TECNOLOGICO NACIONAL DE MEXICO
Instituto Tecnológico de la Laguna
Ingenieria en Sistemas Computacionales

TOPICOS AVANZADOS DE PROGRAMACION

PERIODO: Ene - Jun / 2020 GRUPO: "B" 17 – 18 Hrs

PRACTICA No. U4P01

Copia de archivos por lote utilizando Threads

ALUMNOS:

18131209	José Misael Adame Sandoval
18131227	Ricardo Raúl Castro Luna
18131238	Jorge Arturo Galindo Uribe

PROFESOR:

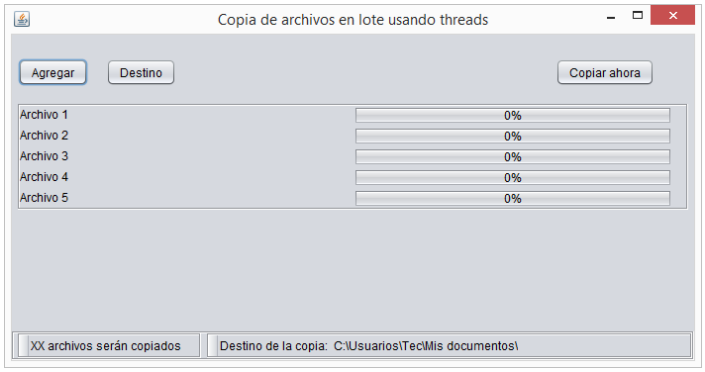
Ing. Luis Fernando Gil Vázquez

Torreón, Coah. a 8 de junio de 2020

Situación didáctica

Z-Soft es una compañía de desarrollo de utilidades de software que permiten automatizar tareas comunes para muchos de los usuarios de computadoras. El área de mercadotécnia de la firma ha detectado un área de oportunidad para desarrollar una nueva herramienta que permita hacer la copia de archivos en grupo (por lote) a una carpeta de destino. Es una necesidad común en el mundo de la computadora doméstica cuando se requiere copiar distintos archivos que se encuentran en diferentes carpetas hacia una sola de destino.

Se ha construido una vista preliminar de la IU para hacer una prueba del concepto. La IU tendrá una vista aproximada a la siguiente figura:



Se tendrá un botón para que el usuario pueda agregar nuevos archivos al lote. La selección del archivo se hará a través de un dialogo estándar ABRIR. Otro botón permitirá especificar la carpeta de destino. Solo se permitirá un máximo de 10 archivos en el lote. Un botón COPIAR AHORA permitirá ejecutar el copiado de los archivos de manera simultánea, todos a la vez.

Para eficientar la tarea de copiado cada archivo será transferido a la carpeta de destino mediante la ejecución en un hilo. Es decir si se van a copiar 5 archivos entonces se crearán 5 hilos. La implementación del hilo será con la clase Thread o el interface Runnable según le sea asignado a cada equipo. Será deseable contar con un botón que permita eliminar todo el contenido de la lista de archivos a copiar.

Una restricción en la implementación de esta aplicación es que cada archivo deberá ser copiado a su destino que tendrá el mismo nombre y se deberá copiar byte por byte, es decir se lee un byte del archivo fuente y se escribe ese byte en el archivo de destino.

En la sección de **ANALISIS** describir la estrategia para la tarea que va a realizar el hilo, es decir qué deberá contener el método run () del hilo para realizar la copia del archivo, puede exponerse en términos de un algoritmo en pseudocódigo y no en instrucciones Java.

En la sección de **DISEÑO** va el diagrama de clases UML de la aplicación.

En la sección de **CODIGO** presentar el código del script de creación de la base de datos, enseguida el código de todas las clases JAVA con la calidad establecida.

En la sección de **PRUEBA DE EJECUCION** presentar un mini manual de usuario incluyendo capturas de pantalla de la aplicación.

