Visor NetCDF

Felipe Triviño Paredes

Leonardo Bello Restrepo

Nicolás Rodriguez Gutiérrez

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

INFORMATICA 1

BOGOTA D.C.

2020

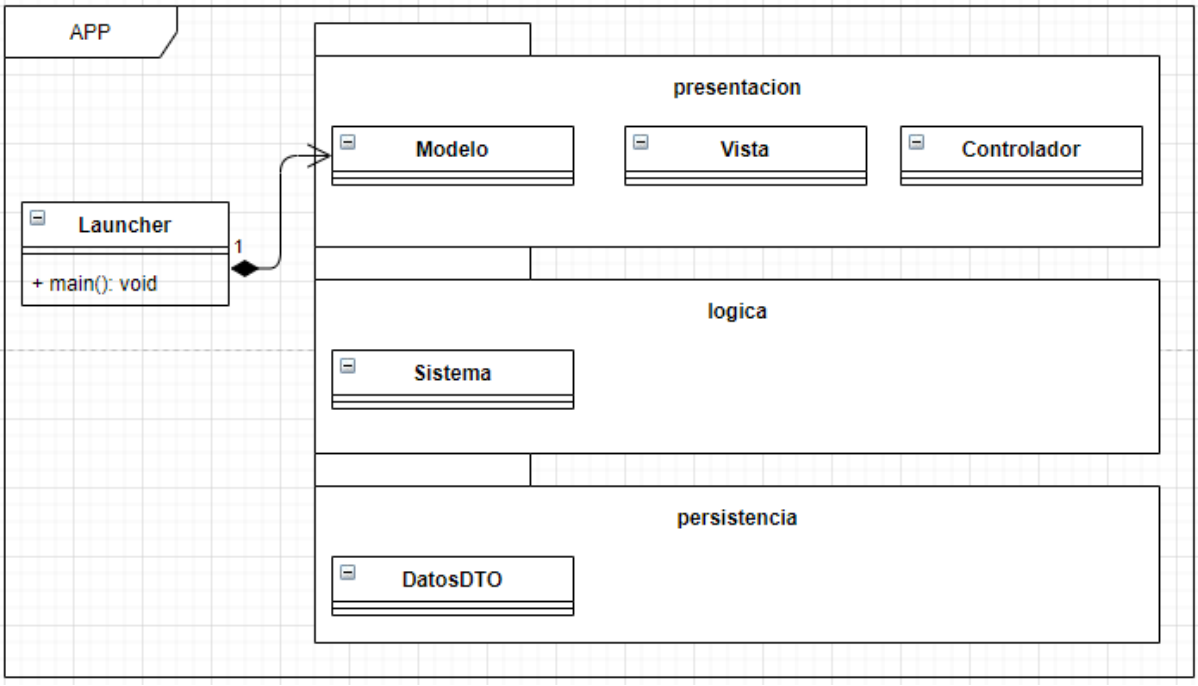
Taller 2 Visor NetCDF

1. Propósito

Crear un visor de archivos en formato NetCDF, un formato de archivos destinado a almacenar datos científicos multivariables, en formato binario. En dicho visor se deben listar mediante una tabla todas las variables contenidas en un archivo de este formato, por cada variable se tienen que evidenciar sus atributos y valores.

