3CM5

Instituto Politécnico Nacional

ESCOM

Redes de Computadoras

Práctica 1

Drag & Drop

Alumno:

Naranjo Miranda Javier Said

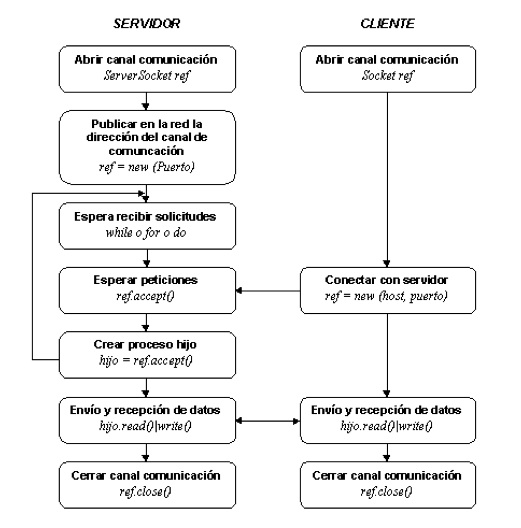
Flores García Sergio

Grupo: 3CM5

**Introducción**

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 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.



El envío de archivos a través de la red es una característica importante para la gran mayoría de las aplicaciones que hoy día se utilizan (blogs, redes sociales, mensajería instantánea, Declaración de impuestos, educación en línea, etc.), sin embargo, no todas las aplicaciones disponibles permiten el envío de archivos de gran tamaño (p.e. El correo electrónico no permite enviar archivos de más de 10 o 20 MB). Esto hace necesario el desarrollo de aplicaciones que permitan transferir archivos sin importar el tamaño de éstos.

**Desarrollo**

**Código**

Envia.java

package archivo.c.s;

import javax.swing.JFileChooser;

import java.net.\*;

import java.io.\*;

import java.util.List;

public class Envia {

private String ruta;

private String nombre;

private long tam;

private long enviados;

public Envia() {

this.ruta = "";

this.nombre = "";

this.tam = 0;

this.enviados = 0;

}

public void enviar\_archivo(File f[]){

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

try{

String host="127.0.0.1";

int pto=8000,porcentaje=0,n;

Socket cl=new Socket(host,pto);

System.out.println("Conexion establecida, comienza envio del archivo");

DataOutputStream dos=new DataOutputStream(cl.getOutputStream());

DataInputStream dis=new DataInputStream(new FileInputStream(f[i].getAbsolutePath()));

nombre=f[i].getName();

dos.writeUTF(nombre);

dos.flush();

dos.writeLong(f[i].length());

dos.flush();

enviados=0;

while(enviados<f[i].length()){

byte[] b = new byte[1500];

n= dis.read(b);

dos.write(b,0,n);

dos.flush();

enviados = enviados+n;

porcentaje=(int)((enviados\*100)/f[i].length());

System.out.println("Se ha transmitido el "+porcentaje+"%");

}

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

}

catch(Exception e){

e.printStackTrace();

}

}

}

public void enviar\_archivo(List<File> f){

for(File file:f){

if (file.isDirectory()){

enviar\_archivo(file.listFiles());

System.out.println("Soy carpeta no se que hacer jaja ");

}else { // Manda un solo archivo

try{

String host="127.0.0.1";

int pto=8000,porcentaje=0,n;

Socket cl=new Socket(host,pto);

System.out.println("Conexion establecida, comienza envio del archivo");

DataOutputStream dos=new DataOutputStream(cl.getOutputStream());

DataInputStream dis=new DataInputStream(new FileInputStream(file.getAbsolutePath()));

nombre=file.getName();

dos.writeUTF(nombre);

dos.flush();

dos.writeLong(file.length());

dos.flush();

enviados=0;

while(enviados<file.length()){

byte[] b = new byte[1500];

n= dis.read(b);

dos.write(b,0,n);

dos.flush();

enviados = enviados+n;

porcentaje=(int)((enviados\*100)/file.length());

System.out.println("Se ha transmitido el "+porcentaje+"%");

}

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

}

catch(Exception e){

e.printStackTrace();

}

}

}

}

}

Recibe.java

package archivo.c.s;

import java.net.\*;

import java.io.\*;

public class Recibe {

public static void main(String[] args) {

try{

int pto=8000,n;

ServerSocket s = new ServerSocket(pto);

s.setReuseAddress(true);

System.out.println("Servicio Iniciado");

for(;;){

Socket cl=s.accept();

System.out.println("Cliente conectado desde "+cl.getInetAddress());

DataInputStream d = new DataInputStream(cl.getInputStream());

String nombre = d.readUTF();

long tam = d.readLong();

DataOutputStream a = new DataOutputStream(new FileOutputStream(nombre));

long recibidos =0;

while(recibidos<tam){

byte[] b = new byte[1500];

n= d.read(b);

a.write(b,0,n);

a.flush();

recibidos=recibidos+n;

