Instituto Politécnico Nacional.

Escuela Superior De Cómputo.

Materia:

Aplicaciones Para Comunicación En Red.

Tema:

Reporte Práctica 2

Profesor:

Axel Ernesto Moreno Cervantes.

Alumno:

Luis Enrique Alvarado Rojas.

Mario Alberto Miranda Sandoval.

Grupo:

3CM5

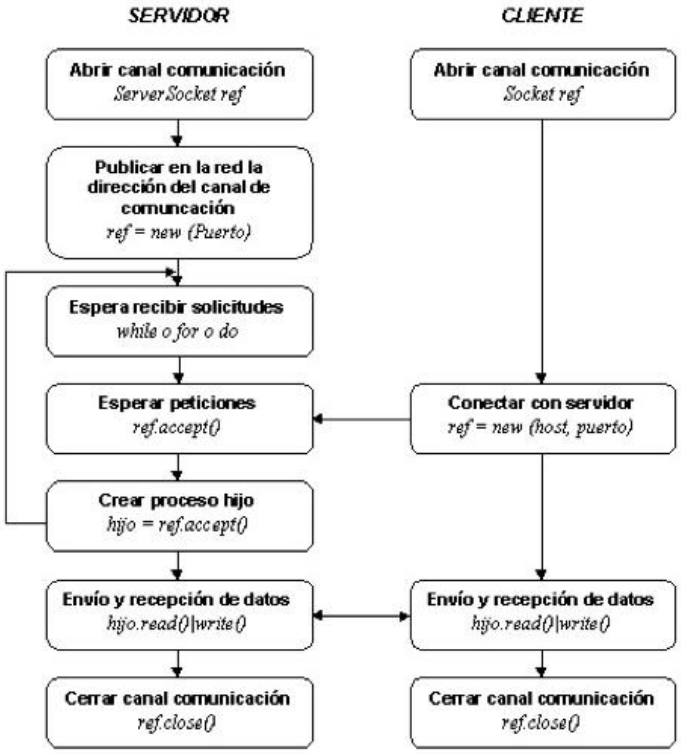
Introducción.

En la actualidad, muchos de los procesos que se ejecutan en una computadora requieren obtener o enviar información a otros procesos que se localizan en una computadora diferente. Para lograr esta comunicación se utilizan los protocolos de comunicación TCP y UDP.

El protocolo TCP (Transmission Control Protocol) establece un conducto de comunicación punto a punto entre dos computadoras, es decir, cuando se requiere la transmisión de un flujo de datos entre dos equipos, el protocolo TCP establece un conducto exclusivo entre dichos equipos por el cual los datos serán transmitidos y este perdurará hasta que la transmisión haya finalizado, gracias a esto TCP garantiza que los datos enviados de un extremo de la conexión lleguen al otro extremo y en el mismo orden en que fueron enviados. Las características que posee TCP hacen que el protocolo sea conocido como un protocolo orientado a conexión.

Los sockets son una forma de comunicación entre procesos que se encuentran en diferentes máquinas de una red, los sockets proporcionan un punto de comunicación por el cual se puede enviar o recibir información entre procesos.

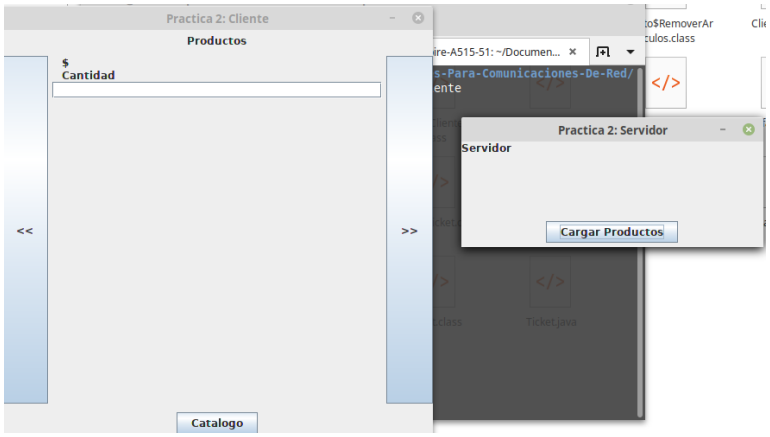
Los sockets tienen un ciclo de vida dependiendo si son sockets de servidor, que esperan a un cliente para establecer una comunicación, o socket cliente que busca a un socket de servidor para establecer la comunicación.



Desarrollo.

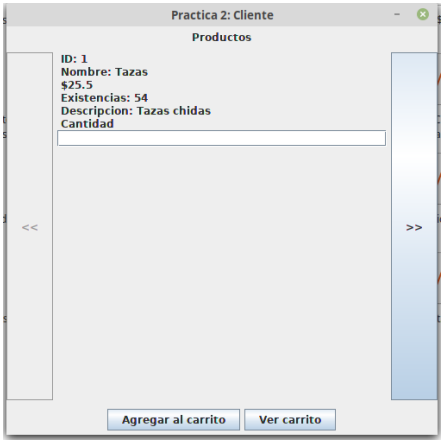
Para esta práctica se debe simular un sistema de compras en línea, donde el servidor guarde los artículos, gestione sus existencias y mande un ticket de compra, mientras que el usuario puede añadir artículos al carrito, removerlos y hacer su compra final.