2. Arquitectura

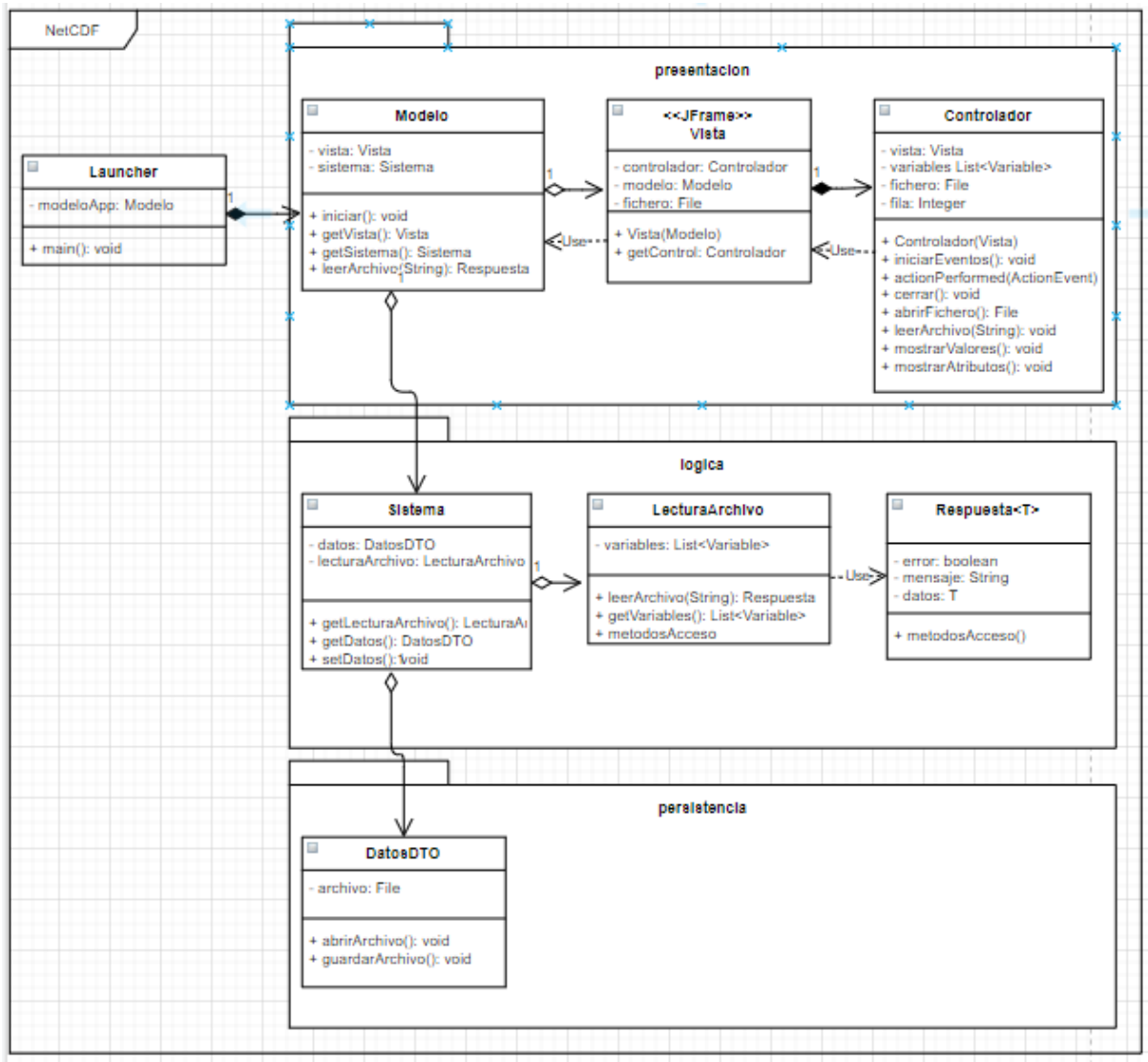
La arquitectura del proyecto usada es la vista en clase, una arquitectura de 3 capas (Presentación, Lógica, Persistencia), se debe garantizar respetar la jerarquía entre las diferentes capas, desligando la lógica de la vista de la aplicación. Esta arquitectura se puede apreciar en el siguiente diagrama:



3. Desarrollo

El proyecto fue realizado con el lenguaje de programación Java, haciendo uso de la librería netcdfAll-5.3.3, mediante esta se puede acceder al contenido de los archivos binarios, dando acceso a todas sus variables.

En el siguiente diagrama de clases de la aplicación se puede ver la arquitectura propuesta:



A continuación, se muestra la distribución de paquetes según la arquitectura descrita en el punto anterior

En el paquete por defecto se encuentra la clase con el método principal de la aplicacion, desde acá se instancia la clase Modelo

```
/**
 * Clase principal
 */
public class Launcher {

    private Modelo modeloApp;

    /**
     * Metodo incial ejecucion
     * @param args
     */
    public static void main(String[] args) {
       new Launcher();
    }

    /**
     * Metodo instanciar Modelo e inicializarlo
     */
    public Launcher() {
        modeloApp = new Modelo();
        modeloApp.iniciar();
    }
```

En el paquete de presentacion se encuentran contenidas las clases de “Modelo”, “Vista” y “Controlador”, para respetar la jerarquía de las capas de la arquitectura, desde Modelo se instancia a la vista y la vista al Controlador, pasando en el constructor una referencia de la clase para poder acceder los atributos y métodos de las clases anteriores:

```java
public class Modelo {

    private Vista vista;
    private Sistema sistema;

    /**
     * Metodo para obtener unica instancia de Vista usando Singleton
     * @return Vista
     */
    public Vista getVista() {
        if (vista == null) {
            vista = new Vista(this);
        }
        return vista;
    }

    /**
     * Metodo para obtener unica instancia de Sistema usando Singleton
```

```java
public class Vista extends javax.swing.JFrame {

    private Controlador controlador;
    private final Modelo modelo;
    File fichero = null;

    public Vista(Modelo modelo) {
        this.modelo = modelo;
        initComponents();
        jLabelFichero.setOpaque(true);
        //titulo e icono aplicacion
        ImageIcon img = new ImageIcon(this.getClass().getResource("/imagenes/logo.png"));
        this.setIconImage(img.getImage());
        this.setTitle("NetCDF / Informatica 1 / Taller 2");
    }
```

```java
/**
 * clase controlador vista principal
 */
public class Controlador implements ListSelectionListener {

    private final Vista vista;

    private List<Variable> variables;
    File fichero = null;
    Integer fila = null;

    public Controlador(Vista ventana) {
        vista = ventana;
        iniciarEventos();
    }
```

