3CM5

Instituto Politécnico Nacional

ESCOM

Redes de Computadoras

Práctica 4

Chat Multicast

Alumno:

Naranjo Miranda Javier Said

Flores García Sergio

Grupo: 3CM5

**Introducción**

La multidifusión o difusión múltiple (en inglés: multicast) es el envío de la información en múltiples redes a múltiples destinos simultáneamente.

Antes del envío de la información, deben establecerse una serie de parámetros. Para poder recibirla, es necesario establecer lo que se denomina "grupo multicast". Ese grupo multicast tiene asociado una dirección de Internet. La versión actual del protocolo de Internet (Internet Protocol o IP), conocida como IPv4, reserva las direcciones de tipo D para la multidifusión. Las direcciones IP tienen 32 bits, y las de tipo D son aquellas en las cuales los 4 bits más significativos son '1110' (224.0.0.0 a 239.255.255.255).

Hoy dia, en muchos tipos de aplicaciones se cuenta con un servicio de chat, ya sea con fines de diversión o de negocio. Este tipo de servicio es muy socorrido cuando se trata de brindar comunicación más personalizada que la brindad por medios tales como correo electrónico o foros, En el chat la comunicación puede ser fluida, es decir, en tiempo real y además pueden interactuar dos o más personas a la vez-

**Desarrollo**

**Código**

UsuariosConectados.java

package Chat;

import java.io.IOException;

import java.io.Serializable;

import java.net.DatagramPacket;

import java.net.InetAddress;

import java.net.MulticastSocket;

import javax.swing.JOptionPane;

public class UsuariosConectados implements Serializable{

static MulticastSocket Servidor;

static InetAddress Grupo;

static byte[] bufer = new byte[1024];

static DatagramPacket mensajeEntrada;

static String linea;

static String Conectados="";

public static void EscribeMensaje(String msg) throws IOException{

byte[] m = msg.getBytes();

DatagramPacket mensajeSalida = new DatagramPacket(m, m.length, Grupo, 4000);

Servidor.send(mensajeSalida);

System.out.println("Actualizan lista todos...");

}

public static void Conectate() throws IOException{

Servidor = new MulticastSocket(4000);

Grupo = InetAddress.getByName("230.1.1.1");

Servidor.joinGroup(Grupo);

System.out.println("Unido al grupo");

while(true){

try{

mensajeEntrada = new DatagramPacket(bufer, bufer.length);

Servidor.receive(mensajeEntrada);

linea = new String(mensajeEntrada.getData(), 0, mensajeEntrada.getLength());

if(linea.indexOf("Inicio sesión")!=-1){

linea="#"+linea.substring(linea.indexOf("[")+1, linea.indexOf("]"));

Conectados+=linea;

EscribeMensaje(Conectados);

}

else if(linea.indexOf("Se ha desconectado")!=-1){

linea=linea.substring(linea.indexOf("[")+1, linea.indexOf("]"));

Conectados=Conectados.replaceFirst("#"+linea, "");

EscribeMensaje(Conectados);

}

}

catch(Exception e){

JOptionPane.showMessageDialog(null,"Ocurrio un problema con los usuarios [Conectados/Desconectados]");

}

}

}

public static void main(String args[]) throws IOException{

Conectate();

}

}

VentanaCliente.java

package Chat;

import java.awt.Color;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

import javax.swing.UIManager;

import javax.swing.UnsupportedLookAndFeelException;

import javax.swing.table.DefaultTableModel;

import javax.swing.table.TableModel;

import fonts.\*;

public class VentanaCliente extends javax.swing.JFrame implements Accent {

Cliente Acciones;

String NombreUsuario;

String Privado;

public void AlCerrar(){

try{

this.setDefaultCloseOperation(JFrame.DO\_NOTHING\_ON\_CLOSE);

addWindowListener(new WindowAdapter(){

public void windowClosing(WindowEvent e){

try {

ConfirmarSalida();

} catch (IOException ex) {

JOptionPane.showMessageDialog(null,"Error al confirmar la salida");

}

}

});

this.setVisible(true);

}

catch(Exception e){

e.printStackTrace();

}

}

public void ConfirmarSalida() throws IOException{

int valor=JOptionPane.showConfirmDialog(this,"¿Desea salir del grupo?","Confirmar salida",JOptionPane.YES\_NO\_OPTION);

if(valor==JOptionPane.YES\_OPTION){

Acciones.EscribeMensaje("["+NombreUsuario+"] Se ha desconectado");

Acciones.Desconectar();

System.exit(0);

}

}

/\*\*

\* Creates new form VentanaCliente

\*/

public VentanaCliente() throws IOException {

Acciones=new Cliente();

initComponents();

AlCerrar();

IniciarCliente();

getContentPane().setBackground(Color.WHITE);

titleLabel.setFont(fontLibrary.setRobotoBlack(14));

setAlwaysOnTop(false);