Para esta práctica se genero la interfaz cliente y servidor como sigue:

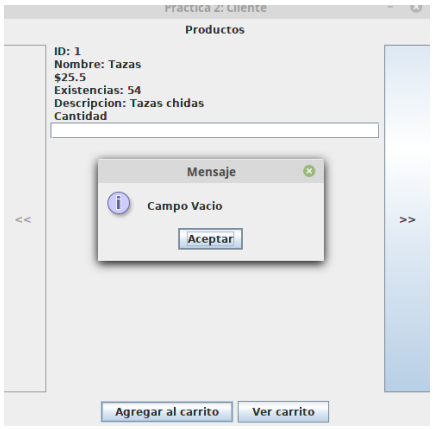


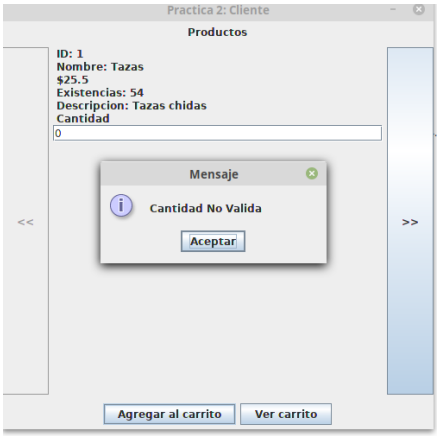
Interfaz de cliente y servidor.

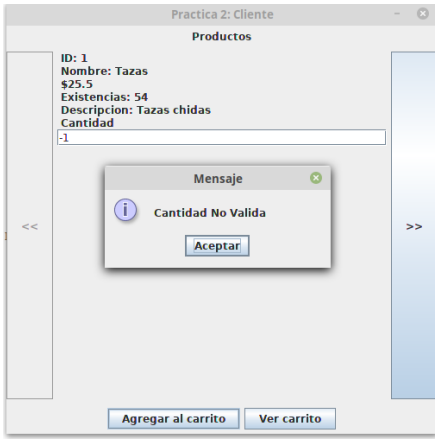
Pruebas.



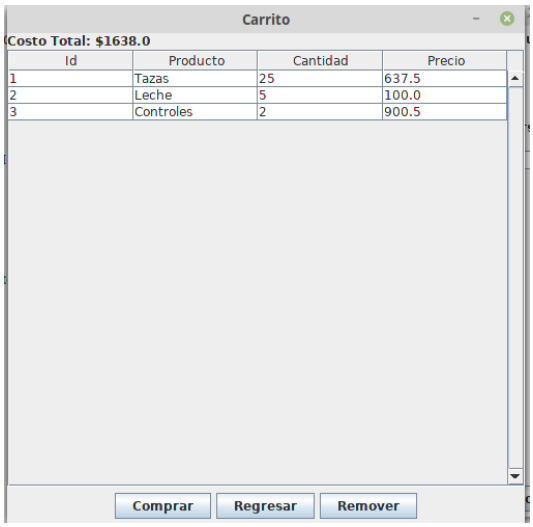
Cliente con productos cargados.

