Université Paris 7
M2 IAP et M2 MIC
Protocole Internet

TP1

1 Prélimaires

1.1 TCP

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

1. 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.
2. Le client se connecte au serveur et affiche les nombres reçus. L'arrêt de la communication se fait à l'initiative du client.
   (a) Définir un protocole de communication entre le client et le serveur.
   (b) Donner le code du serveur et du client.

1.2 UDP

On réalise une communication client serveur en UDP. Le client lit une chaine 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 revoie un message au client.

2 Conversation

On veut réaliser une application qui permet à des utilisateurs de discuter deux à deux et de discuter en groupe.

Un utilisateur a un pseudonyme sur 6 caractères. Il doit pouvoir quand il se "logue" connaitre les utilisateurs et les groupes de discussion présents. Quand il rentre dans une conversation de groupe il doit recevoir le dernier message de chaque participant à la conversation. Lors d'une conversation de groupe ou deux à deux, il doit recevoir dans l'ordre les messages qui proviennent d'un autre utilisateur. Il doit aussi recevoir tous les messages à partir du moment où il est arrivé dans la conversation.

1. Quelle architecture proposez vous pour cette application ?

2. Précisez le type de communication (TCP,UDP), les ports utilisées, le contenu des messages (et éventuellement les réponses attendues).

3. Une fois vos choix présentés et validés lors de la seance de TP, rédigez un document contenant votre specification.

3 Travail à rendre

Déposez sur Didel pour le 3 novembre pour le groupe 1, 4 novembre pour le groupe 2 , 5 novembre pour le groupe 3:

1. Les codes java correspondant aux sections 1 et 2 en indiquant comment vos programmes doivent être utilisés.

2. L'architecture et la spécification du protocole de communication (un par groupe) pour la "conversation".

3. Le code java correspondant à une premiere version de la conversation (un par monôme ou binôme) où on ne considère que des communications deux à deux et on suppose qu'il n'y a aucune perte de message.