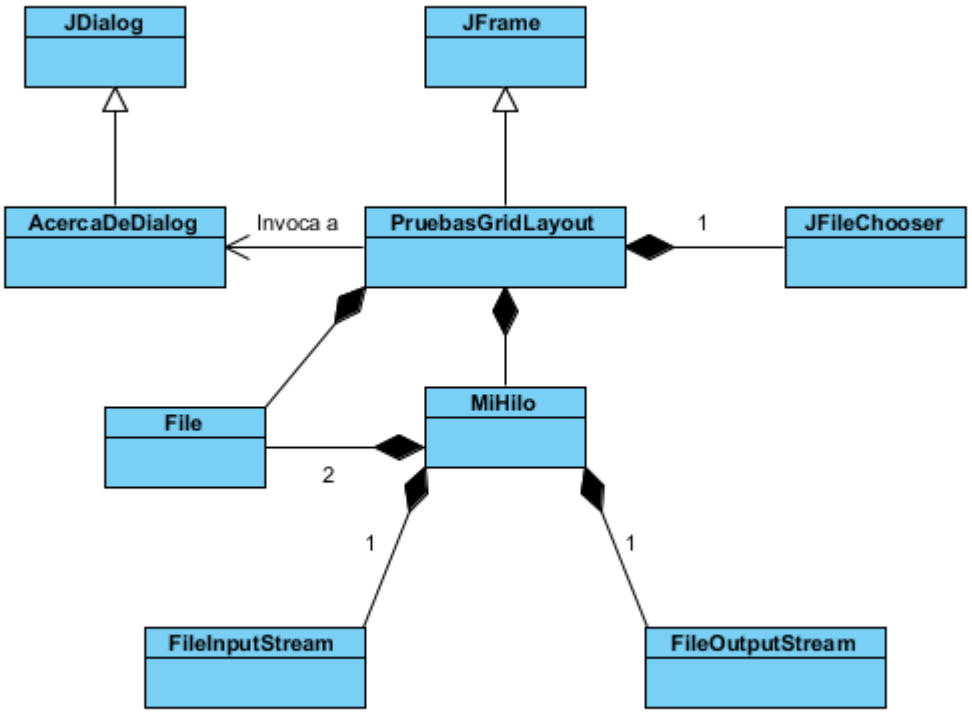
Análisis

En el método run del hilo (La clase llamada MiHilo), la primera instrucción que hace es crear un archivo que será el origen, tomando la ruta del mismo para el constructor. Luego, comprueba que no haya ningún archivo con el nombre de la copia en el destino. En el caso de que sí lo haya, no se realizará la copia, pero en caso contrario, se sigue con el código.

Luego de la comprobación, se crea el archivo copia con la ruta de destino, los objetos FileOutputStream y FileInputStream (fout y fis respectivamente), que utilizaremos para realizar la copia, contadores que nos servirán para el método, el arreglo de bytes buf, y el long largo, que será igual al tamaño del origen.

Después, dentro de una sentencia try-catch, inicializamos tanto el fis con el archivo origen como el fout con la copia. Seguido, se inicia un ciclo while, que se realizará mientras que r sea diferente a -1, siendo r el valor que devuelve fin, es decir, mientra lea al archivo original. Durante este ciclo se copiará el archivo con ayuda del fout, quien recibe como parámetro el byte (buf), 0 y r. A su vez, se mostrará el incremento en la JProgressBar del frame. Una vez terminado el ciclo, se asignará como 100 el valor del ProgressBar y se cerrarán tanto fin como fout.

Diseño


JDialog
JFrame
AcercaDeDialog
Invoca a
PruebasGridLayout
1
JFileChooser
File
MiHilo
2
1
1
FileInputStream
FileOutputStream

