

Visor NetCDF – servicio REST

Felipe Triviño Paredes
Leonardo Bello Restrepo
Nicolás Rodríguez Gutiérrez

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS
INFORMATICA 1
BOGOTA D.C.
2020

Taller 3 Visor NetCDF - servicio REST

1. Propósito

El objetivo es crear un servicio REST que devuelva los datos solicitados (backend) en formato NetCDF, un formato de archivos destinado a almacenar datos científicos multivariantes, en formato binario. En dicho servicio deben retornar una lista de variables contenidas en el archivo ingresado en formato NetCDF.

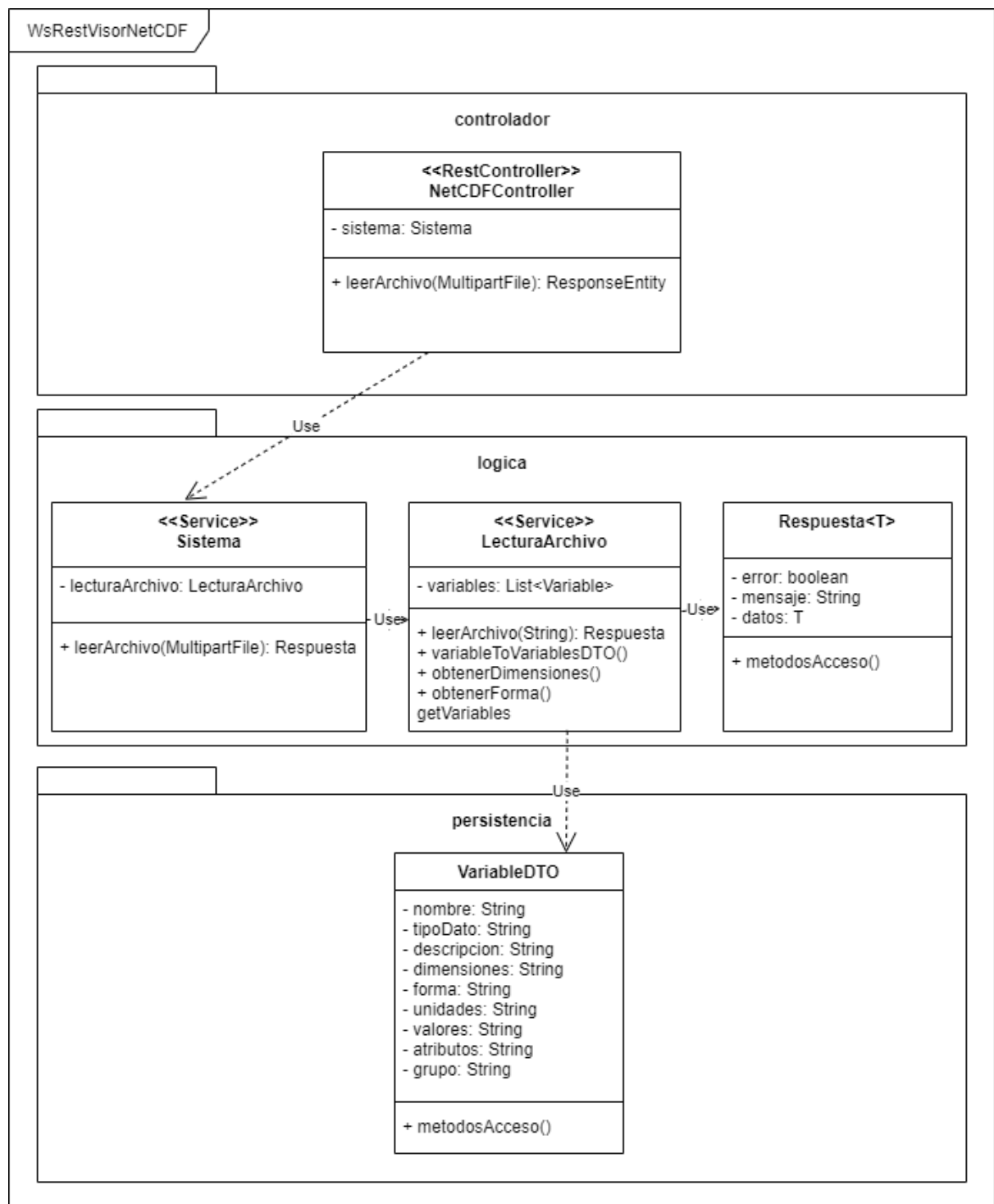
2. Arquitectura

La arquitectura del proyecto usada es la vista en clase, una arquitectura de 3 capas (Controlador, Lógica, Persistencia), se debe garantizar respetar la jerarquía entre las diferentes capas, desligando la lógica de la vista de la aplicación.