Mensaje.setFont(fontLibrary.setRobotoRegular(13));

// Emoticones.setFont(fontLibrary.setRobotoRegular(13));

ListaUsuarios.setBorder(null);

ListaUsuarios.setBackground(null);

}

public void IniciarCliente(){

try {

NombreUsuario= JOptionPane.showInputDialog("Nombre de usuario: ");

titleLabel.setText(NombreUsuario);

Acciones.Conectate(ListaUsuarios,NombreUsuario,Emoticones);

Acciones.EscribeMensaje("<b>["+NombreUsuario+"]</b> Inicio sesión");

} catch (IOException ex) {

JOptionPane.showMessageDialog(null,"Error al conectarme");

}

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

Mensaje = new javax.swing.JTextField();

EnviarMensaje = new javax.swing.JButton();

ScrollTabla = new javax.swing.JScrollPane();

ListaUsuarios = new javax.swing.JTable();

ScrollArea = new javax.swing.JScrollPane();

Emoticones = new javax.swing.JEditorPane();

jPanel1 = new javax.swing.JPanel();

titleLabel = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setBackground(new java.awt.Color(255, 255, 255));

setName("Ventana"); // NOI18N

setResizable(false);

EnviarMensaje.setText("Enviar");

EnviarMensaje.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

EnviarMensajeActionPerformed(evt);

}

});

ListaUsuarios.setFont(new java.awt.Font("Roboto", 0, 12)); // NOI18N

ListaUsuarios.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Usuarios"

}

) {

Class[] types = new Class [] {

java.lang.String.class

};

boolean[] canEdit = new boolean [] {

false

};

public Class getColumnClass(int columnIndex) {

return types [columnIndex];

}

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

ListaUsuarios.setAutoscrolls(false);

ListaUsuarios.setFocusable(false);

ListaUsuarios.setGridColor(new java.awt.Color(255, 255, 255));

ListaUsuarios.setShowHorizontalLines(false);

ListaUsuarios.setShowVerticalLines(false);

ScrollTabla.setViewportView(ListaUsuarios);

if (ListaUsuarios.getColumnModel().getColumnCount() > 0) {

ListaUsuarios.getColumnModel().getColumn(0).setResizable(false);

}

ListaUsuarios.setFont(fontLibrary.setRobotoRegular(13));

Emoticones.setEditable(false);

Emoticones.setBorder(null);

Emoticones.setContentType("text/html"); // NOI18N

Emoticones.setFont(new java.awt.Font("Roboto", 0, 12)); // NOI18N

Emoticones.setText("<html>\r <head>\r \r </head>\r <body>\r <p style=\"margin-top: 10px\" font-face=\"Roboto\">\r \r </p>\r </body>\r </html>\r ");

Emoticones.setAutoscrolls(false);

ScrollArea.setViewportView(Emoticones);

Emoticones.setFont(fontLibrary.setRobotoRegular(14));

ScrollArea.setBorder(null);

jPanel1.setBackground(accent);

titleLabel.setForeground(new java.awt.Color(255, 255, 255));

titleLabel.setText("CHAT MULTICAST");

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(23, 23, 23)

.addComponent(titleLabel)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addComponent(titleLabel, javax.swing.GroupLayout.DEFAULT\_SIZE, 23, Short.MAX\_VALUE)

.addContainerGap())

);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(8, 8, 8)

.addComponent(ScrollTabla, javax.swing.GroupLayout.PREFERRED\_SIZE, 170, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addGroup(layout.createSequentialGroup()

.addComponent(Mensaje, javax.swing.GroupLayout.PREFERRED\_SIZE, 338, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(EnviarMensaje))

.addComponent(ScrollArea))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addGap(0, 0, 0)

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addGroup(layout.createSequentialGroup()

.addComponent(ScrollArea, javax.swing.GroupLayout.PREFERRED\_SIZE, 196, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(Mensaje)

.addComponent(EnviarMensaje, javax.swing.GroupLayout.DEFAULT\_SIZE, 54, Short.MAX\_VALUE)))

.addComponent(ScrollTabla, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private void EnviarMensajeActionPerformed(java.awt.event.ActionEvent evt) {

try {

// TODO add your handling code here:

if(ListaUsuarios.getSelectedRow()==-1){

Acciones.EscribeMensaje("<b>"+NombreUsuario+"</b>: "+Mensaje.getText());

Mensaje.setText("");

}

else{

DefaultTableModel DTMPrivado = (DefaultTableModel) ListaUsuarios.getModel();

TableModel TMPrivado = ListaUsuarios.getModel();

Privado = TMPrivado.getValueAt(ListaUsuarios.getSelectedRow(),0).toString();

Privado = NombreUsuario+"&"+Privado+"$"+Mensaje.getText();

Acciones.EscribeMensaje(Privado);

ListaUsuarios.clearSelection();

Mensaje.setText("");

