# Instituto Politécnico Nacional

## Escuela Superior de Cómputo

Aplicaciones para comunicaciones en red

Moreno Cervantes Axel Ernesto

Práctica 3 Chat

Integrantes del equipo:

GARCIA MARCIANO EDGAR

HERNANDEZ OBLE AXEL

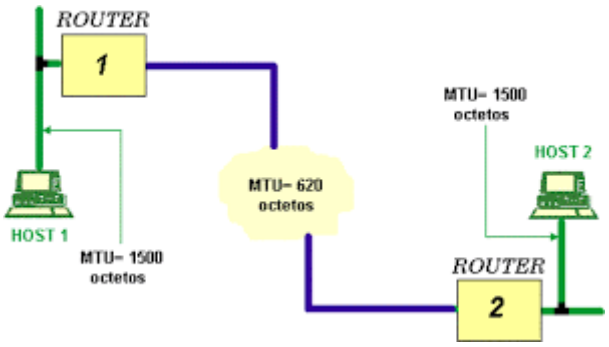
**3CM16**

# Introducción

## Datagrama

Un datagrama es un paquete de datos que constituye el mínimo bloque de información en una red de conmutación por datagramas, la cual es uno de los dos tipos de protocolo de comunicación por conmutación de paquetes usados para encaminar por rutas diversas dichas unidades de información entre nodos de una red, por lo que se dice que no está orientado a conexión. La alternativa a esta conmutación de paquetes es el circuito virtual, orientado a conexión.

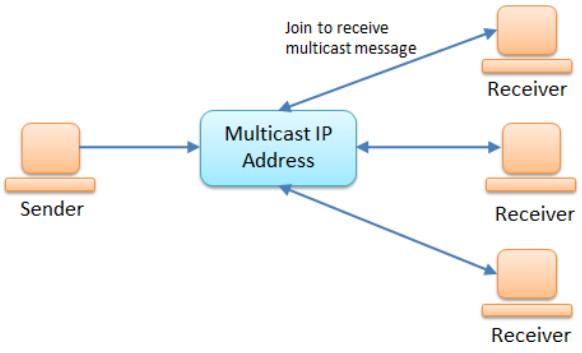
Una de las mayores ventajas que presentan los datagramas es que evitan que la red se cargue, y controlan perfectamente el tráfico de datos. Así no suelen darse bloqueos ni muchos fallos en el sistema. También hay que tener en cuenta que el uso de los datagramas es más barato que los circuitos virtuales, dado que el proceso de transmisión es totalmente independiente a otros elementos informáticos. Pero como ocurre en la gran mayoría de sectores los datagramas también presentan desventajas.




## Socket Multicast

La clase de socket de datagrama de multidifusión es útil para enviar y recibir paquetes de multidifusión IP. Un MulticastSocket es un DatagramSocket (UDP), con capacidades adicionales para unirse a "grupos" de otros hosts de multidifusión en Internet.

Un grupo de multidifusión se especifica mediante una dirección IP de clase D y un número de puerto UDP estándar. Las direcciones IP de Clase D están en el rango 224.0.0.0 a 239.255.255.255, inclusive. La dirección 224.0.0.0 está reservada y no debe utilizarse.



## Desarrollo

Muchas empresas hacen uso de Internet para ofrecer sus productos y servicios, ya que a través de este medio pueden estar en contacto directo con los clientes potenciales. Algunas de las principales ventajas de Internet como medio para realizar negocios son la cobertura a nivel global, así como la disponibilidad del servicio los 365 días, las 24 hrs. del día. Sin embargo, estas ventajas se convierten también en un reto para las empresas, pues se debe proporcionar a los clientes una vía de comunicación directa para resolver las dudas de los clientes potenciales, atender sus peticiones, así como proporcionarles información adicional a la expuesta en sus portales. Existen diversos recursos que pueden ser utilizados, como los foros o el correo electrónico, pero ninguno de ellos permite la comunicación en tiempo real. El chat es un excelente recurso para este tipo de propósito, ya que además de ser una vía de comunicación en tiempo real, permite comunicar a dos o más usuarios entre sí. Típicamente los chats se han implementado haciendo uso de sockets de flujo, pero existen otras alternativas que también merece la pena probar, tales como los sockets de datagrama o los de multidifusión.