Código

PruebasGridLayout.java

```
/*-------------------------------------------------------------------------------------------
:*                       INSTITUTO TECNOLOGICO DE LA LAGUNA
:*                     INGENIERIA EN SISTEMAS COMPUTACIONALES
:*                       TOPICOS AVANZADOS DE PROGRAMACION "B"
:*
:*                    SEMESTRE: ENE-JUN/2020    HORA: 17-18 HRS
:*
:*                        Frame principal de copia de archivos
:*
:*  Archivo     : PruebasGridLayout.java
:*  Autor       :  Jorge Arturo Galindo Uribe        18131238
:*                    José Misael Adame Sandoval     18131209
                     Ricardo Raúl Castro Luna        18131227
:*
:*  Fecha       : 8 de junio de 2020
:*  Compilador  : JAVA J2SE v1.8.0
:*  Descripci�n : Frame que permite al usuario cargar hasta
                      10 archivos distintos a copiar, elegir la carpeta
                      en donde serán copiados y limpiar los archivos ya copiados.
                      A su vez, permite ver el progreso del proceso y los datos de
                      los autores.
:*  Ultima modif:
:*  Fecha       Modific�             Motivo
:*=========================================================================================
:*  08/junio/2020      Arturo Galindo       Agregar Prólogo
:*-----------------------------------------------------------------------------------------*/package app;


import hilo.thread.MiHilo;
import java.awt.GridLayout;
import java.io.File;
import java.util.ArrayList;
import javax.swing.JFileChooser;
import javax.swing.JLabel;
import javax.swing.JProgressBar;

/*-------------------------------------------------------------------------*/

public class PruebasGridLayout extends javax.swing.JFrame {

    private static int cuenta = 0;
    ArrayList<MiHilo> Lote = new ArrayList();
    public String destino = "";

    /*-------------------------------------------------------------------------*/

    public PruebasGridLayout() {
        initComponents();
        jplArchivos.setLayout ( new GridLayout ( 0, 2 ) );
    }

    /*-------------------------------------------------------------------------*/

    /*-------------------------------------------------------------------------*/

    private void jbtnAgregarActionPerformed(java.awt.event.ActionEvent evt) {
        if(cuenta < 10){
            JFileChooser jfc = new JFileChooser();

            if(jfc.showOpenDialog(this) == JFileChooser.APPROVE_OPTION){
                File archivo = jfc.getSelectedFile();

                jplArchivos.add ( new JLabel ( "Archivo " + ++cuenta) );
                jlblCuenta.setText(cuenta +" Archivos seran copiados");
                JProgressBar progressBar = new JProgressBar ();
                progressBar.setStringPainted ( true );

                jplArchivos.add ( progressBar  );
                jplArchivos.setVisible ( false );
                jplArchivos.setVisible ( true  );
```

```
                Lote.add( new MiHilo ( archivo, destino, progressBar) );
            }
        }
    }

    /*--------------------------------------------------------------------------*/

    private void jbtnDestinoActionPerformed(java.awt.event.ActionEvent evt) {
        JFileChooser jfc = new JFileChooser ();
        jfc.setFileSelectionMode ( JFileChooser.DIRECTORIES_ONLY );
        if (  jfc.showOpenDialog ( this ) == JFileChooser.APPROVE_OPTION ) {
            jlblDestino.setText ("Destino de la copia: " + jfc.getSelectedFile().getAbsolutePath() );
            destino = jfc.getSelectedFile().getPath();
        }
    }

    /*--------------------------------------------------------------------------*/

    private void jbtnCopiarActionPerformed(java.awt.event.ActionEvent evt) {
        for (int i = 0; i < cuenta; i++) {
            Lote.get(i).setDestino(destino);
            Lote.get(i).start();
        }
    }

    /*--------------------------------------------------------------------------*/

    private void jbtnLimpiarActionPerformed(java.awt.event.ActionEvent evt) {
        for (int i = 0; i < cuenta; i++) {
            Lote.remove(i);
            jplArchivos.removeAll();
        }
        jlblCuenta.setText(0 +" Archivos seran copiados");
        cuenta = 0;
        jplArchivos.validate();
        jplArchivos.repaint();
    }

    private void jbtnAcercaDeActionPerformed(java.awt.event.ActionEvent evt) {
        AcercaDeDialog acd=new AcercaDeDialog(this, true);
        acd.setVisible(true);
    }

    /*--------------------------------------------------------------------------*/

    synchronized public void setProgreso(double progreso, JProgressBar progressBar){
        mostrarProgreso(progreso, progressBar);
    }

    /*--------------------------------------------------------------------------*/

    private void mostrarProgreso(double progreso, JProgressBar jpbrProgreso){
        jpbrProgreso.setValue((int) progreso);
    }

    /*--------------------------------------------------------------------------*/

    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(PruebasGridLayout.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(PruebasGridLayout.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        } catch (IllegalAccessException ex) {
```

```
java.util.logging.Logger.getLogger(PruebasGridLayout.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(PruebasGridLayout.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        }
        //</editor-fold>
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new PruebasGridLayout().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JToolBar jToolBar1;
    private javax.swing.JToolBar jToolBar2;
    private javax.swing.JButton jbtnAcercaDe;
    private javax.swing.JButton jbtnAgregar;
    private javax.swing.JButton jbtnCopiar;
    private javax.swing.JButton jbtnDestino;
    private javax.swing.JButton jbtnLimpiar;
    private javax.swing.JLabel jlblCuenta;
    private javax.swing.JLabel jlblDestino;
    private javax.swing.JPanel jplArchivos;
    // End of variables declaration
}
```