}

} catch (IOException ex) {

JOptionPane.showMessageDialog(null,"Error al mandar mensaje");

}

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

System.setProperty("awt.useSystemAAFontSettings","on");

System.setProperty("swing.aatext", "true");

try {

// Set System L&F

UIManager.setLookAndFeel(

UIManager.getSystemLookAndFeelClassName());

}

catch (UnsupportedLookAndFeelException e) {

// handle exception

}

catch (ClassNotFoundException e) {

// handle exception

}

catch (InstantiationException e) {

// handle exception

}

catch (IllegalAccessException e) {

// handle exception

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

try {

new VentanaCliente().setVisible(true);

} catch (IOException ex) {

Logger.getLogger(VentanaCliente.class.getName()).log(Level.SEVERE, null, ex);

}

}

});

}

// Variables declaration - do not modify

private javax.swing.JEditorPane Emoticones;

private javax.swing.JButton EnviarMensaje;

private javax.swing.JTable ListaUsuarios;

private javax.swing.JTextField Mensaje;

private javax.swing.JScrollPane ScrollArea;

private javax.swing.JScrollPane ScrollTabla;

private javax.swing.JPanel jPanel1;

private javax.swing.JLabel titleLabel;

// End of variables declaration

}

Cliente.java

package Chat;

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.InetAddress;

import java.net.MulticastSocket;

import javax.swing.JEditorPane;

import javax.swing.JTable;

import javax.swing.JTextArea;

class Cliente {

static MulticastSocket Cliente;

static InetAddress Grupo;

static DatagramPacket mensajeEntrada;

static String linea;

static HiloRecibeMensajes Receptor;

public void Conectate(JTable ListaUsuarios,String YoSoy, JEditorPane Emoticones) throws IOException{

Cliente = new MulticastSocket(4000);

Grupo = InetAddress.getByName("230.1.1.1");

Cliente.joinGroup(Grupo);

Receptor=new HiloRecibeMensajes(Cliente,ListaUsuarios,YoSoy,Emoticones);

Receptor.start();

System.out.println("Unido al grupo");

}

public void EscribeMensaje(String msg) throws IOException{

byte[] m = msg.getBytes();

DatagramPacket mensajeSalida = new DatagramPacket(m, m.length, Grupo, 4000);

Cliente.send(mensajeSalida);

System.out.println("Mensaje enviado...");

}

void Desconectar() throws IOException {

Cliente.leaveGroup(Grupo);

Receptor.stop();

}

}

HiloRecibe.java

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Chat;

import fonts.fontLibrary;

import java.awt.Image;

import java.net.DatagramPacket;

import java.net.MulticastSocket;

import java.util.ArrayList;

import javax.swing.ImageIcon;

import javax.swing.JEditorPane;

import javax.swing.JOptionPane;

import javax.swing.JTable;

import javax.swing.JTextArea;

import javax.swing.table.DefaultTableModel;

import javax.swing.text.Document;

/\*\*

\*

\* @author Harold

\*/

public class HiloRecibeMensajes extends Thread{

DatagramPacket mensajeEntrada;

byte[] bufer = new byte[1024];

String linea;

MulticastSocket Cliente;

JTable ListaUsuarios;

String Nombre;

String NombrePrivado="";

String MiNombre;

static String FilaTabla[] = new String[1];

ArrayList<Integer> IndicesNumerales = new ArrayList<Integer>();

JEditorPane AreaEmoticones;

String LoqueHabia="";

String QuienEnvia="";

public HiloRecibeMensajes(MulticastSocket Cliente, JTable Usuarios,String YoSoyPrivado,JEditorPane Emoticones){

this.Cliente=Cliente;

this.ListaUsuarios=Usuarios;

this.MiNombre=YoSoyPrivado;

this.AreaEmoticones=Emoticones;

AreaEmoticones.setFont(fontLibrary.setRobotoRegular(12));

}

public void run(){

while(true){

try{

mensajeEntrada = new DatagramPacket(bufer, bufer.length);

Cliente.receive(mensajeEntrada);

linea = new String(mensajeEntrada.getData(), 0, mensajeEntrada.getLength());

if(linea.indexOf("#")!=-1){

IndicesNumerales.clear();

DefaultTableModel DTMVaciado=(DefaultTableModel) ListaUsuarios.getModel();

int Filas=ListaUsuarios.getRowCount();

for(int i=0;Filas>i; i++){

DTMVaciado.removeRow(0);

}

for(int i=0;i<linea.length();i++){

if(linea.charAt(i)=='#'){

if(!IndicesNumerales.contains(i)){

IndicesNumerales.add(i);

}

}

}

for(int j=0;j<IndicesNumerales.size();j++){

if(j==IndicesNumerales.size()-1){

Nombre=linea.substring(IndicesNumerales.get(j)+1,linea.length());

