

TP1

1 Préliminaires

1.1 TCP

On veut créer une application client/serveur. Cette application utilise des sockets TCP.

- Le serveur attend les clients sur le port 1027 et envoie régulièrement à chaque client un message contenant le nombre de clients qui sont connectés. Le serveur doit donc être capable de gérer plusieurs clients simultanément.
- Le client se connecte au serveur et affiche les nombres reçus. L'arrêt de la communication se fait à l'initiative du client.

1. Définir un protocole de communication entre le client et le serveur.
2. Donner le code du serveur et du client.

1.2 UDP

On réalise une communication client/serveur en UDP. Le client lit une chaîne au clavier et l'envoie au serveur sur le port 9876, le serveur attend un message du client et l'affiche.

On propose les codes suivants pour le serveur et le client:

```
-----  
import java.io.*;  
import java.net.*;  
  
public class ServeurUDP {  
  
    public static void main(String args[]) throws Exception {  
        DatagramSocket serverSocket = new DatagramSocket(9876);  
        byte[] receiveData = new byte[1024];  
        while (true) {
```

```

        DatagramPacket receivePacket = new
            DatagramPacket(receiveData, receiveData.length);
        serverSocket.receive(receivePacket);
        String sentence = new String(receivePacket.getData());
        System.out.println(sentence);
    }
}
}
}
-----
import java.net.*;
import java.util.Scanner;

public class ClientUDP {
    public static void main(String args[]) throws Exception {
        Scanner inFromUser = new Scanner(System.in);
        DatagramSocket clientSocket = new DatagramSocket();
        InetAddress IPAddress = InetAddress.getByName("localhost");
        byte[] sendData = new byte[1024];
        String sentence = inFromUser.nextLine();
        sendData = sentence.getBytes();
        DatagramPacket sendPacket = new
            DatagramPacket(sendData, sendData.length, IPAddress, 9876);
        clientSocket.send(sendPacket);
    }
}
}
}
-----

```

Que se passe-t-il quand le serveur est utilisé par plusieurs clients? (en particulier quand un client envoie un message long et un autre un message court)

Modifier le code du serveur pour qu'il affiche aussi l'adresse et le port de la source du message. Modifier le code du serveur pour qu'il renvoie un message au client.

2 Petites annonces

2.1 Première version

On veut réaliser une application de vente entre particulier par petite annonce à l'aide d'un gestionnaire.

Un utilisateur doit pouvoir poster une annonce (le domaine de l'annonce (voiture, moto, musique, électroménager, téléphone, autres), le prix et un descriptif) et connaître les petites annonces postées par les autres utilisateurs. Si il est intéressé par une annonce, il doit pouvoir correspondre **directement** avec celui qui a posté cette annonce. Un utilisateur doit pouvoir retirer son annonce. Le rôle du gestionnaire est de collecter les annonces, de les diffuser

aux clients, de les mettre à jour et de donner aux utilisateurs les informations pour correspondre avec les autres utilisateurs.

Vous faites partie d'un groupe pour définir l'architecture et les protocoles. Ce groupe sera défini lors de cette séance de TP

1. Quelle architecture proposez vous pour cette application ?
2. Décrivez les protocoles
3. Rédigez un document décrivant votre architecture et vos protocoles.

Pour le développement du code vous pouvez être en monôme ou binôme.

1. Ecrivez le code java correspondant aux différents éléments.
2. Votre application est-elle robuste: que se passe-t-il en cas de mauvaise rédaction des annonces, de perte de connection d'un utilisateur

2.2 Version sécurisé

On veut que les utilisateurs soient "sûrs" de la personne avec laquelle ils correspondent et que les communications entre utilisateurs soient privées et ne puissent être comprises par une tierce personne. On suppose que l'on peut faire confiance au gestionnaire. Modifier votre protocole et vos codes pour intégrer cet aspect.

3 Travail à rendre

Déposez sur Moodle pour le 2 novembre 2020

1. Les réponses correspondants aux sections 1.1 et 1.2 en indiquant comment vos programmes doivent être utilisés.
2. L'architecture et la spécification des protocoles de communication (un par groupe) pour l'application distribuée décrite section 2.1. Vous pouvez aussi me proposer lors du TP votre propre sujet.
3. Le code java du client et du serveur correspondant à une première version (un par monôme ou binôme) où il n'y aura pas de communication directe entre utilisateurs. Joignez un court rapport précisant vos choix d'implémentation.

Cette partie sera évaluée lors des TP de la semaine du 5 novembre

La communication directe entre client et les aspects sécurité seront à effectuer ensuite.