En el paquete de logica se encuentran las clases “Sistema” la cual centraliza toda la lógica de la aplicación instanciando en este caso otras clases de lógica como la de

“LecturaArchivo”, en esta clase se encuentra el método de lógica para abrir el archivo y retorna una lista de variables con sus respectivos datos mediante una clase “Respuesta” con una estructura para ser usada por un cliente, en este caso la vista

```java
/**
 * Clase principal logica, centraliza funciones para tratamiento de archivo
 */
public class Sistema {

    private DatosDTO datos;
    private final LecturaArchivo lecturaArchivo;

    public Sistema() {
        this.lecturaArchivo = new LecturaArchivo();
    }

     public LecturaArchivo getLecturaArchivo() {
        return lecturaArchivo;
```

```java
public class LecturaArchivo {

    private List<Variable> variables;

    /**
     * Metodo que lee el archivo mediante NetcdfFile.open
     * @param filename nombre del archivo a leer
     * @return Respuesta
     */
    public Respuesta leerArchivo(String filename) {
        NetcdfFile ncfile;
        Respuesta<List<Variable>> respuesta = new Respuesta();
        try {
            ncfile = NetcdfFile.open(filename);
            System.out.println("leyendo archivo...");
            variables = ncfile.getVariables();
            respuesta.setDatos(variables);
        } catch (IOException ex) {
            respuesta.setError(true);
            respuesta.setMensaje("error al leer el archivo");
        }
        return respuesta;
    }
```

```java
/**
 * Clase generica con estructura de respuesta con error, mensaje
 * y datos mediante el objeto de la calse generica
 * @param <T> Objeto generico
 */
public class Respuesta<T> {

    private boolean error;
    private String mensaje;
    private T datos;

    public Respuesta() {
    }

    public Respuesta(boolean error, String mensaje, T datos) {
        this.error = error;
        this.mensaje = mensaje;
        this.datos = datos;
    }
```

En el paquete persistencia se encuentra la clase DatosDTO la cual será usada más adelante cuando se implemente la funcionalidad de guardar datos ingresados en un archivo NetCDF

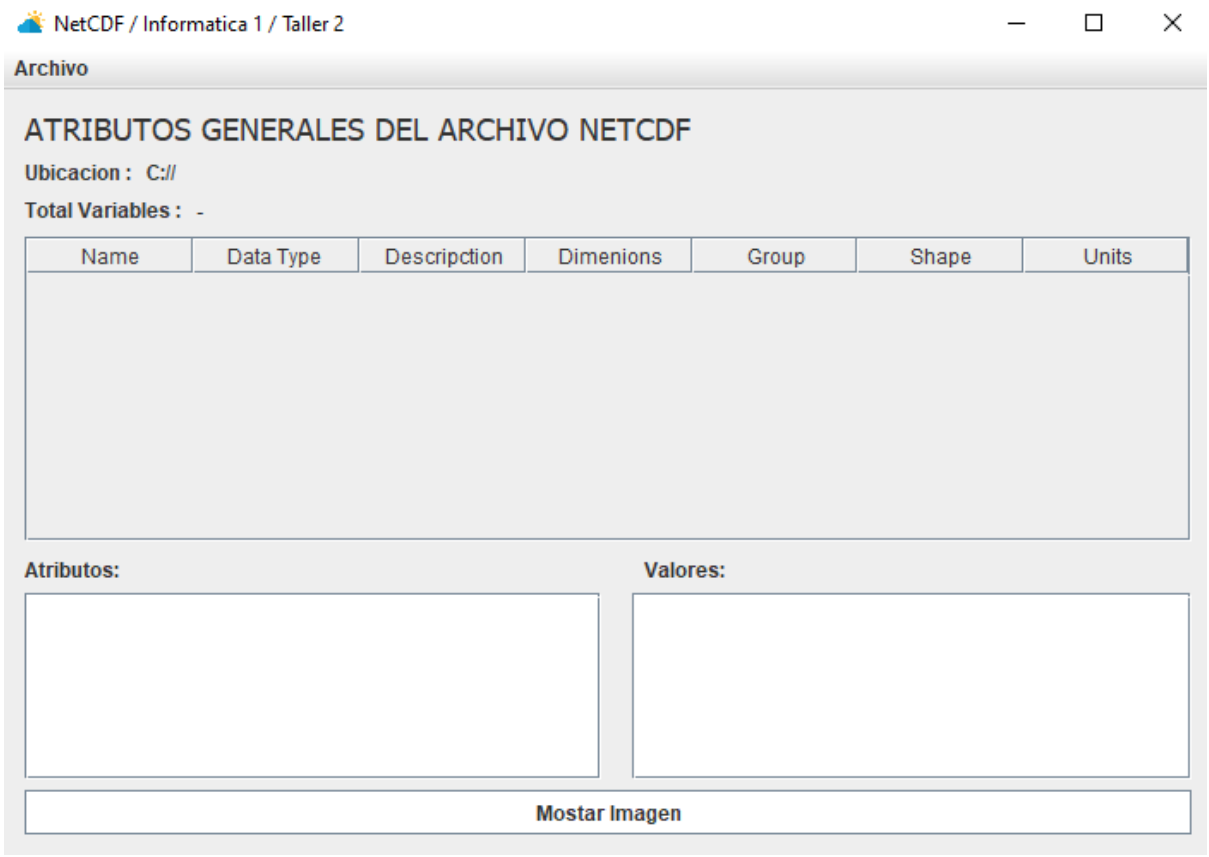
```java
/**
 * Clase para persistencia
 */
public abstract class DatosDTO {


    public abstract void guardarArchivo() throws Exception;

}
```