### Clase chat

La clase chat se centra en definir las variables necesarias para la interfaz gráfica, se declaran todos los elementos necesarios junto con el tamaño y posición de cada uno.

```
public class Chat82 extends JFrame implements ActionListener {

    private JButton enviar, emoji1, emoji2, emoji3, emoji4, emoji5, emoji6, emoj
    private JTextField entrada;
    private JEditorPane editor;
    public String nombreGuardado;
    private enviar hiloEnvia;
    private JScrollPane scrollPane;
    private JComboBox<String> combo1;
    private String usuarioOnline;
    private String[] nombres = new String[100];
    private int numNombres = 0;
    private int privado = 0;
    private String seleccionado = "";
    //private int tipo=0;// 0 es publico ,1 es privado
    //img=0 no img, si es 1 es img
    String mensajeAnterior = "<b>Chat Online con MulticastSocket</b>", mensajeGu

    public Chat82() {
        super("Chat");
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setSize(800, 800);
        this.setVisible(true);
        this.setLayout(null); //Layout absoluto
        this.getContentPane().setBackground(Color.BLUE);
        scrollPane = new JScrollPane();
        scrollPane.setBounds(5, 0, 790, 600);
        editor = new JEditorPane("text/html", null); //Incilaizamos el JTexpane
        editor.setPreferredSize(new Dimension(755, 600));
        editor.setEditable(false);
        scrollPane.setViewportView(editor);
        this.add(scrollPane);
        editor.setText("<b>Chat Online con MulticastSocket</b>");
        entrada = new JTextField();
        entrada.setBounds(5, 620, 670, 50);
        this.add(entrada);

        combo1 = new JComboBox<String>();
        combo1.setBounds(5, 710, 150, 50);
        this.add(combo1);

        combo1.addItem("Todos");
        combo1.addActionListener((ActionEvent e) -> {
```

Así mismo, esta clase tiene el apartado override el cual contiene el actionPerformed, ahí se le asignará los eventos que tendrá cada posible acción dentro de la interfaz gráfica de usuario.

```
@Override
public void actionPerformed(ActionEvent e) {
    JButton btn = (JButton) e.getSource();
    if (btn == enviar) {
          System.out.println("pRIVADO: "+privado);
        System.out.println("tam selc:" + seleccionado.length());
        if (privado == 1) {
            String message = entrada.getText();
            hiloEnvia = new enviar(message, nombreGuardado, selecci
            hiloEnvia.start();
            entrada.setText("");
            setMensaje(nombreGuardado,message,seleccionado);
        }
        if (privado == 0) {
            String message = entrada.getText();
            hiloEnvia = new enviar(message, nombreGuardado, selecci
            hiloEnvia.start();
            entrada.setText("");
        }

    }
    if (btn == emoji1) {

        String imgsrc = Chat82.class.getClassLoader().getSystemReso
```

De igual forma, contendrá la función enviar la cual extiende hilos, con esta se usará el multicast y un buffer para poder enviar los mensajes, tanto al chat grupal como al chat individual.

```
public class enviar extends Thread {
    // El dato que queramos enviar en el mensaje, como array de bytes.

    byte buffer[] = new byte[6500];
    MulticastSocket enviador;
    String name, mensaje, destino;
    int tipo;
    int img;

    public enviar(String mensaje, String name, String destino, int tipo, int
        this.mensaje = mensaje;
        this.name = name;
        this.destino = destino;
        this.tipo = tipo;
        this.img = img;
    }

    @Override
    public void run() {
        mensajeDeUsuario c;
        c = new mensajeDeUsuario(mensaje, name, destino, tipo, img);
        ByteArrayOutputStream bs = new ByteArrayOutputStream();
        ObjectOutputStream os;
```

El apartado de escuchar usara las mismas propiedades que enviar, con la diferencia que este apartado se encargara de mostrar los mensaje, y los emojis disponibles.

```
public class escuchar extends Thread {

    // El mismo puerto que se uso en la parte de enviar.
    byte[] buffer2 = new byte[6500];
    MulticastSocket escucha;

    public int busquedaBinariaRecursiva(String[] arreglo, String busqueda, i
        // Si izquierda es mayor que derecha significa que no encontramos na
        int resultadoDeLaComparacion = -2;
        if (izquierda > derecha) {
            return -1;
        }

        // Calculamos las mitades...
        int indiceDelElementoDelMedio = (int) Math.floor((izquierda + derech
        String elementoDelMedio = arreglo[indiceDelElementoDelMedio];

        // Primero vamos a comparar y luego vamos a ver si el resultado es n
        // positivo o 0
        if (elementoDelMedio == null) {
            resultadoDeLaComparacion = -1;
        } else {
            resultadoDeLaComparacion = busqueda.compareTo(elementoDelMedio);
        }
```