3. Desarrollo

El proyecto fue realizado con el lenguaje de programación Java, usando el framework Spring Boot v 2.3.4. además de la librería netcdfAll-4.3.22, 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

..	
controlador	inicio documentacion proyecto
logica	commit inicial
persistencia	commit inicial
SwaggerConfig.java	commit inicial
WsRestVisorNetCdfApplication.java	commit inicial

En el paquete por defecto se encuentra la clase con el método principal de la aplicación, desde acá se instancia el `SpringApplication.run`:

```
package com.visorNetCDF;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class WsRestVisorNetCdfApplication {

    public static void main(String[] args) {
        SpringApplication.run(WsRestVisorNetCdfApplication.class, args);
    }

}
```

En el paquete de Controlador y para respetar la jerarquía de las capas de la arquitectura, desde la Lógica 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:

```
package com.visorNetCDF.controlador;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.multipart.MultipartFile;

import com.visorNetCDF.logica.Respuesta;
import com.visorNetCDF.logica.Sistema;
import com.visorNetCDF.persistencia.VariableDTO;

/**
 * clase controlador con peticiones del servicio netCDF
 */
@RestController
@RequestMapping("/netCDF")
public class NetCDFController {
```




```

@Autowired
private Sistema sistema;

@PostMapping("/leerArchivo")
public ResponseEntity<Respuesta<List<VariableDTO>>>
leerArchivo(@RequestParam("file") MultipartFile file) {
    Respuesta<List<VariableDTO>> respuesta = new Respuesta<>();
    respuesta = sistema.leerArchivo(file);
    if (respuesta.isError()) {
        return new ResponseEntity<Respuesta<List<VariableDTO>>>(respuesta,
        HttpStatus.EXPECTATION_FAILED);
    }
    return new ResponseEntity<Respuesta<List<VariableDTO>>>(respuesta,
    HttpStatus.OK);
}
}

```

En el paquete de lógica 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

..	
 LecturaArchivo.java	commit inicial
 Respuesta.java	commit inicial
 Sistema.java	commit inicial

```

package com.visorNetCDF.logica;

import org.springframework.stereotype.Service;

import com.visorNetCDF.persistencia.VariableDTO;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import ucar.ma2.Array;
import ucar.nc2.Dimension;
import ucar.nc2.NetcdfFile;
import ucar.nc2.Variable;

/**
 * Clase lectura archivo netCDF mediante libreria externa
 */
@Service
public class LecturaArchivo {

    private List<VariableDTO> variables;

    /**

```

```

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

/**
 * Metodo para convertir variables libreria a variableDTO
 * @param variables
 * @return Lista VariableDTO
 */
public List<VariableDTO> variableToVariablesDTO(List<Variable>
variables) throws IOException{
    List<VariableDTO> variablesDTO = new ArrayList<>();
    for (Variable var : variables) {
        VariableDTO varDTO = new VariableDTO(var.getFullName(),
var.getDataType().toString(), var.getDescription(),
obtenerDimensiones(var.getDimensions()),
obtenerForma(var.getShape(), var.getUnitsString());
        Array data = var.read();
        varDTO.setValores(data.toString());
        varDTO.setAtributos(var.toString());
        varDTO.setGrupo(var.getGroup().getGroups().toString());
        variablesDTO.add(varDTO);
    }
    return variablesDTO;
}

/**
 * metodo para obtener dimensiones
 * @param d lista de dimensiones
 * @return String
 */
public String obtenerDimensiones(List<Dimension> d) {
    String cadena = "";
    if (!d.isEmpty()) {
        for (Dimension dim : d) {
            if (cadena.isEmpty()) {
                cadena += dim.getFullName();
            } else {
                cadena += "," + dim.getFullName();
            }
        }
    }
    return cadena;
}

/**