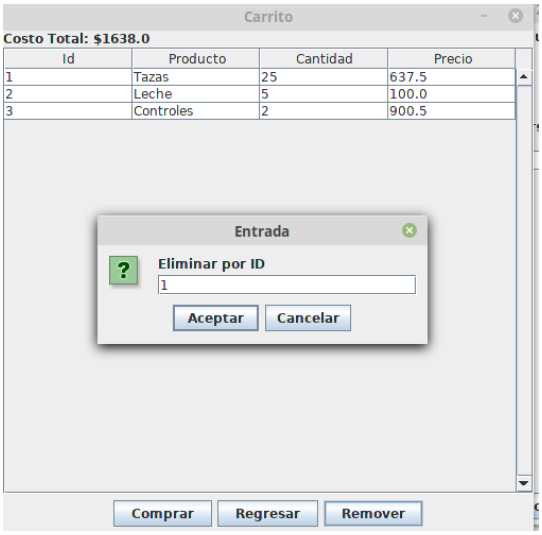


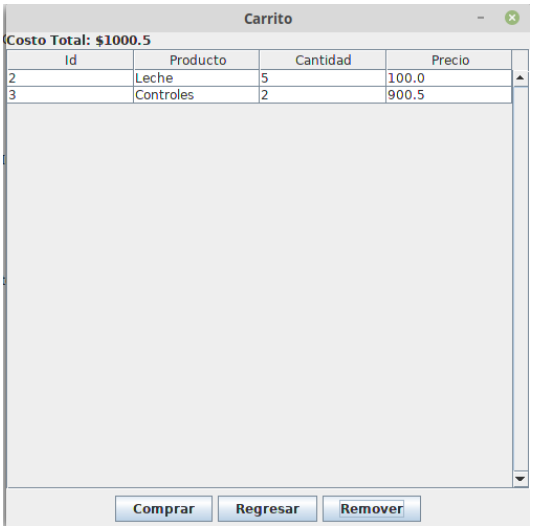




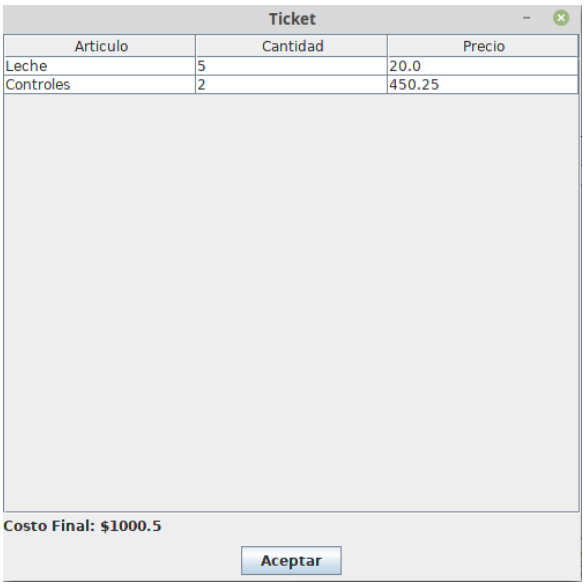
Validaciones.



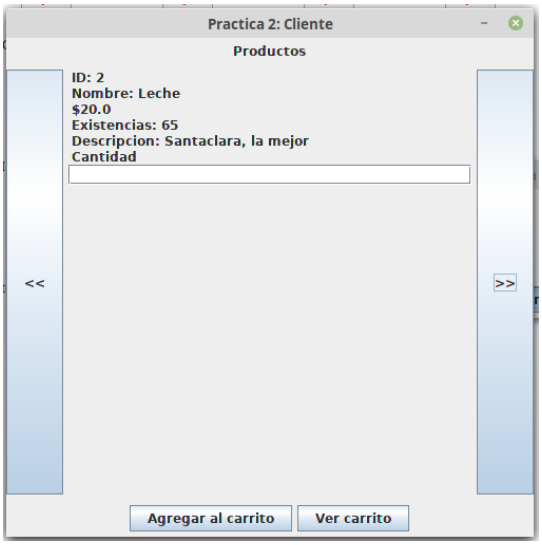


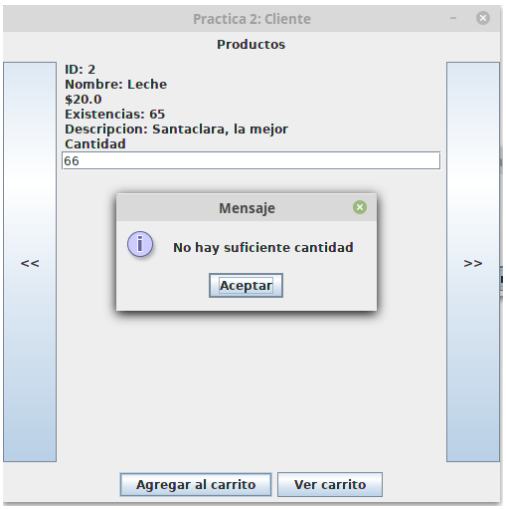


Cliente manipulando el carrito.



Ticket generado.





Un segundo cliente se conecta, las existencias se actualizan, y las validaciones continúan.

Código

Producto.java

```java
import java.io.Serializable;

public class Producto implements Serializable {
    private static final long serialVersionUID = 3L;
    public Producto(int id, String nombre, double precio, int existencias, String descripcion, boolean promocion, String imagen) {
        this.id = id;
        this.nombre = nombre;
        this.precio = precio;
        this.existencias = existencias;
        this.descripcion = descripcion;
        this.promocion = promocion;
        this.imagen = imagen;
        cantidad = 0;
    }

    public int getID() { return id; }
    public String getNombre() { return nombre; }
    public double getPrecio() { return precio; }
    public int getExistencias() { return existencias; }
    public String getDescripcion() { return descripcion; }
    public boolean getPromocion() { return promocion; }
    public String getImagen() { return imagen; }
    public int getCantidad() { return cantidad; }

    public void setID(int id) { this.id = id; }
    public void setNombre(String nombre) { this.nombre = nombre; }
    public void setPrecio(double precio) { this.precio = precio; }
    public void setExistencias(int existencias) { this.existencias = existencias; }
    public void setDescripcion(String descripcion) { this.descripcion = descripcion; }
```

```
30.     public void setPromocion(boolean promocion) { this.promocion = promocion; }
31.     public void setImagen(String imagen) { this.imagen = imagen; }
32.     public void setCantidad(int cantidad) { this.cantidad = cantidad; }
33.
34.     private int id;
35.     private String nombre;
36.     private double precio;
37.     private int existencias;
38.     private String descripcion;
39.     private boolean promocion;
40.     private String imagen;
41.     private int cantidad;
42. }
```