## Clase inicioApp

Esta clase tiene como fin iniciar el programa mandando a llamas la gui generada en la clase chat incluyéndole el panel y un mensaje de inicio solicitando el nombre de usuario.

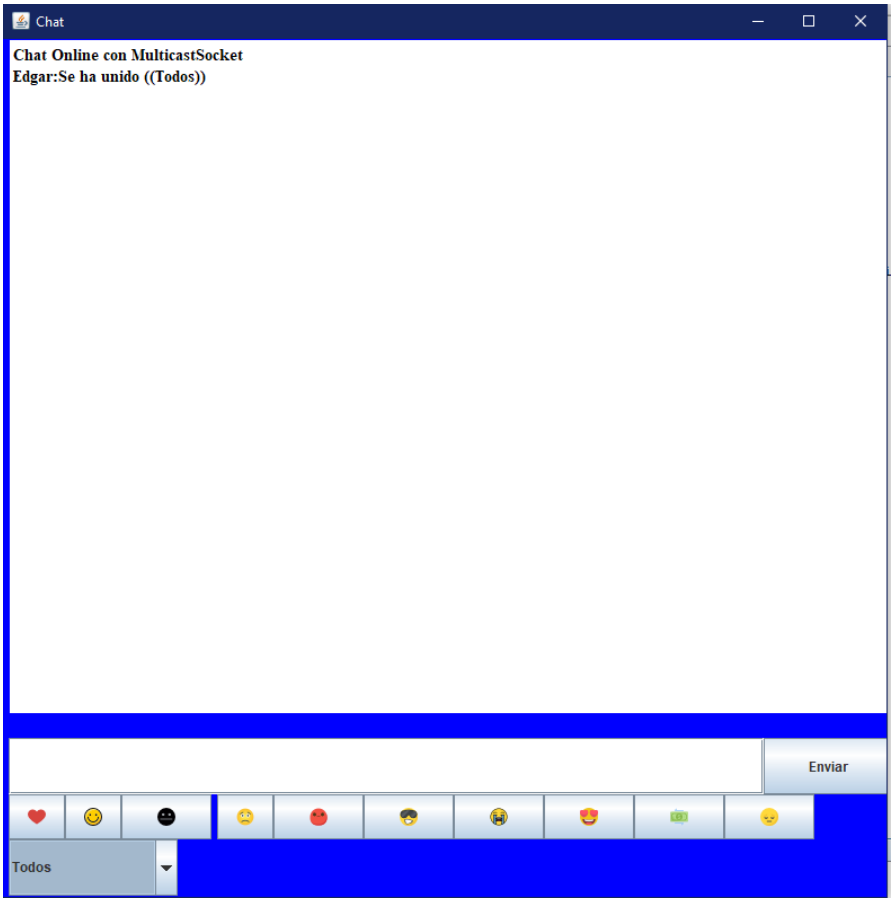
```
public class inicioApp {

    public static void main(String[] args) {
        String name = JOptionPane.showInputDialog("Ingresa tu nombre por favor");
        if ("".equals(name)) {
            JOptionPane.showMessageDialog(null, "Nombre no aceptado");
            System.exit(0);
        } else {
            Chat82 c = new Chat82();
            c.h(name, "Se ha unido", "(Todos)");
        }
    }
```

