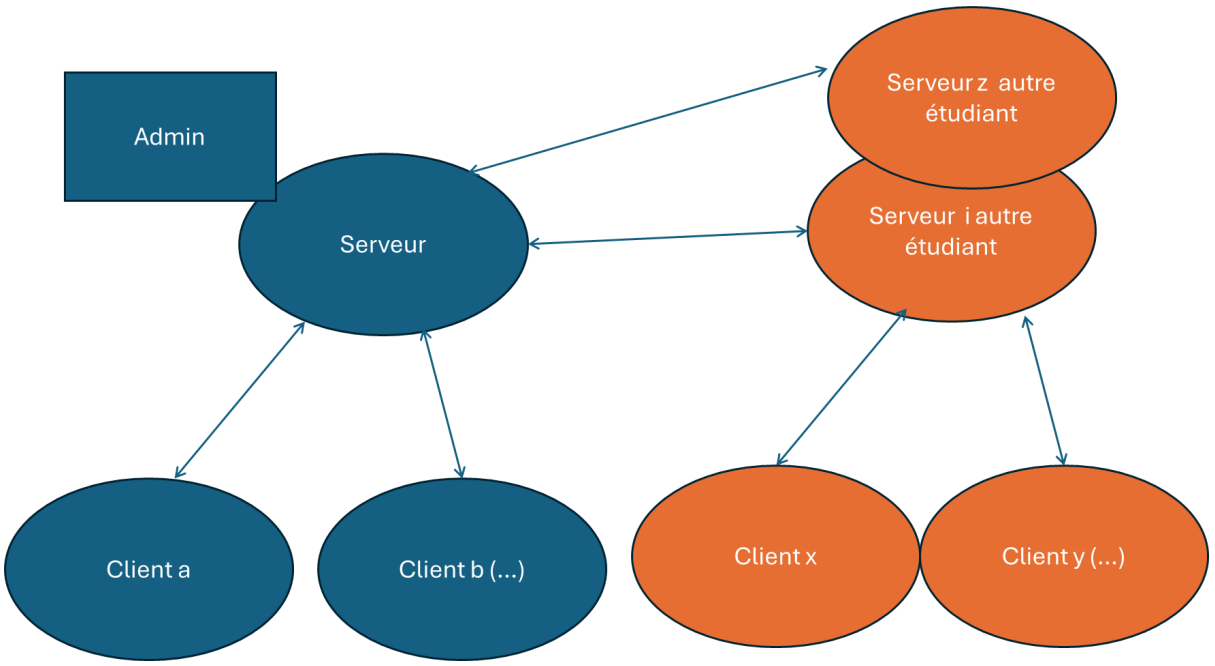

Schéma de direction des échanges, on pourra avoir jusqu'à n clients (n à définir dans votre programme via le programme admin).

Les DU seront soit codés en "LZ78, vu en cours. Un bit global de parité sera ajouté et contrôlé par le récepteur. En cas d'erreur, un message à l'utilisateur sera envoyé pour signifier le risque de corruption de données.

Le projet est prévu pour un groupe de trois étudiants. L'admin de chaque groupe fournira la configuration collective donnée (nom/IP des serveurs, et des clients).

Les actions à réaliser dans ce projet :

1. Effectuer un diagramme d'échange de message, lister de manière **exhaustive** tous les messages possibles entre les différents programmes, et les messages via l'utilisateur. (Rendu par groupe avant l'étape suivante)
2. Spécifier les trames et les protocoles d'échanges (en tête, fin de trame …). C'est à vous de définir les spécifications des trames.
3. Mise en commun collective du système de trame. **Spécification commune**.
4. Écrire en pseudo code l'algorithmie des échanges de trame, de création d'un client, d'arrêt de fonctionnement d'un client, modification de table routage.
5. Développer les programmes clients, admin et serveurs, faites des tests régulièrement.
6. Le code sera fourni avec un fichier expliquant les diverses parties du code. Le programme admin doit pouvoir se lancer avec le minimum de configuration spécifique de la part de l'utilisateur.
7. Il est tout à fait possible de faire des bonus qui seront comptabilisés lors de la notation. A voir avec l'enseignant pour les applications supplémentaires qui vous semble pertinentes. (Exemple : Possibilité d'envoyer un message multicast)

Modification 09/06/2025 :

Au lancement chaque serveur aura un numéro S01->S12
Chacun aura de 1 à 3 clients fixes : C1, C2, C3.
Chaque serveur aura une communication avec au moins 1 autre serveur.
Pas de modification en temps réel/ensuite.

Chaque admin dira : ok tout est ok, on lance la construction du réseau.

Chaque serveur doit envoyer à ses voisins une construction de ce qu'il connaît.

Ensuite chacun envoie ses messages de DU, au plus court chemin (si plus court équitable plus bas numéro serveur).

Pas de modification en cours.

Le fichier Java doit être fait collectivement, pour répondre à cette demande : drive ? Mail ?
Par exemple :

```java
package IN363.wifi;
import java.io.Serializable;

public class Message implements Serializable {
    private static final long serialVersionUID = 1L;

    private String auteur;
    private String contenu;
```

Todo : prévoir collectifs tests ?