Last.java

```
1.  import java.io.Serializable;
2.  import java.util.ArrayList;
3.
4.  public class Last implements Serializable {
5.      private static final long serialVersionUID = 8L;
6.      public Last(int ac, ArrayList<Producto> misProductos) {
7.          this.ac = ac;
8.          this.misProductos = misProductos;
9.      }
10.
11.     public int getAccion() { return ac; }
12.     public ArrayList<Producto> getProducto() { return misProductos; }
13.
14.     public void setAccion(int ac) { this.ac = ac; }
15.     public void setProducto(ArrayList<Producto> misProductos) { this.misProductos
    = misProductos; }
16.
17.     private int ac;
18.     private ArrayList<Producto> misProductos;
19. }
```

Ticket.java

```
1.  import java.io.Serializable;
2.  import java.util.ArrayList;
3.
4.  public class Ticket implements Serializable {
5.      private static final long serialVersionUID = 9L;
6.      public Ticket(ArrayList<Producto> productos, double precio) {
7.          this.productos = productos;
8.          this.precio = precio;
9.      }//constructor
10.
11.     public ArrayList<Producto> getProductos() { return productos; }
12.     public double getPrecio() { return precio; }
13.
14.     private ArrayList<Producto> productos;
15.     private double precio;
16. }//clase
```

InterfazCliente.java

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class InterfazCliente extends JFrame{
    private static final long serialVersionUID = 1L;
    public InterfazCliente() {
        setBounds(450, 150, 500, 500);
        setTitle("Practica 2: Cliente");
        setResizable(false);

        //Creacion de componentes
        panelPrincipal = new JPanel();
        panelInferior = new JPanel();
        panelSuperior = new JPanel();
        panelCentral = new JPanel();
        panelCentralCentral = new JPanel();
        botonAnterior = new JButton("<<");
        botonSiguiente = new JButton(">>");
        agregarCarrito = new JButton("Agregar al carrito");
        agregarCarrito.setVisible(false);
        verCarrito = new JButton("Ver carrito");
        verCarrito.setVisible(false);
        pedirCatalogo = new JButton("Catalogo");
        cliente = new JLabel("Productos");
        imagen = new JLabel("");
        nombre = new JLabel("");
        id = new JLabel("");
        precio = new JLabel("$");
        existencia = new JLabel("");
        descripcion = new JLabel("");
        texto = new JTextField(4);
        miCarrito = new Carrito();
        i = 0;

        //Colocando a la escucha
        pedirCatalogo.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                pedirCatalogo.setVisible(false);
                verCarrito.setVisible(true);
                agregarCarrito.setVisible(true);
                miCliente = new Cliente(PUERTO, HOST);
                misProductos = miCliente.recibirCatalogo();
                botonAnterior.setEnabled(false);
                dibujo(misProductos[0]);
            }
        });

        botonSiguiente.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                if(contador == 9)
                    botonSiguiente.setEnabled(false);
                else {
                    dibujo(misProductos[++contador]);
                    botonSiguiente.setEnabled(true);
                    botonAnterior.setEnabled(true);
                }
            }
        });
```

```
60.
61.         botonAnterior.addActionListener(new ActionListener() {
62.             public void actionPerformed(ActionEvent e) {
63.                 if (contador > 0) {
64.                     botonAnterior.setEnabled(true);
65.                     botonSiguiente.setEnabled(true);
66.                     dibujo(misProductos[--contador]);
67.                 } else if(contador == 9){
68.                     botonSiguiente.setEnabled(false);
69.                 } else {
70.                     botonAnterior.setEnabled(false);
71.                 }
72.             }
73.         });
74.
75.         agregarCarrito.addActionListener(new ActionListener() {
76.             public void actionPerformed(ActionEvent e) {
77.                 if(texto.getText().equals("")){
78.                     JOptionPane.showMessageDialog(InterfazCliente.this, "Campo Vac
    io");
79.                 }
80.                 else {
81.                 cantidad = Integer.parseInt(texto.getText());
82.                 int ids = Integer.parseInt(id.getText().substring(4));
83.                 if(cantidad > misProductos[ids - 1].getExistencias()) {
84.                     JOptionPane.showMessageDialog(InterfazCliente.this, "No hay su
    ficiente cantidad");
85.                 } else if(cantidad <= 0) {
86.                     JOptionPane.showMessageDialog(InterfazCliente.this, "Cantidad
    No Valida");
87.                 }
88.                 else {
89.                     misProductos[ids - 1].setExistencias(misProductos[ids - 1].get
    Existencias() - cantidad);
90.                     misProductos[ids - 1].setCantidad(misProductos[ids - 1].getCan
    tidad() + cantidad);
91.                     //System.out.println(misProductos[ids - 1].getNombre() + " : "
     + misProductos[ids - 1].getExistencias());
92.                     existencia.setText("Existencias: " + misProductos[ids - 1].get
    Existencias());
93.                     miCarrito.crearCarrito(misProductos[ids - 1], ++i);
94.                 }
95.                 texto.setText("");
96.             }
97.         }
98.         });
99.
100.                verCarrito.addActionListener(new ActionListener(){
101.                    public void actionPerformed(ActionEvent arg0) {
102.                        miCarrito.setVisible(true);
103.                    }
104.                });
105.
106.                //Propiedades de los componentes
107.                panelPrincipal.setLayout(new BorderLayout(5, 5));
108.                panelCentral.setLayout(new BorderLayout(3, 3));
109.                panelCentralCentral.setLayout(new BoxLayout(this.panelCentralCentr
    al, BoxLayout.Y_AXIS));
110.
111.                //Adicion de los componentes
112.                panelCentralCentral.add(id);
```