MiHilo.Java

```
/*-------------------------------------------------------------------------------------------
:*                       INSTITUTO TECNOLOGICO DE LA LAGUNA
:*                     INGENIERIA EN SISTEMAS COMPUTACIONALES
:*                       TOPICOS AVANZADOS DE PROGRAMACION "B"
:*
:*                   SEMESTRE: ENE-JUN/2020    HORA: 17-18 HRS
:*
:*                       Clase que realiza la copia de archivos
:*
:*  Archivo     : MiHilo.java
:*  Autor       :  Jorge Arturo Galindo Uribe        18131238
:*                 José Misael Adame Sandoval     18131209
:*                 Ricardo Raúl Castro Luna        18131227
:*
:*  Fecha       : 8 de juniode 2020
:*  Compilador  : JAVA J2SE v1.8.0
:*  Descripci?n : Clase que realiza la copia de los archivos usando
                      clases como File, FileOutPutStream, FileInputStream,
                      y utilizando hilos para realizar la operación con el método
                      run, así como getters y setter para el destino y el archivo a copiar.
:*  Ultima modif:
:*  Fecha       Modific?            Motivo
:*===========================================================================================
:*  08/junio/2020      Arturo Galindo       Agregar Prólogo
:*-------------------------------------------------------------------------------------------*/
package hilo.thread;

import app.PruebasGridLayout;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import javax.swing.JOptionPane;
import javax.swing.JProgressBar;

/*-----------------------------------------------------------------------------*/

public class MiHilo extends Thread {
```