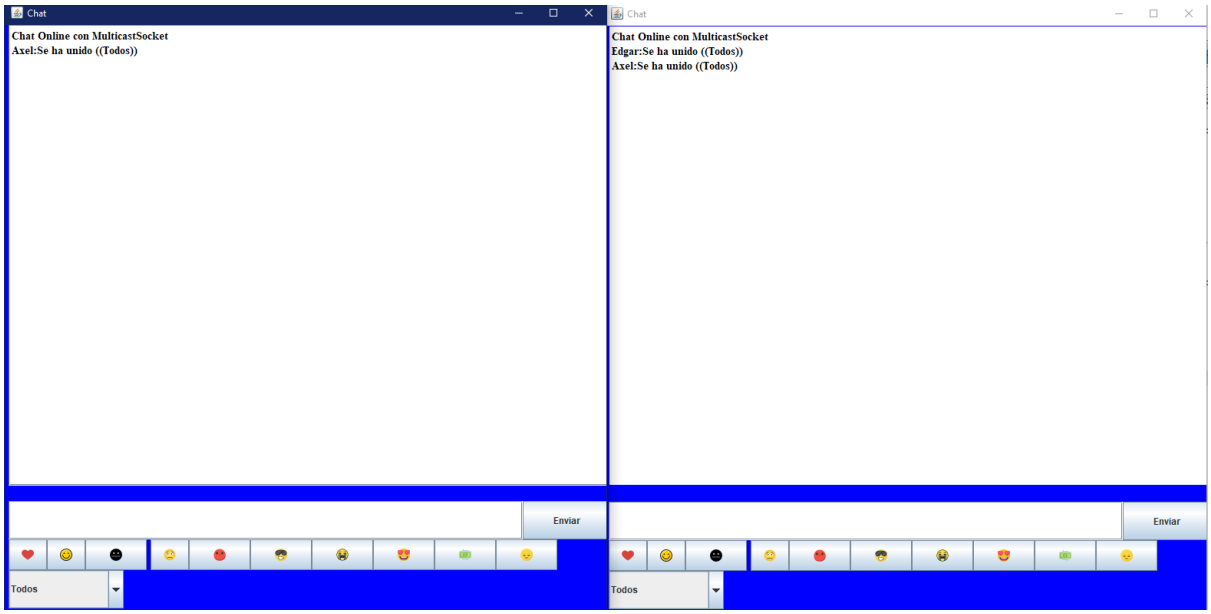
## Clase mensajeDelUsario

Esta clase está destinada únicamente a declarar variables y funciones para cada variable, viene siendo un crud con las variables necesarias para la clase chat y destinada a los mensajes individuales y no grupales.

```java
public class mensajeDeUsuario implements Serializable {

    private String mensaje;
    private String usuarioOrigen;
    private String usuarioDestino;
    private int tipo;
    private int img;

    public mensajeDeUsuario (String mensaje, String usuarioOrigen, String usuario
        this.mensaje = mensaje;
        this.usuarioOrigen = usuarioOrigen;
        this.usuarioDestino = usuarioDestino;
        this.tipo=tipo;
        this.img=img;

    }

    public int getImg() {
        return img;
    }

    public void setImg(int img) {
        this.img = img;
    }
```

# Muestreo de la GUI



## Preguntas

1.- ¿Qué ventajas presenta el uso de sockets de multidifusión contra unidifusión?

La gran diferencia radica en el numero de receptores, en el unidifusión es una conexión punto a punto donde solo hay un emisor y un receptor mientras que el multidifusión tiene un emisor y muchos receptores.

2. ¿Qué modificaciones a nivel de campo Tiempo de vida es necesario considerar?

El tiempo entre que un usuario ha mandado un ultimo mensaje, si pasa un tiempo determinado tomar a ese usuario como ausente y desconectarlo del servidor.

3. ¿Qué consecuencias tendrá el desempeño de la aplicación habilitar el algoritmo de Neagle?

Aumentaría la eficiencia de la aplicación al disminuir la cantidad de paquetes que deben enviarse.

## Conclusiones

Esta práctica nos sirvió para revisar el uso del multidifusión, de igual manera nos explica de forma práctica la gran diferencia que hay entre unidifusión y multidifusión, también se continúa reforzando el aprendizaje de los buffer y como impacta el algoritmo de Neagle en estos.

## Referencias

[1] Wikipedia. (2022). “Datagrama”. [En línea]. Recuperado de https://es.wikipedia.org/wiki/Datagrama

[2] Equipo de Expertos en Ciencia. (2017 de febrero). “¿Qué es un datagrama?” [En línea]. Recuperado de https://www.universidadviu.com/es/actualidad/nuestros-expertos/que-es-un-datagrama

[3] Ecured. (s.f). “Sopa de letras”. [En línea]. Recuperado de https://www.ecured.cu/Sopa_de_letras