}

else{

Nombre=linea.substring(IndicesNumerales.get(j)+1,IndicesNumerales.get(j+1));

}

DefaultTableModel DTM=(DefaultTableModel) ListaUsuarios.getModel();

FilaTabla[0]=Nombre;

DTM.addRow(FilaTabla);

}

}

else{

if(linea.indexOf("&")!=-1){

QuienEnvia=linea.substring(0,linea.indexOf("&"));

NombrePrivado=linea.substring(linea.indexOf("&")+1,linea.indexOf("$"));

if(MiNombre.equals(NombrePrivado)){

linea = "<font face = \"Roboto\" <i>[Privado]</i> <b>"+QuienEnvia+"</b>: "+linea.substring(linea.indexOf("$")+1, linea.length())+"</font>";

//CHECA SI HAY EMOTICONES

ReemplazaEmoticones();

LoqueHabia+=linea+"<br>";

System.out.println(LoqueHabia);

AreaEmoticones.setText(LoqueHabia);

}

}

else{

//CHECA SI HAY EMOTICONES

ReemplazaEmoticones();

LoqueHabia+=linea+"<br>";

System.out.println(LoqueHabia);

AreaEmoticones.setContentType("text/html");

AreaEmoticones.setText("<font face = \"Roboto\">"+LoqueHabia+"</font>");

}

}

}

catch(Exception e){

JOptionPane.showMessageDialog(null,"Ocurrio un problema");

}

}

}

private void ReemplazaEmoticones() {

linea=linea.replace(":)", "<img src="+"\""+"file:src\\Imagenes\\Feliz.png"+"\""+" width=\"4px\" height=\"4px\"/>");

linea=linea.replace(":(", "<img src="+"\""+"file:src\\Imagenes\\Triste.png"+"\""+" width=\"4px\" height=\"4px\"/>");

linea=linea.replace(":'(","<img src="+"\""+"file:src\\Imagenes\\Llora.png"+"\""+" width=\"4px\" height=\"4px\"/>");

linea=linea.replace(":\*", "<img src="+"\""+"file:src\\Imagenes\\Amor.png"+"\""+" width=\"4px\" height=\"4px\"/>");

linea=linea.replace(";)", "<img src="+"\""+"file:src\\Imagenes\\Picaro.png"+"\""+" width=\"4px\" height=\"4px\"/>");

linea=linea.replace("B|", "<img src="+"\""+"file:src\\Imagenes\\Lentes.png"+"\""+" width=\"4px\" height=\"4px\"/>");

linea=linea.replace(":P", "<img src="+"\""+"file:src\\Imagenes\\Lengua.png"+"\""+" width=\"4px\" height=\"4px\"/>");

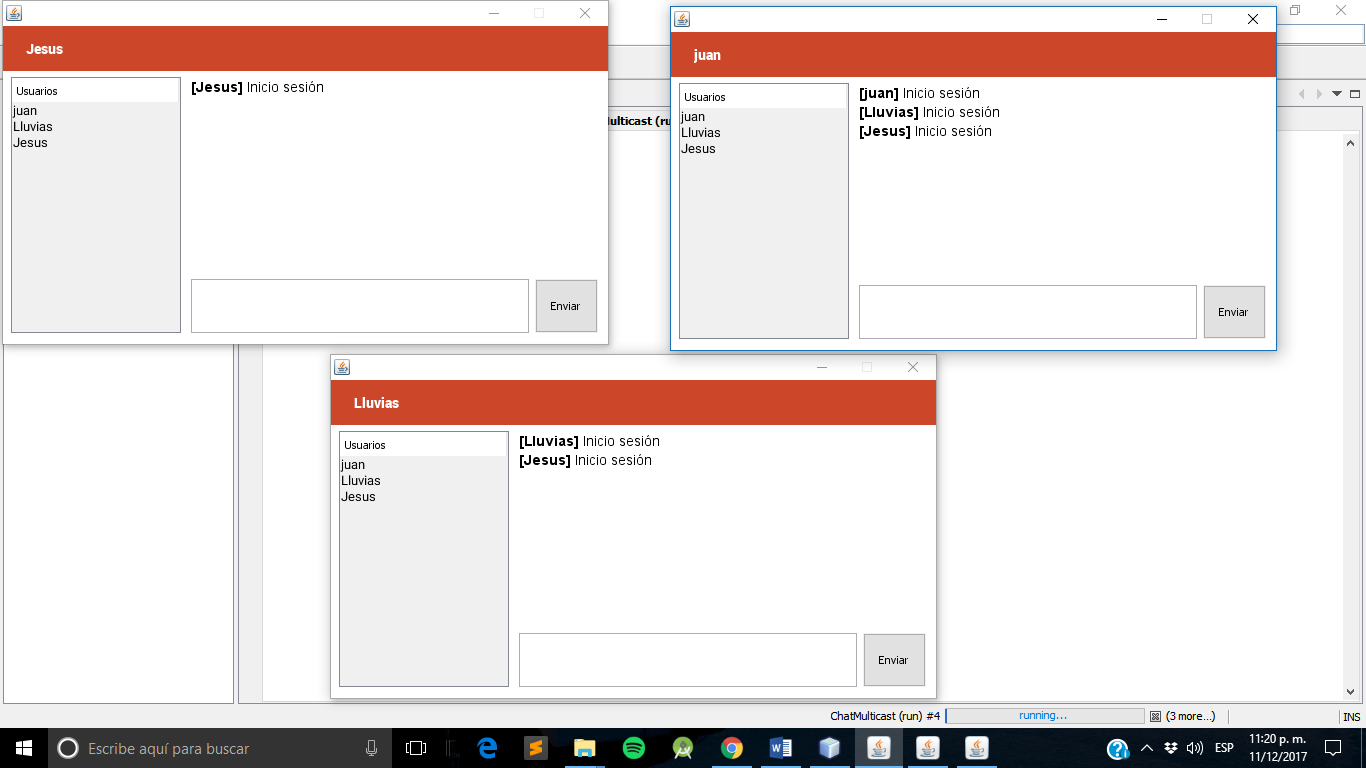
AreaEmoticones.repaint();