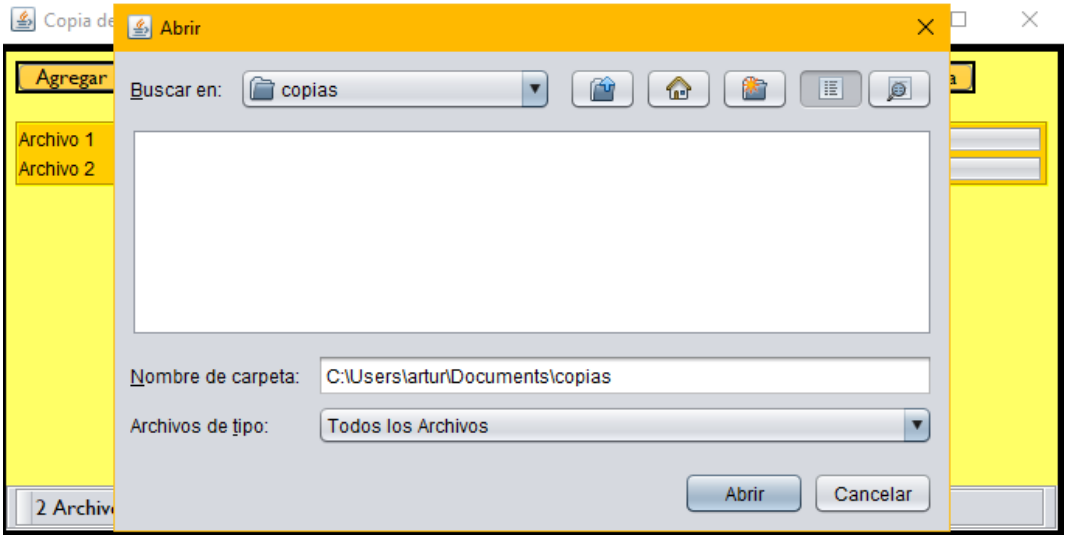
```
/*---------------------------------------------------------------------------*/

private File archivo;
private PruebasGridLayout frame = new PruebasGridLayout();
private String destino = "";
private JProgressBar progressBar = new JProgressBar();

/*---------------------------------------------------------------------------*/

public MiHilo(){
    super();
}

/*---------------------------------------------------------------------------*/

public MiHilo( File archivo, String destino, JProgressBar progressBar){
    super( archivo.getName() );
    this.archivo = archivo;
    this.destino = destino;
    this.progressBar = progressBar;
}

/*---------------------------------------------------------------------------*/

@Override
public void run(){
    File origen = new File(archivo.getAbsolutePath());
    if(!(Files.exists(Paths.get(destino + File.separator +  archivo.getName())))){
        File copia = new File(destino + File.separator +  archivo.getName());
        FileInputStream fin = null;
        FileOutputStream fout = null;

        byte[] buf = new byte[1024];
        long largo = origen.length();

        int r = 0;
        long count = 0;

        try {

            fin = new FileInputStream( origen );
            fout = new FileOutputStream( copia );

            while ((r = fin.read(buf)) != -1) {
                count += r;
                frame.setProgreso( ( (100 * count) / largo ), progressBar );
                fout.write(buf, 0, r);


                Thread.sleep( 10 );
            }

            frame.setProgreso( 100 , progressBar );
            fin.close();
            fout.close();

        } catch (IOException ex) {
            JOptionPane.showMessageDialog( null, ex );
        } catch (InterruptedException ex) {
            JOptionPane.showMessageDialog( null, ex );
        }
    }
}

/*---------------------------------------------------------------------------*/

public File getArchivo() {
    return archivo;
}

/*---------------------------------------------------------------------------*/

public void setArchivo(File archivo) {
    this.archivo = archivo;
}

/*---------------------------------------------------------------------------*/

public String getDestino() {
    return destino;
```

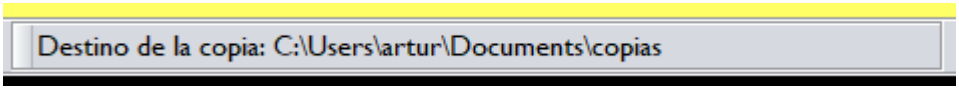
```
    }

    /*--------------------------------------------------------------------------*/

    public void setDestino(String destino) {
        this.destino = destino;
    }

    /*--------------------------------------------------------------------------*/
}
```

Pruebas de ejecución

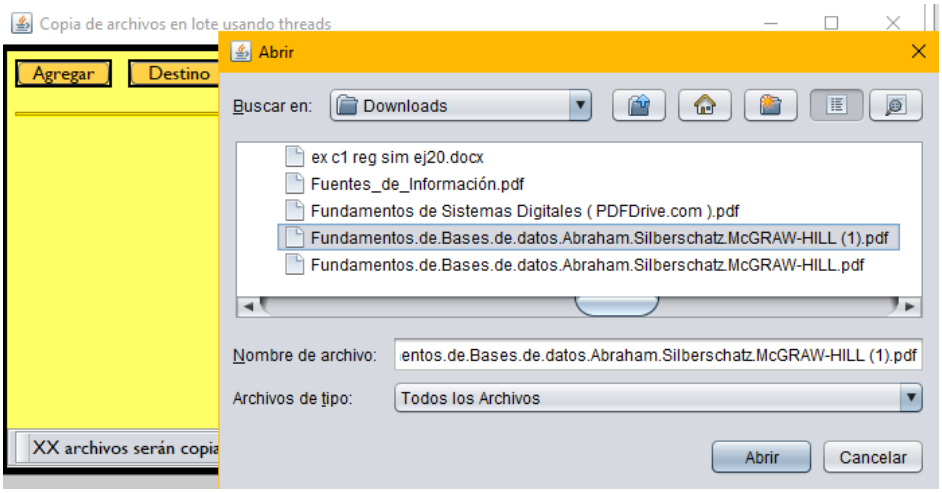
Al iniciar la aplicación, seremos presentados con la vista inicial:

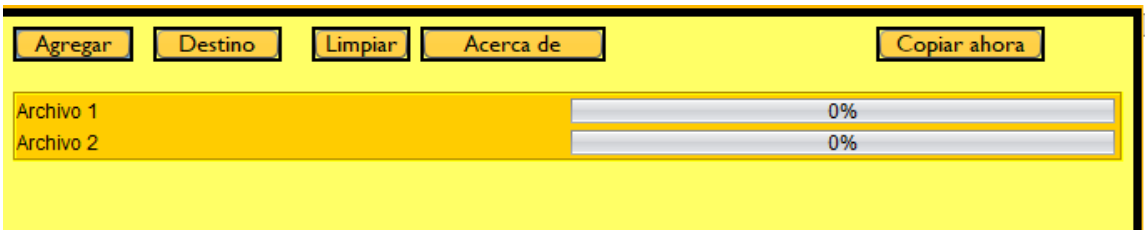
Para utilizarla, debemos ingresar la dirección de destino con el botón Destino. En la parte inferior podemos ver la ruta:





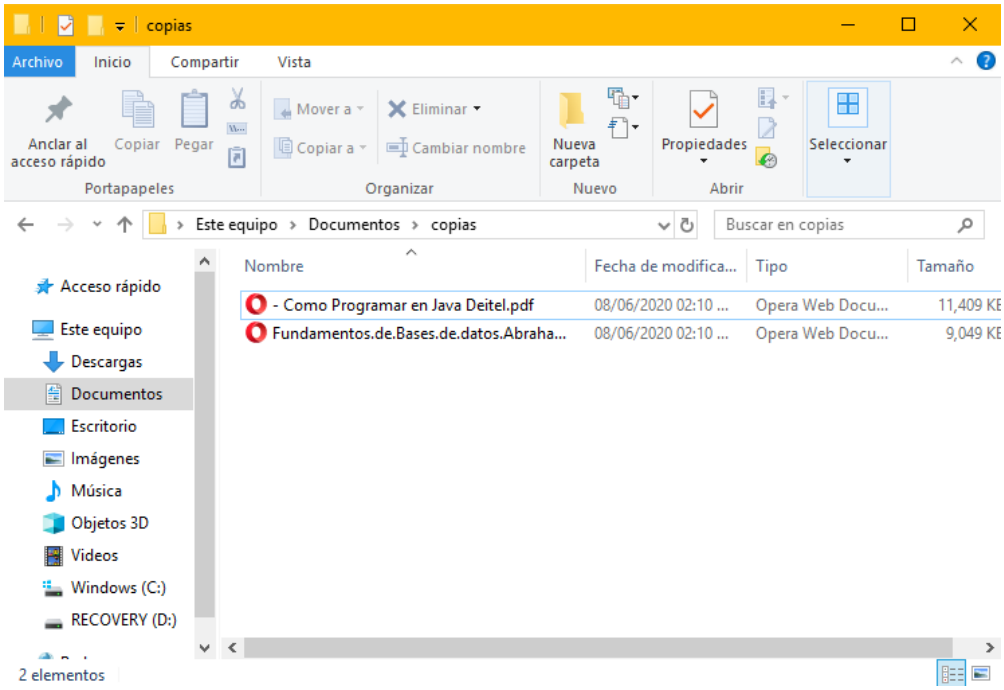
Una vez seleccionada la ruta de destino, se procederá a escoger los archivos a copia, usando el botón agregar, donde serán agregados a la lista:





Después, se debe seleccionar el botón copiar ahora y esperar a que se realice la copia, indicado con la barra de progreso:

Ahora podemos ver la carpeta y ver las copias



El botón acerca de muestra los datos del programa y los autores