```
113.                panelCentralCentral.add(nombre);
114.                panelCentralCentral.add(precio);
115.                panelCentralCentral.add(existencia);
116.                panelCentralCentral.add(descripcion);
117.                panelCentralCentral.add(new JLabel("Cantidad"));
118.                panelCentralCentral.add(texto);
119.                panelCentral.add(imagen, BorderLayout.NORTH);
120.                panelCentral.add(panelCentralCentral, BorderLayout.NORTH);
121.                panelInferior.add(pedirCatalogo);
122.                panelInferior.add(agregarCarrito);
123.                panelInferior.add(verCarrito);
124.                panelSuperior.add(cliente);
125.                panelPrincipal.add(panelSuperior, BorderLayout.NORTH);
126.                panelPrincipal.add(panelCentral, BorderLayout.CENTER);
127.                panelPrincipal.add(botonAnterior, BorderLayout.WEST);
128.                panelPrincipal.add(botonSiguiente, BorderLayout.EAST);
129.                panelPrincipal.add(panelInferior, BorderLayout.SOUTH);
130.                add(panelPrincipal);
131.
132.                setDefaultCloseOperation(3);
133.                setVisible(true);
134.            }//Constructor
135.
136.            private void dibujo(Producto producto) {
137.                id.setText("ID: " + producto.getID());
138.                imagen.setText(producto.getImagen());
139.                nombre.setText("Nombre: " + producto.getNombre());
140.                precio.setText("$" + producto.getPrecio());
141.                existencia.setText("Existencias: " + producto.getExistencias());
142.                descripcion.setText("Descripcion: " + producto.getDescripcion());

143.            }//dibujo
144.
145.            public static void main(String[] args) {
146.                new InterfazCliente();
147.            }//main
148.
149.            //Bloque de instancias
150.            private JPanel panelPrincipal;
151.            private JPanel panelInferior;
152.            private JPanel panelSuperior;
153.            private JPanel panelCentral;
154.            private JPanel panelCentralCentral;
155.            private JButton botonSiguiente;
156.            private JButton botonAnterior;
157.            private JButton agregarCarrito;
158.            private JButton verCarrito;
159.            private JButton pedirCatalogo;
160.            private JLabel cliente;
161.            private JLabel id;
162.            private final int PUERTO = 9999;
163.            private final String HOST = "127.0.0.1";
164.            private Cliente miCliente;
165.            private Producto[] misProductos;
166.            private JLabel imagen, nombre, precio, existencia, descripcion;
167.            private int contador = 0;
168.            private int i = 0;
169.            private Carrito miCarrito;
170.            private JTextField texto;
171.            private int cantidad;
172.        }//Clase
```

InterfazServidor.java

```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class InterfazServidor extends JFrame {
    private static final long serialVersionUID = 2L;

    public InterfazServidor() {
        setBounds(550, 150, 350, 150);
        setTitle("Practica 2: Servidor");
        setResizable(false);

        // Creacion de componentes
        panelPrincipal = new JPanel();
        panelInferior = new JPanel();
        cargarProductos = new JButton("Cargar Productos");
        online = new JButton("Online");
        online.setVisible(false);
        servidor = new JLabel("Servidor");
        misProductos = new Producto[10];
        productos = new GestionProductosServidor(misProductos);

        // Propiedades de los componentes
        panelPrincipal.setLayout(new BorderLayout(5, 5));

        // Colocando a la escucha
        cargarProductos.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                misProductos = productos.generarProductos();
                online.setVisible(true);
                cargarProductos.setVisible(false);

                System.out.println("Productos Cargados");
                for(int i = 0; i < misProductos.length; i += 1)
                    System.out.println(misProductos[i].getNombre() +" " +
                    misProductos[i].getExistencias());
            }
        });

        online.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                miServidor = new Servidor(PUERTO, misProductos);
                try {
                    miServidor.online();
                } catch (ClassNotFoundException e1) {
                    e1.printStackTrace();
                }
            }
        });

        //A�adiendo componentes
        panelInferior.add(cargarProductos);
        panelInferior.add(online);
        panelPrincipal.add(servidor, BorderLayout.NORTH);
        //panelPrincipal.add(new JScrollPane(tablaProductos), BorderLayout.CENTER)
;
        panelPrincipal.add(panelInferior, BorderLayout.SOUTH);
        add(panelPrincipal);

```

```
59.         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
60.         setVisible(true);
61.     }//Constructor
62.
63.     public static void main(String[] args) {
64.         new InterfazServidor();
65.     }//main
66.
67.     //Bloque de instancias
68.     private JPanel panelPrincipal;
69.     private JPanel panelInferior;
70.     private JButton cargarProductos;
71.     private JButton online;
72.     private JLabel servidor;
73.     private GestionProductosServidor productos;
74.     private Producto[] misProductos;
75.     private final int PUERTO = 9999;
76.     private Servidor miServidor;
77. }//clase
```

InterfazTicket.java

