Université Ain Temouchent BELHADJ Bouchaib
Institut des Sciences
Département de Mathématiques et Informatique

Master1 RID Systèmes distribués

TP N°3
Multicast et Broadcast

Le concept de multicast permet à un groupe de machines de communiquer de telle sorte que chaque message envoyé soit reçu par l'ensemble des membres du groupe. Le groupe utilise pour communiquer une adresse IP virtuelle .Le premier octet d'une adresse IP multicast commence toujours par les 4 bits '1110' suivi par 28 bits (adresse du groupe multicast). 11100000 => 224 (valeur minimale). 11101111 => 239 (valeur maximale).

Le protocole IP utilise les adresses (virtuelles) de : 224.0.0.0 à 239.255.255.255 pour le multicast.

On choisit également un numéro de port compris entre 1024 et 65535 qui sera le numéro de port sur lequel les messages seront attendus.

Pour recevoir les messages du groupe, il faut, dans le programme Java, disposer d'un socket relié au port retenu ; on utilise pour cela la classe MulticastSocket qui hérite de la classe DatagramSocket ; il faut alors que ce socket indique qu'il joint le groupe, avec la méthode adéquate, en indiquant l'adresse IP virtuelle de ce groupe. Après cela, il suffit d'attendre des datagrammes par le socket, de la même façon que pour toute réception de datagramme.

Pour envoyer des messages au groupe, il faut construire un datagramme en indiquant l'adresse IP virtuelle et en spécifiant le port de réception.

Examinez dans ce qui suit le lien suivant :
https://docs.oracle.com/javase/7/docs/api/java/net/MulticastSocket.html

- Quel est le rôle de cette classe ? Examinez en particulier les deux méthodes joinGroup et leaveGroup.
- Soit la classe java suivante. Lancez une exécution. Que fait ce code ?
- Mettre la ligne socket.joinGroup(group); en commentaire. Pourquoi ça ne marche pas ?

```
import java.io.*;
import java.net.*;
public class Multicast {
public static void main(String argv [ ]) throws IOException{
String msg = "JE SUIS ETUDIANT EN INFORMATIQUE";
InetAddress group = InetAddress.getByName("230.0.0.0");
MulticastSocket socket = new MulticastSocket(1000) ;
socket.joinGroup(group);
DatagramPacket hi = new DatagramPacket(msg.getBytes(),
msg.length(),group, 1000);
```

```
socket.send(hi);
byte[] buf = new byte[1024];
DatagramPacket recv = new DatagramPacket(buf, buf.length);
socket.receive(recv);
String ch= new String (recv.getData());
System.out.println(ch);
socket.leaveGroup(group);
}}
```

- Reprendre la classe etudiant, créer un objet de cette classe et envoyer le par MulticastSocket. Bien évidemment il faut aussi le recevoir.
- Examinez le code suivant :

```
import java.net.*;
import java.io.*;
public class Multicast {
public static void main(String[] args) throws IOException {
DatagramSocket socket = new DatagramSocket(5000);
socket.setBroadcast(true);
InetAddress address = InetAddress.getByName("255.255.255.255");
byte[] buffer = "Master1 RID".getBytes();
DatagramPacket packet = new DatagramPacket(buffer, buffer.length,
address, 5000);
socket.send(packet);
byte[] buf = new byte[1024];
DatagramPacket recv = new DatagramPacket(buf, buf.length);
socket.receive(recv);
String ch= new String (recv.getData());
System.out.println(ch);
socket.close();
}}
```

- Quel est le rôle de la méthode setBroadcast ? Testez votre code avec le paramètre false.
- Quelle est la différence principale entre le Multicast et le Broadcast (vu en cours) ?