package lte;

import java.net.DatagramPacket;

import java.net.InetAddress;

import java.net.MulticastSocket;

import java.util.HashMap;

import java.util.StringTokenizer;

import java.util.Vector;

import javafx.application.Platform;

public class SuperSourcemultiReceiver extends Thread{

String supersource,sportno,ssysno,sstatus,availableserve="",dg="";;

public Vector vn = new Vector();

public Vector supersourcename = new Vector();

public HashMap supersourcesysno = new HashMap();

public HashMap supersourceportno = new HashMap();

public HashMap gatewayportno = new HashMap();

public HashMap gatewaysysno = new HashMap();

public Vector allgateway = new Vector();

public HashMap allstatus =new HashMap();

public HashMap alltraffic = new HashMap();

public HashMap allinsysno = new HashMap();

public HashMap allinport = new HashMap();

public HashMap allintraffic = new HashMap();

public Vector allinnodes = new Vector();

Observer o;

public SuperSourcemultiReceiver(String videoname,String sportno,String ssysno,SuperSourceController ssc)

{

this.supersource = videoname;

System.out.println("this.supersource : "+this.supersource);

this.sportno = sportno;

this.ssysno = ssysno;

this.o = ssc.getObserver();

start();

}

@Override

public void run()

{

while (true)

{

try

{

InetAddress in = InetAddress.getByName("225.89.67.45");

MulticastSocket ms = new MulticastSocket(4567);

ms.joinGroup(in);

byte[] b = new byte[1024];

DatagramPacket dp = new DatagramPacket(b, b.length);

ms.receive(dp);

String data1 = new String(dp.getData()).trim();

StringTokenizer str = new StringTokenizer(data1, "$");

String status = str.nextToken();

if(status.equalsIgnoreCase("SuperSourceNode"))

{

String videoname = str.nextToken();

String sys = str.nextToken();

String port = str.nextToken();

String sstatus = str.nextToken();

supersourcename.add(videoname);

supersourcesysno.put(videoname,sys);

supersourceportno.put(videoname,port);

}

else if(status.equalsIgnoreCase(this.supersource))

{

String availablesource = str.nextToken();

System.out.println("multireceiver receive the supersource node"+availablesource);

if(!vn.contains(availablesource))

{

vn.add(availablesource);

availableserve=availableserve+availablesource+"\n";

}

Platform.runLater(new Runnable()

{

@Override

public void run()

{

o.setavailableserver(availableserve);

}

});

}

else if(status.equalsIgnoreCase("LOCALGATEWAY"))

{

String gateway = str.nextToken();

String sys = str.nextToken();

String port = str.nextToken();

String sstatus = str.nextToken();

String traffic = str.nextToken();

gatewayportno.put(gateway,port);

gatewaysysno.put(gateway, sys);

if(!allgateway.contains(gateway))

allgateway.add(gateway);

allstatus.put(gateway,sstatus);

alltraffic.put(gateway,traffic);

}

else if(status.equalsIgnoreCase("TRAFFIC"))

{

String gateway = str.nextToken();

String traffic = str.nextToken();

alltraffic.put(gateway,traffic);

}

else if(status.equalsIgnoreCase("INTERMEDIATE"))

{

String node = str.nextToken();

String sys = str.nextToken();

String port = str.nextToken();

String intraffic = str.nextToken();

allinport.put(node,port);

allinsysno.put(node,sys);

allintraffic.put(node, intraffic);

if(!allinnodes.contains(node))

{

allinnodes.add(node);

}

}

else if(status.equalsIgnoreCase("VIDEODELETE"))

{

String dvideoname = str.nextToken();

String dgateway = str.nextToken();

vn.remove(dgateway);

for(int i =0 ;i<vn.size();i++)

{

String avai = vn.get(i).toString();

dg = dg+avai+"\n";

}

Platform.runLater(new Runnable()

{

@Override

public void run()

{

o.setavailableserver(dg);

}

});

}

}

catch(Exception e)

{

e.printStackTrace();

}

}

}

}