```

```

    * Metodo para obtener formas
    * @param f formas
    * @return String
    */
    public String obtenerForma(int[] f) {
        String cadena = "";
        if (f.length > 0) {
            for (int i : f) {
                if (cadena.isEmpty()) {
                    cadena += i;
                } else {
                    cadena += "," + i;
                }
            }
        }
        return cadena;
    }

    /**
     * retorna lista de variables del archivo leido
     * @return lista Variables
     */
    public List<VariableDTO> getVariables() {
        return variables;
    }
}

```

..	
 LecturaArchivo.java	commit inicial
 Respuesta.java	commit inicial
 Sistema.java	commit inicial

```

package com.visorNetCDF.logica;

/**
 * Clase generica con estructura de respuesta con error, mensaje
 * y datos mediante el objeto de la clase 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;
    }
}

```

```

    public boolean isError() {
        return error;
    }

    public void setError(boolean error) {
        this.error = error;
    }




    public String getMensaje() {
        return mensaje;
    }

    public void setMensaje(String mensaje) {
        this.mensaje = mensaje;
    }

    public T getDatos() {
        return datos;
    }

    public void setDatos(T datos) {
        this.datos = datos;
    }
}

```

..	
 LecturaArchivo.java	commit inicial
 Respuesta.java	commit inicial
 Sistema.java	commit inicial

```

package com.visorNetCDF.logica;

import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.web.multipart.MultipartFile;

import com.visorNetCDF.persistencia.VariableDTO;

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

    @Autowired
    private LecturaArchivo lecturaArchivo;

    public Respuesta<List<VariableDTO>> leerArchivo(MultipartFile file) {

```



```

Respuesta<List<VariableDTO>> respuesta = new Respuesta<>();
try {
    if (file.isEmpty()) {
        respuesta.setError(true);
        respuesta.setMensaje("error, archivo vacio");
        return respuesta;
    }
    byte[] bytes = file.getBytes();
    Path path = Paths.get(file.getOriginalFilename());
    Files.write(path, bytes);
    respuesta = lecturaArchivo.leerArchivo(file.getOriginalFilename());
} catch (Exception e) {
    respuesta.setError(true);
    respuesta.setMensaje(e.getMessage());
}
return respuesta;
}

}

```

En el paquete persistencia se encuentra la clase DatosDTO la cual se implementa para guardar datos ingresados en un archivo NetCDF

```

package com.visorNetCDF.persistencia;

/**
 * Clase para persistencia
 */
public class VariableDTO {

    private String nombre;
    private String tipoDato;
    private String descripcion;
    private String dimensiones;
    private String forma;
    private String unidades;
    private String valores;
    private String atributos;
    private String grupo;

    public VariableDTO() {
    }

    public VariableDTO(String nombre, String tipoDato, String descripcion,
        String dimensiones, String forma,
        String unidades) {
        this.nombre = nombre;
        this.tipoDato = tipoDato;
        this.descripcion = descripcion;
        this.dimensiones = dimensiones;
        this.forma = forma;
        this.unidades = unidades;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
}

```

```
public String getTipoDato() {  
    return tipoDato;  
}  
  
public void setTipoDato(String tipoDato) {  
    this.tipoDato = tipoDato;  
}  
  
public String getDescripcion() {  
    return descripcion;  
}  
  
public void setDescripcion(String descripcion) {  
    this.descripcion = descripcion;  
}  
  
public String getDimensiones() {  
    return dimensiones;  
}  
  
public void setDimensiones(String dimensiones) {  
    this.dimensiones = dimensiones;  
}  
  
public String getForma() {  
    return forma;  
}  
  
public void setForma(String forma) {  
    this.forma = forma;  
}  
  
public String getUnidades() {  
    return unidades;  
}  
  
public void setUnidades(String unidades) {  
    this.unidades = unidades;  
}  
  
public String getValores() {  
    return valores;  
}  
  
public void setValores(String valores) {  
    this.valores = valores;  
}  
  
public String getAtributos() {  
    return atributos;  
}  
  
public void setAtributos(String atributos) {  
    this.atributos = atributos;  
}  
  
public String getGrupo() {  
    return grupo;  
}
```

```
public void setGrupo(String grupo) {  
    this.grupo = grupo;  
}  
  
}
```