}

System.out.println("Archivo Recibido");

a.close();

d.close();

cl.close();

}

}catch(Exception e){

e.printStackTrace();

}

}

}

Clase ListTransfer.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 Vista;

import archivo.c.s.Envia;

import java.util.List;

import java.awt.datatransfer.\*;

import java.io.File;

import javax.swing.\*;

public class ListaTransfer extends TransferHandler{

public boolean canImport(TransferHandler.TransferSupport info){

if(!info.isDataFlavorSupported(DataFlavor.javaFileListFlavor)){ //Si no es del tipo File

System.out.println("No es archivo");

return false;

}

JList.DropLocation dl = (JList.DropLocation)info.getDropLocation(); //Si esta fuera del JList

if(dl.getIndex()==-1){

System.out.println("Estas fuera del JList");

return false;

}

return true;

}

public boolean importData(TransferHandler.TransferSupport info){

if(!info.isDrop()){

System.out.println("No es Drop");

return false;

}

if(!info.isDataFlavorSupported(DataFlavor.javaFileListFlavor)){

System.out.println("No es un archivo");

return false;

}

JList lista = (JList)info.getComponent();

DefaultListModel listModel = new DefaultListModel();

lista.setModel(listModel);

List <File> droppedFiles =null;

try{

droppedFiles = (List<File>) info.getTransferable()

.getTransferData(DataFlavor.javaFileListFlavor);

Envia en=new Envia();

en.enviar\_archivo(droppedFiles);

// Manda las carpetas recursivamente

//if (file.isDirectory()) en.enviar\_archivo(file);

// else en.enviar\_archivo(file); // Manda un solo archivo

for (File file : droppedFiles) {

listModel.addElement(file.getName());

}

}catch(Exception e){

return false;

}

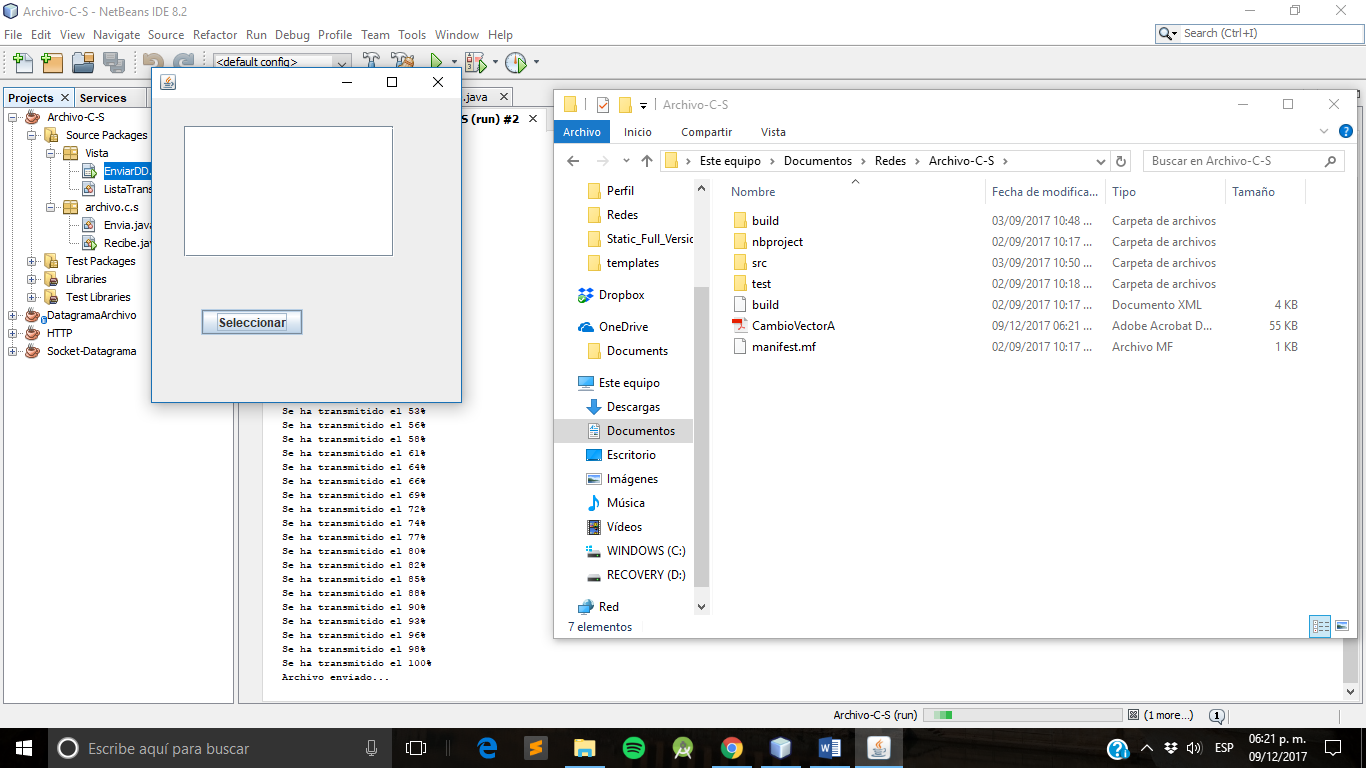
return true;