}

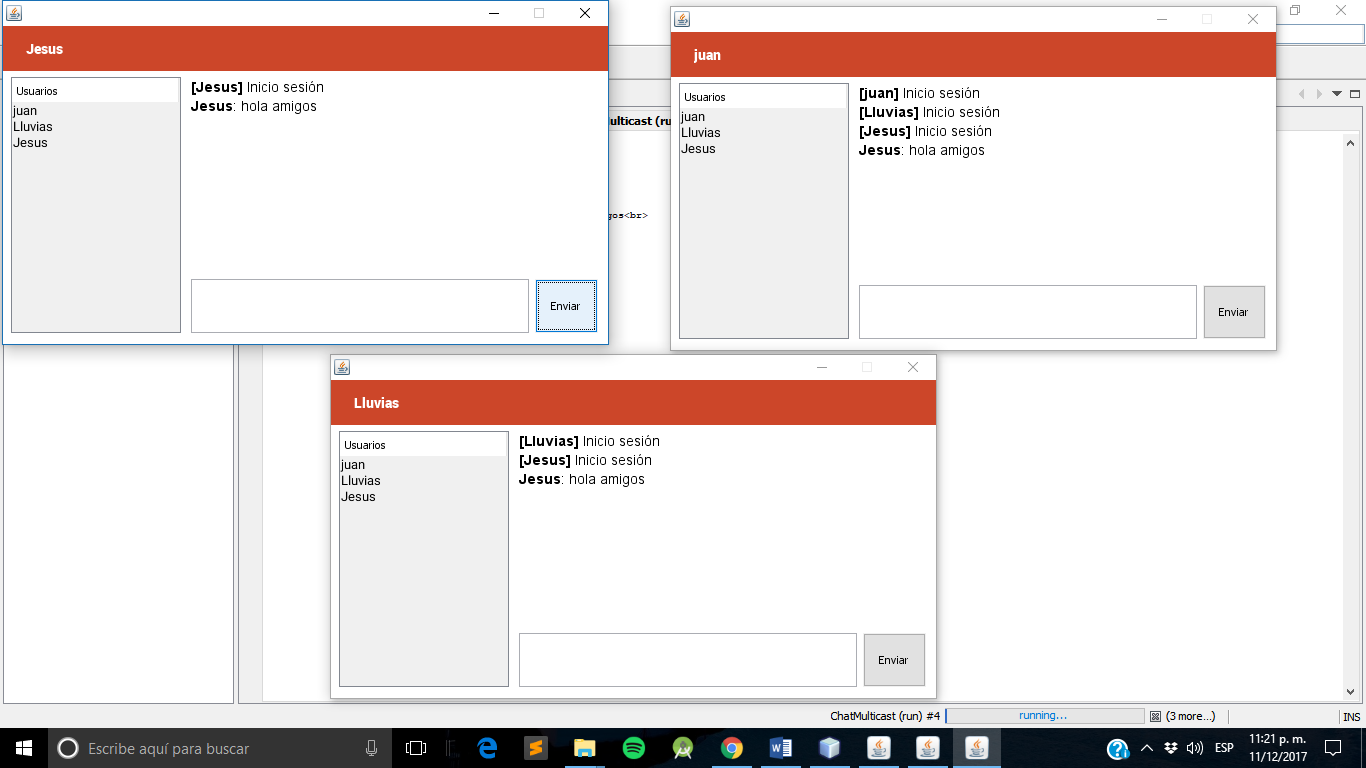
}

**Pruebas**

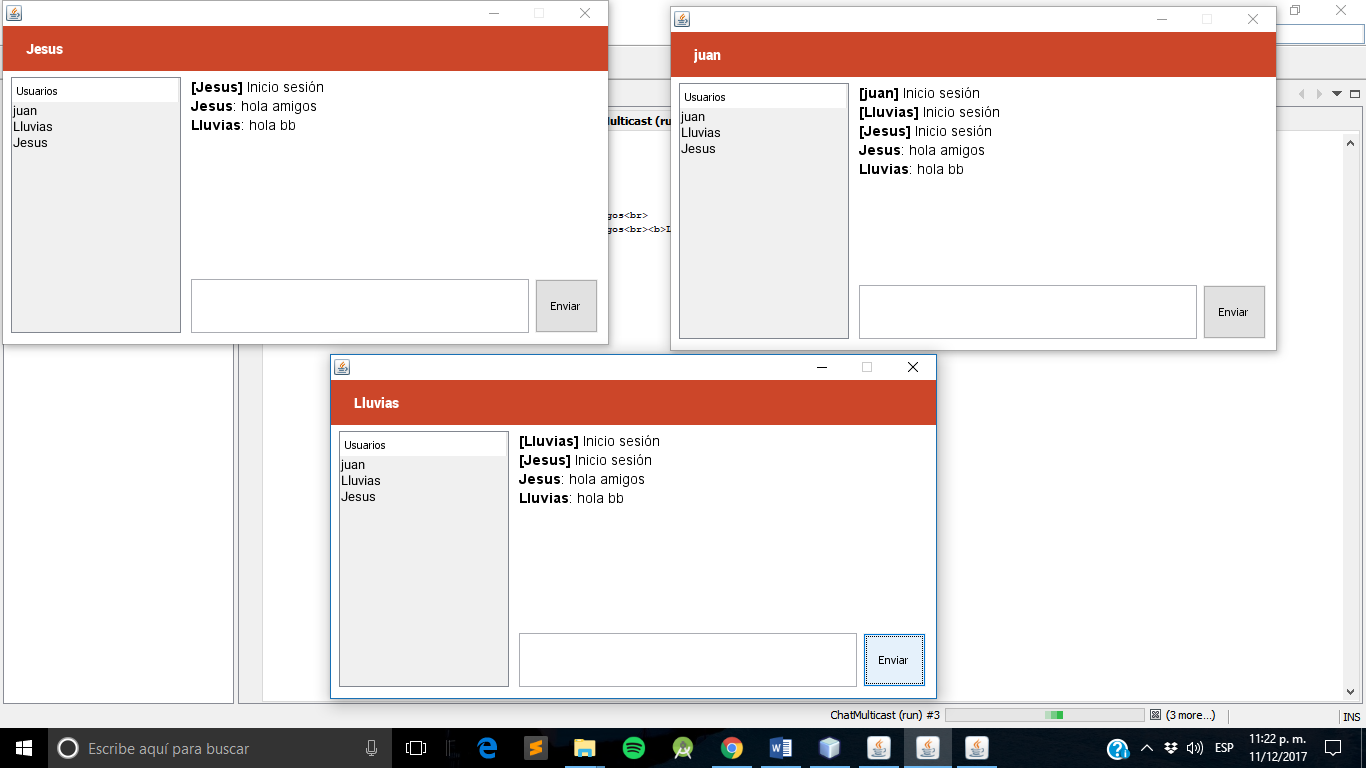
Unimos los usuarios al grupo y el servidor anuncia a todos, por lo cual podemos ver los usuarios conectados.