```
1.  import javax.swing.*;
2.  import javax.swing.table.*;
3.  import java.awt.event.*;
4.  import java.util.ArrayList;
5.  import java.awt.*;
6.
7.  public class InterfazTicket extends JFrame {
8.      private static final long serialVersionUID = 11L;
9.      public InterfazTicket() {
10.         setTitle("Ticket");
11.         setBounds(450, 150, 500, 500);
12.         setResizable(false);
13.
14.         panelPrincipal = new JPanel();
15.         panelInferior = new JPanel();
16.         panelCostoFinal = new JPanel();
17.         aceptar = new JButton("Aceptar");
18.         tabla = new JTable();
19.         costo = new JLabel("Costo Final: $");
20.         modelo = (DefaultTableModel) tabla.getModel();
21.         modelo.addColumn("Articulo");
22.         modelo.addColumn("Cantidad");
23.         modelo.addColumn("Precio");
24.
25.         aceptar.addActionListener(new ActionListener() {
26.             public void actionPerformed(ActionEvent e) {
27.                 modelo.setRowCount(0);
28.                 costo.setText("Costo Final: $");
29.                 setVisible(false);
30.             }
31.         });
32.
33.         panelPrincipal.setLayout(new BorderLayout(5, 5));
34.         panelCostoFinal.setLayout(new BorderLayout(3, 3));
35.         panelInferior.add(aceptar);
36.         panelCostoFinal.add(new JScrollPane(tabla), BorderLayout.CENTER);
37.         panelCostoFinal.add(costo, BorderLayout.SOUTH);
38.         panelPrincipal.add(panelCostoFinal, BorderLayout.CENTER);
```

```
        panelPrincipal.add(panelInferior, BorderLayout.SOUTH);
        add(panelPrincipal);

        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    }

    public void crear(Ticket t) {
        ArrayList <Producto> miProducto = new ArrayList<>();
        miProducto = t.getProductos();
        for(Producto p : miProducto) {
            modelo.addRow(new Object[]{p.getNombre(), p.getCantidad(), p.getPrecio
    ()});
        }

        costo.setText("Costo Final: $" + t.getPrecio());
    }

    private JPanel panelPrincipal;
    private JPanel panelInferior;
    private JPanel panelCostoFinal;
    private JTable tabla;
    private DefaultTableModel modelo;
    private JButton aceptar;
    private JLabel costo;
}
```

GestionProductosServidor.java

```
public class GestionProductosServidor {
    public GestionProductosServidor(Producto[] misProductos) {
        this.misProductos = misProductos;
    }

    public Producto[] generarProductos() {
        for(int i = 0; i < misProductos.length; i += 1)
            misProductos[i] = new Producto(id[i], nombre[i], precio[i], generarExi
    stencias(), descripcion[i], false, imagen[i]);

        return misProductos;
    }

    public int generarExistencias() {
        return (int)((Math.random() * ((100 - 50) + 1)) + 50);
    }

    private Producto[] misProductos;
    private int[] id = {1,2,3,4,5,6,7,8,9,10};
    private String[] nombre = {"Tazas", "Leche", "Audifonos", "Controles", "Juguet
    es", "Tortillas", "Gorras", "Celulares", "Pan", "Cafe"};
    private double[] precio = {25.50, 20.0, 850.0, 450.25, 360.95, 15.25, 200.56,
    1500.58, 6.70, 29.99};
    private String[] descripcion = {
        "Tazas chidas",
        "Santaclara, la mejor",
        "Skullcandy ink'd wireless 2.0",
        "Control universal",
        "Juguete variado",
        "Tortillas frescas, no existe duras",
        "Gorras diversos modelos",
        "Celulares pa'l kevin",
```

```
        "Pan hecho en casa",
        "Cafe barato pero bueno"
    };
    private String[] imagen = {
        "Tazas chidas",
        "Leche Santaclara",
        "Audifonos Skullcandy",
        "Control universal",
        "Juguetes diversos",
        "Tortillas",
        "Gorras bordadas",
        "Celulares Chinos",
        "Pan de casa",
        "Cafe"
    };
}
```

Cliente.java

```
import java.io.*;
import java.net.*;
import java.util.*;

public class Cliente {
    public Cliente(int puerto, String host) {
        this.puerto = puerto;
        this.host = host;
    }

    public Producto[] recibirCatalogo() {
        try {
            cliente = new Socket(this.host, this.puerto);
            Last l = new Last(1, null);
            ObjectOutputStream oos = new ObjectOutputStream(cliente.getOutputStrea
m());
            oos.writeObject(l);
            oos.flush();
            ObjectInputStream ois = new ObjectInputStream(cliente.getInputStream()
);
            misProductos = (Producto[]) ois.readObject();
            ois.close();
            oos.close();

            for(Producto p : misProductos)
                System.out.println(p.getNombre() + "," + p.getExistencias());

            return misProductos;
        } catch(Exception e) {
            e.printStackTrace();
            return null;
        }//try/catch
    }//recibirCatalogo

    public void hacerCompra(ArrayList<Producto> productosFinal) {
        try {
            cliente = new Socket(this.host, this.puerto);
            Last l = new Last(2, productosFinal);
            ObjectOutputStream oos = new ObjectOutputStream(cliente.getOutputStrea
m());

```