4. Funcionamiento

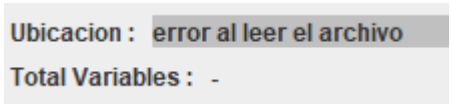
- Interfaz gráfica de la aplicación:



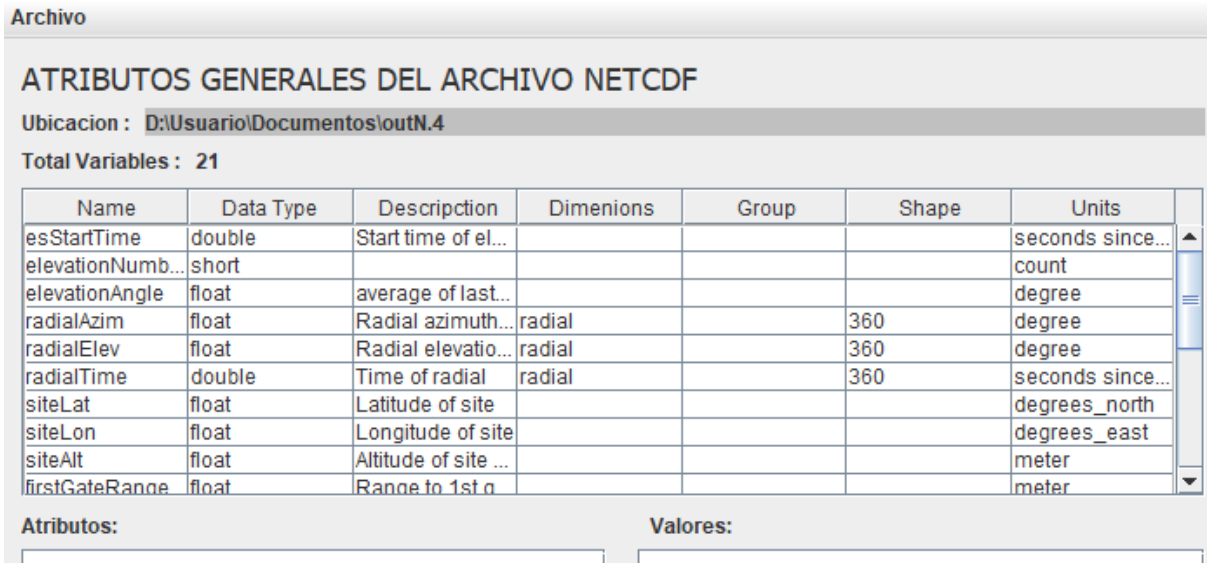
- Abrir archivo:

La aplicación permite seleccionar un fichero a ser leído mediante la opción del menú Archivo – Abrir,

cargando todas las variables y sus datos de ser un archivo en formato netCDF, en caso contrario informa error al leer el archivo



Cuando se selecciona un archivo netCDF muestra la ubicación del mismo, el número de variables encontradas y una tabla con la información de las variables



- Ver atributos y valores de una variable

Para visualizar los atributos y valores de una variable se debe seleccionar la variable en la tabla de variables.