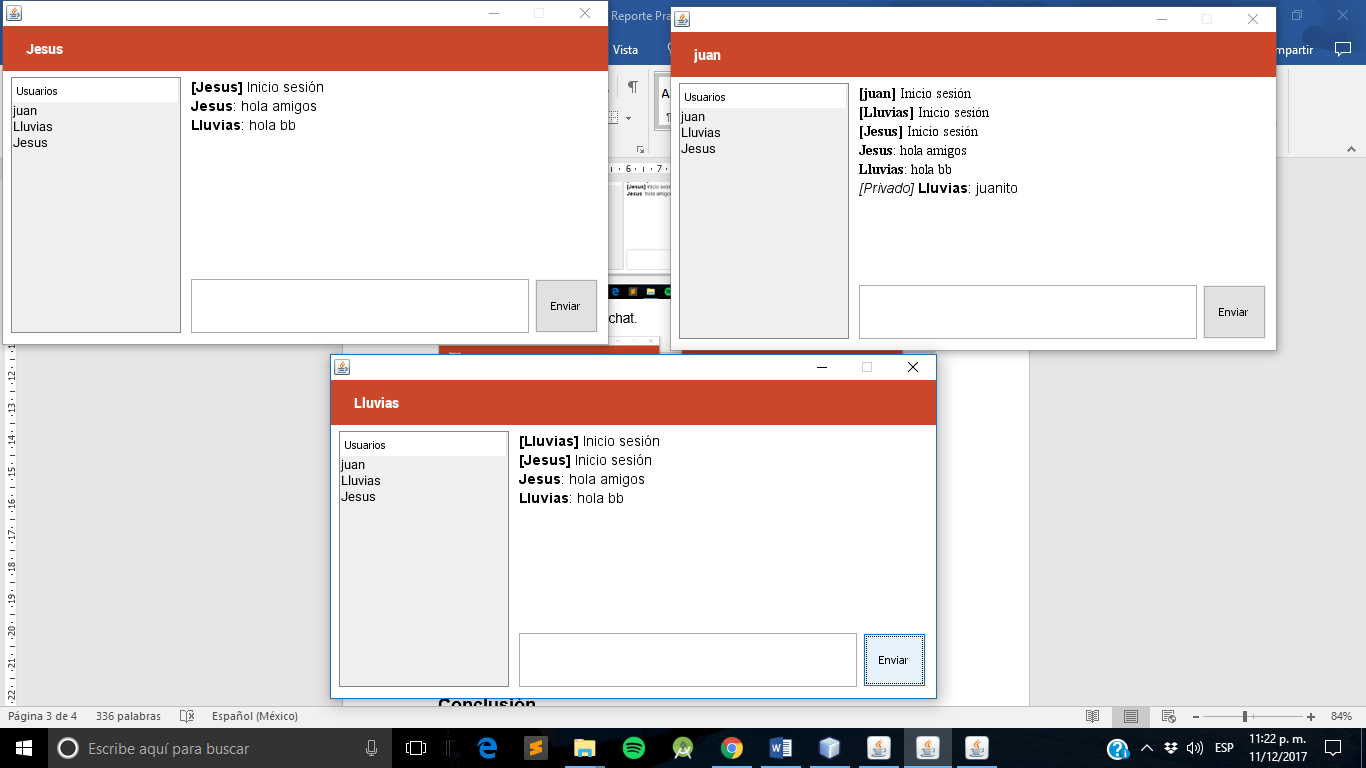
Ahora mandaremos un mensaje a todos los usuarios



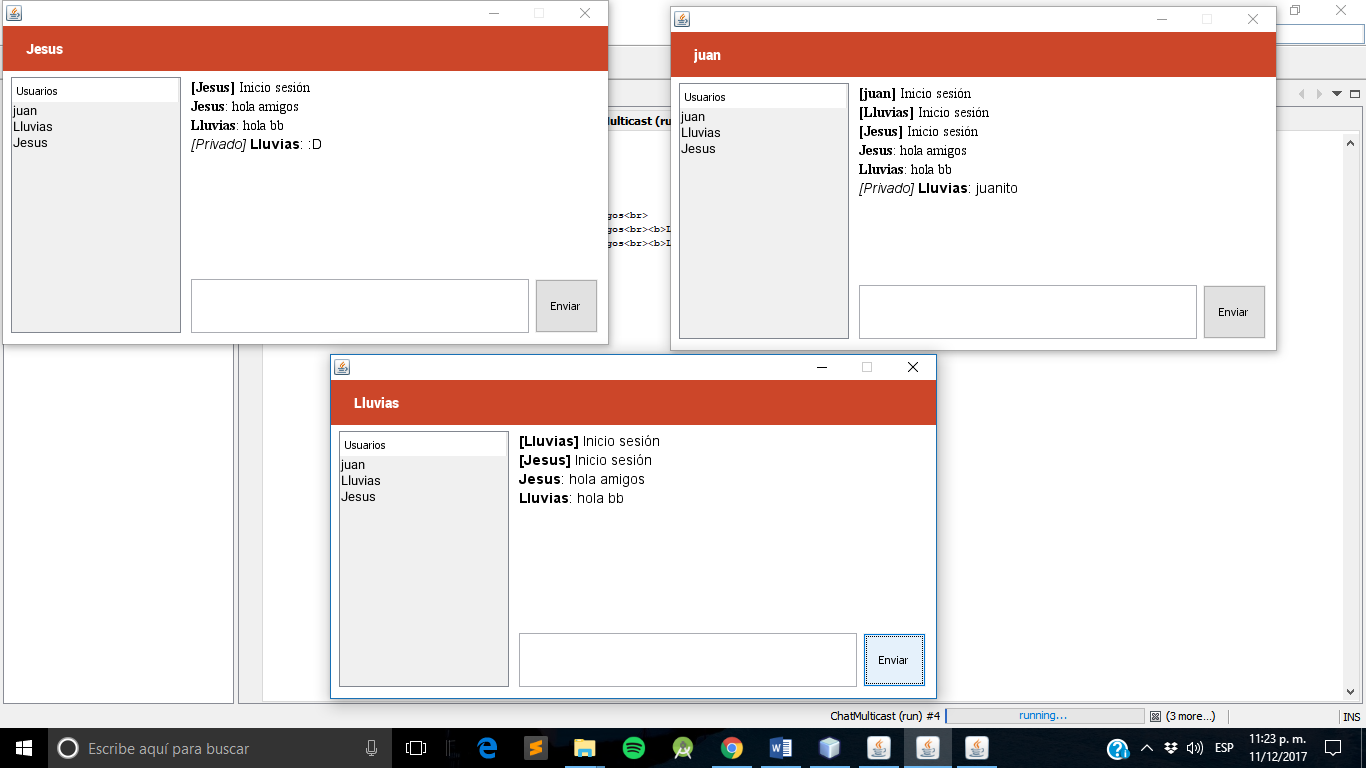
Responderemos desde otro chat.