```
39.             oos.writeObject(l);
40.             oos.flush();
41.
42.             ObjectInputStream ois = new ObjectInputStream(cliente.getInputStream()
    );
43.             Ticket t = (Ticket) ois.readObject();
44.
45.             //System.out.println(t.getPrecio());
46.
47.             ois.close();
48.             oos.close();
49.             cliente.close();
50.
51.             intTicket = new InterfazTicket();
52.             intTicket.crear(t);
53.             intTicket.setVisible(true);
54.         } catch(Exception e) { e.printStackTrace(); }
55.     }//hacerCompra
56.
57.     private int puerto;
58.     private String host;
59.     private Producto[] misProductos;
60.     private Socket cliente;
61.     private InterfazTicket intTicket;
62. }
```

Carrito.java

```
1.  import java.util.*;
2.  import javax.swing.*;
3.  import java.awt.*;
4.  import java.awt.event.*;
5.  import javax.swing.table.*;
6.
7.
8.  public class Carrito extends JFrame {
9.      private static final long serialVersionUID = 4L;
10.
11.     public Carrito() {
12.         setTitle("Carrito");
13.         setBounds(450, 150, 500, 500);
14.         setResizable(false);
15.
16.         //Creacion Componentes
17.         panelPrincipal = new JPanel();
18.         panelBotones = new JPanel();
19.         comprar = new JButton("Comprar");
20.         regresar = new JButton("Regresar");
21.         remover = new JButton("Remover");
22.         costo = new JLabel("Costo Total: $");
23.         miCarrito = new ArrayList<>();
24.         tablaProductos = new JTable();
25.         modelo = (DefaultTableModel) tablaProductos.getModel();
26.         modelo.addColumn("Id");
27.         modelo.addColumn("Producto");
28.         modelo.addColumn("Cantidad");
29.         modelo.addColumn("Precio");
30.
31.         regresar.addActionListener(new ActionListener() {
32.             public void actionPerformed(ActionEvent e) {
```

```
                setVisible(false);
            }
        });

        comprar.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                miCliente = new Cliente(PUERTO, HOST);
                miCliente.hacerCompra(miCarrito);
                miCarrito.clear();
                modelo.setRowCount(0);
                costo.setText("Costo Total: $");
                setVisible(false);
            }
        });

        remover.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                int idFila = Integer.parseInt(JOptionPane.showInputDialog(panelPri
ncipal, "Eliminar por ID"));
                double precioRestar = (double) modelo.getValueAt(idFila - 1, 3);
                //System.out.println(precioRestar);
                costo.setText("Costo Total: $" + (precio - precioRestar));
                miCarrito.remove(idFila - 1);
                modelo.removeRow(idFila - 1);
            }
        });

        panelPrincipal.setLayout(new BorderLayout(1, 1));

        panelBotones.add(comprar);
        panelBotones.add(regresar);
        panelBotones.add(remover);
        panelPrincipal.add(costo, BorderLayout.NORTH);
        panelPrincipal.add(new JScrollPane(tablaProductos, JScrollPane.VERTICAL_SC
ROLLBAR_ALWAYS, JScrollPane.HORIZONTAL_SCROLLBAR_NEVER), BorderLayout.CENTER);
        panelPrincipal.add(panelBotones, BorderLayout.SOUTH);
        add(panelPrincipal);

        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setVisible(false);
    }//Constructor

    public void crearCarrito(Producto p, int i) {
       modelo.addRow(new Object[]{i, p.getNombre(), p.getCantidad(), (p.getCantida
d() * p.getPrecio())});
        calcularPrecio(p);
        miCarrito.add(p);
    }//crearCarrito

    private void calcularPrecio(Producto producto) {
        precio += (producto.getPrecio() * producto.getCantidad());
        costo.setText("Costo Total: $" + precio);
    }


    private JPanel panelPrincipal;
    private JPanel panelBotones;
    private JButton comprar;
    private JButton regresar;
    private JButton remover;
    private JLabel costo;
```

```
    private double precio = 0;
    private Cliente miCliente;
    private final int PUERTO = 9999;
    private final String HOST = "127.0.0.1";
    private ArrayList<Producto> miCarrito;
    private DefaultTableModel modelo;
    private JTable tablaProductos;
}//clase
```

Servidor.java