}

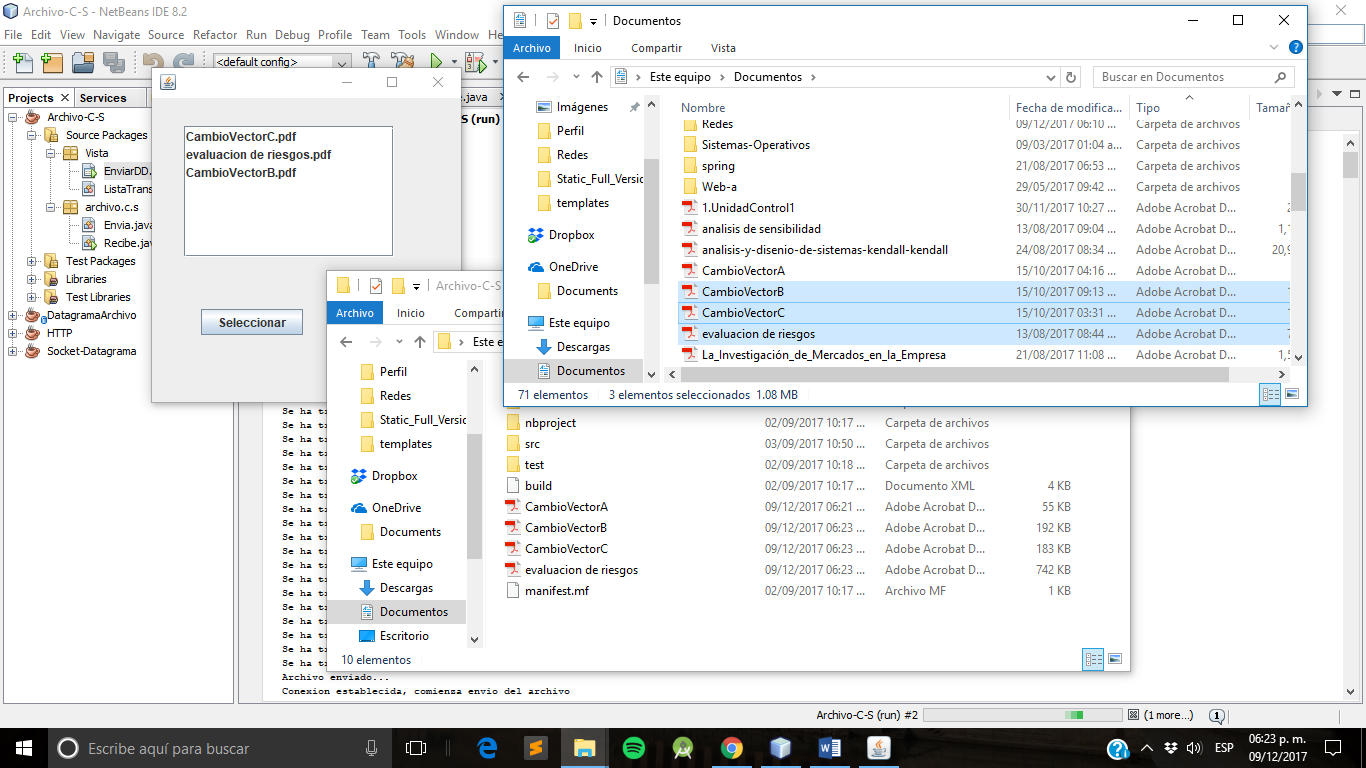
}

**Pruebas**

Para las pruebas primero se ejecuta el servidor y asi esperar los archivos que serán recibidos. En primera instancia se muestra cómo se envía un archivo sin drag & drop.



Ahora se mostrar como se envían archivos por medio de drag & drop, este proceso se realiza gracias a la clase ListTransfer, java la cual nos ayuda a ejecutar el drag & drop en una lista en java.



**Conclusión**

El desarrollo de esta práctica, al ser la primera, nos muestra la importancia de los sockets de flujo para hacer el envió de archivos, en este caso la utilizacion de sockets de flujo nos facilitan el envió de archivos debido a que nos manda los datos ordenados, al contrario de un socket de datagrama con el cual podemos correr el riesgo de que estos lleguen en desorden o que tengamos alguna perdida de datos al momento de haber creado el datagrama.

La utilización de los sockets nos muestra el gran panorama que tienen las redes, ya que nos muestra con exactitud un ejemplo de cómo las grandes páginas de internet hacen el envió de cada uno de los archivos que nosotros hacemos de una forma fácil gracias a ellos.