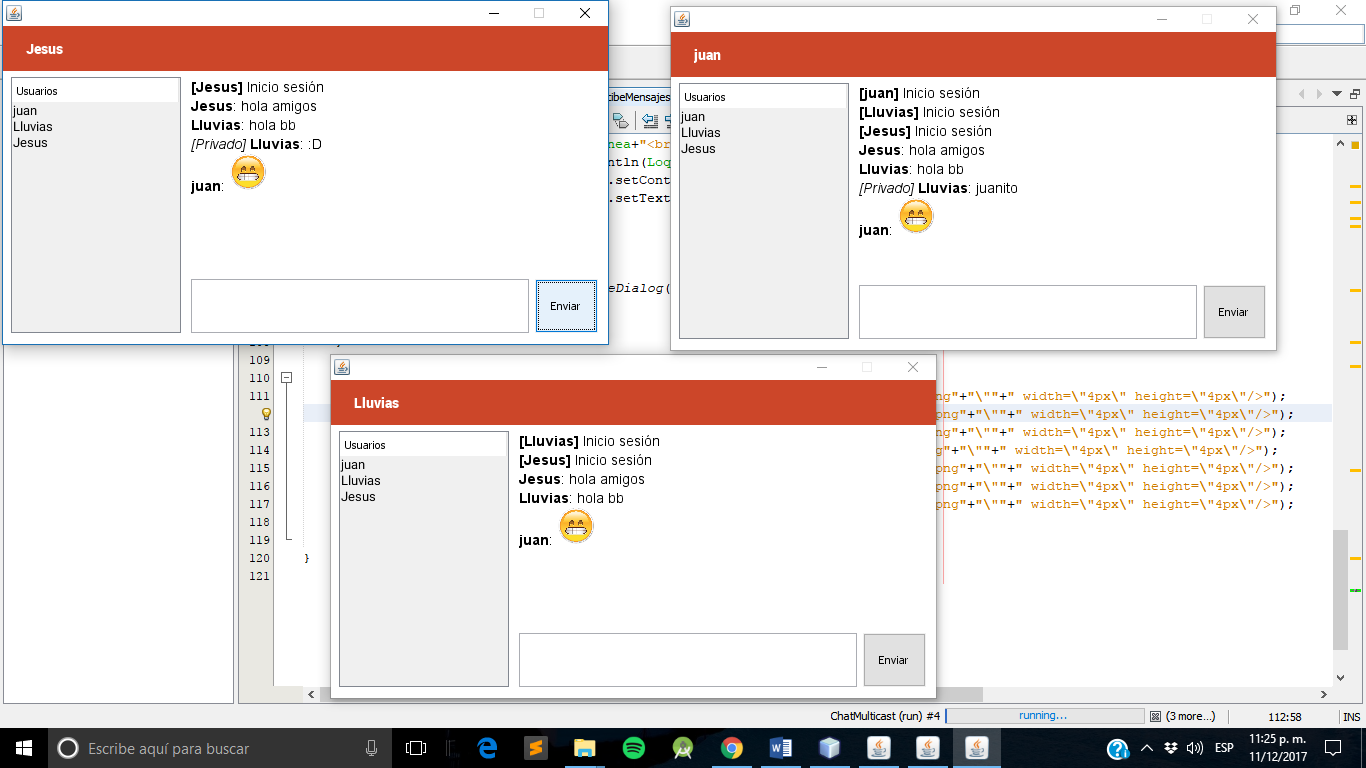
Ahora mandaremos el mensaje solo a una persona en particular, en este caso será a juan.



Ahora un mensaje privado a Jesus



De nuevo mandaremos un mensaje a todos pero ahora se verá un emoticón



**Conclusión**

Con la ayuda de esta práctica pudimos observar el funcionamiento de las conexiones multicas, que podrían llevar cierta ventaja contra los unicat, cuando es necesario compartir información con un grupo de personas, ya que de esta forma basta con compartir la información en el grupo para que todos los usuarios que pertenecen al grupo la obtengan.

El desarrollo del servidor en los sockets multicast es muy importante debido a que este es el encargado de anunciar a todos los usuarios que se están uniendo al grupo.