```
import java.io.*;
import java.net.*;
import java.util.ArrayList;

public class Servidor {
    public Servidor(int puerto, Producto[] misProductos) {
        this.puerto = puerto;
        this.misProductos = misProductos;
    }// Constructor

    public void online() throws ClassNotFoundException {
        try {
            servidor = new ServerSocket(this.puerto);
            System.out.println("Servidor Activo");
            for (;;) {
                cliente = servidor.accept();
                ObjectInputStream ois = new ObjectInputStream(cliente.getInputStre
am());
                Last l = (Last) ois.readObject();
                int accion = l.getAccion();
                if (accion == 1) {
                    //System.out.println("Mandando catalogo");
                    mandarCatalogo();
                } else if (accion == 2) {
                    //System.out.println("Procesando Compra");
                    registrarCompra(l);
                }
                ois.close();
                cliente.close();
            } // for
        } catch (IOException e) {
            e.printStackTrace();
        }
    }// online

    private void mandarCatalogo() {
        try {
            ObjectOutputStream oos = new ObjectOutputStream(cliente.getOutputStrea
m());
            oos.writeObject(misProductos);
            oos.flush();
            oos.close();
        } catch (IOException e) {
            e.printStackTrace();
        } // try/catch
    }// mandarCatalogo

    private void registrarCompra(Last l) {
        ArrayList<Producto> productoRecibido = l.getProducto();
```

```
48.         /*for (Producto p : productoRecibido)
49.             System.out.println(p.getNombre());*/
50.
51.         actualizarExistencia(productoRecibido);
52.     }// registrarCompra
53.
54.     private void actualizarExistencia(ArrayList<Producto> productoRecibido) {
55.         for(Producto p : productoRecibido) {
56.             if(p.getID() == 1) {
57.                 misProductos[0].setExistencias(misProductos[0].getExistencias() -
    p.getCantidad());
58.             } else if(p.getID() == 2) {
59.                 misProductos[1].setExistencias(misProductos[1].getExistencias() -
    p.getCantidad());
60.             } else if(p.getID() == 3) {
61.                 misProductos[2].setExistencias(misProductos[2].getExistencias() -
    p.getCantidad());
62.             } else if(p.getID() == 4) {
63.                 misProductos[3].setExistencias(misProductos[3].getExistencias() -
    p.getCantidad());
64.             } else if(p.getID() == 5) {
65.                 misProductos[4].setExistencias(misProductos[4].getExistencias() -
    p.getCantidad());
66.             } else if(p.getID() == 6) {
67.                 misProductos[5].setExistencias(misProductos[5].getExistencias() -
    p.getCantidad());
68.             } else if(p.getID() == 7) {
69.                 misProductos[6].setExistencias(misProductos[6].getExistencias() -
    p.getCantidad());
70.             } else if(p.getID() == 8) {
71.                 misProductos[7].setExistencias(misProductos[7].getExistencias() -
    p.getCantidad());
72.             } else if(p.getID() == 9) {
73.                 misProductos[8].setExistencias(misProductos[8].getExistencias() -
    p.getCantidad());
74.             } else if(p.getID() == 10) {
75.                 misProductos[9].setExistencias(misProductos[9].getExistencias() -
    p.getCantidad());
76.             }
77.         }
78.
79.         generarTicket(productoRecibido);
80.
81.         System.out.println("Productos Actualizados");
82.         for(Producto p : misProductos)
83.             System.out.println(p.getNombre() + " : " + p.getExistencias());
84.     }// actualizarExistencia
85.
86.     private void generarTicket(ArrayList<Producto> productoRecibido) {
87.         double precio = 0;
88.         for (Producto p : productoRecibido)
89.             precio += p.getPrecio() * p.getCantidad();
90.
91.         Ticket t = new Ticket(productoRecibido, precio);
92.         try {
93.             ObjectOutputStream oos = new ObjectOutputStream(cliente.getOutputStrea
    m());
94.             oos.writeObject(t);
95.             oos.flush();
96.             oos.close();
97.         } catch (IOException e) {
```

```
98.             e.printStackTrace();
99.         }
100.                //System.out.println(t.getPrecio());
101.            }//generarTicket
102.
103.            private int puerto;
104.            private ServerSocket servidor;
105.            private Socket cliente;
106.            private Producto[] misProductos;
107.        }//Clase
```

Dificultades Encontradas.

Sobre esta práctica la mayor dificultad, fue gestionar y sincronizar las interfaces de acuerdo a los sockets, mientras que la implementación en los sockets fue relativamente sencilla, el único inconveniente era mandar más de un solo objeto, por lo que se creo la clase Last, para poder mandar un objeto dentro de esta y a su vez poder mandar nuestro objeto con distintos descriptores.

Conclusiones.

Luis Enrique Rojas Alvarado.

En esta práctica pudimos observar en una aplicación más del "uso cotidiano" las diferentes aplicaciones de los sockets y la importancia de ellos, puesto que saber cómo funcionan y qué hacen internamente las aplicaciones de este uso.

Personalmente pienso que los sockets no solo son de uso académico, si no que en realidad se usan en el mundo real para hacer sistemas en red que puedan ayudar a una pequeña empresa a administrar sus productos o cualquier otra cosa. Sin mencionar que parece lo más natural posible.

Miranda Sandoval Mario Alberto.

Sobre esta práctica, a diferencia de la anterior pude obtener demasiadas conclusiones, acerca del como funcionan los sockets y además ver como meter objetos dentro de otros objetos para mandarlos dentro de un solo paquete es un caso practico y funcional, solo con la desventaja del uso de más memoria.

El mover interfaces mientras se sincroniza con el servidor, mientras a su vez mantener las validaciones respectivas en el servidor fueron las que conllevaron más tiempo, mientras que la implementación de los sockets fue fácil.