

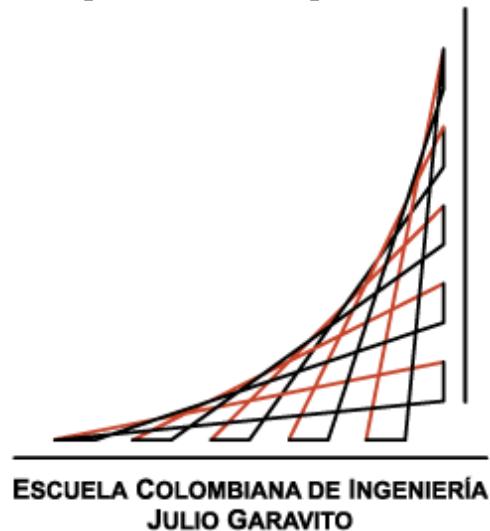
# Laboratorio #5

Taller de de modularización con virtualización e Introducción a  
AWS

**Juan David Navarro Jimenez**

**Luis Daniel Benavides Navarro**

Arquitectura Empresarial



# 1. Introducción

En este laboratorio se busca conocer y utilizar algunos de los servicios prestados en AWS, en este laboratorio vamos a utilizar el servicio EC2 de AWS en el cual vamos a crear una instancia de una maquina virtual y en ella vamos a probar un cliente concurrente el cual hace solicitudes a nuestro servidor desplegado en heroku. En este laboratorio se utiliza el servidor realizado en laboratorios pasados el cual permite la resolución de archivos con extensiones .jpg, .js y .html

# 2. Contenido

En el laboratorio usamos Meta protocolos de objetos, Patrón IoC y Reflexión en el desarrollo del servidor el cual funciona concurrentemente y se realizo una mejora en el cliente dado en el laboratorio para que pudiera ser concurrente y pudiera ejecutar la cantidad de peticiones que el cliente desee.

Para realizar este tipo de arquitectura el proyecto realizamos el siguiente diseño.

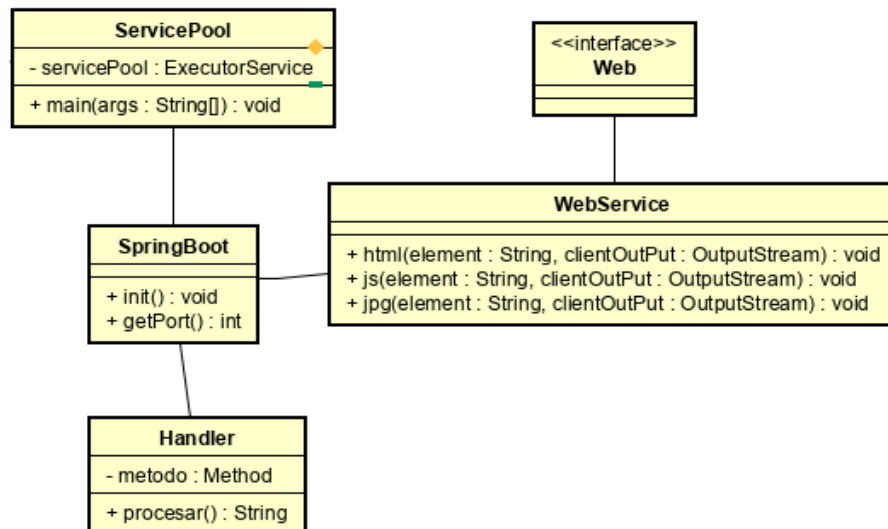


Figura 1: Diagrama de clases.

- **ServicePool** Es la clase principal donde tenemos un pool de ejecuciones.
- **SpringBoot** Es la clase donde se ejecuta como tal el servidor ya que en esta clase establecemos la conexión con el cliente por medio de sockets y según la inversión de dependencias ejecutamos la solicitud correspondiente y es almacenada en el headler para futuras consultas.
- **Handler** Es la clase que se encarga de revisar si ya esta almacenada una solicitud.
- **Web** Es una interfaz la cual permite el funcionamiento de la etiqueta @Web
- **WebService** Es la clase que tiene los métodos con la etiqueta @web

## 2.1. Demostración

Para mejorar el cliente realizamos un cliente que además de poder ingresar la URL de la cual queremos obtener los datos tengamos la posibilidad de ingresar la cantidad de solicitudes que queremos realizar y esta cantidad de solicitudes se ejecutaran concurrentemente el código del cliente desarrollado es el siguiente.

```
public class ClientAws extends Thread {

    //Atributos
    private static URL url;
    private static int numThread;
    private static ArrayList<Thread> poolThread;

    public ClientAws(URL url) {
        this.url = url;
    }

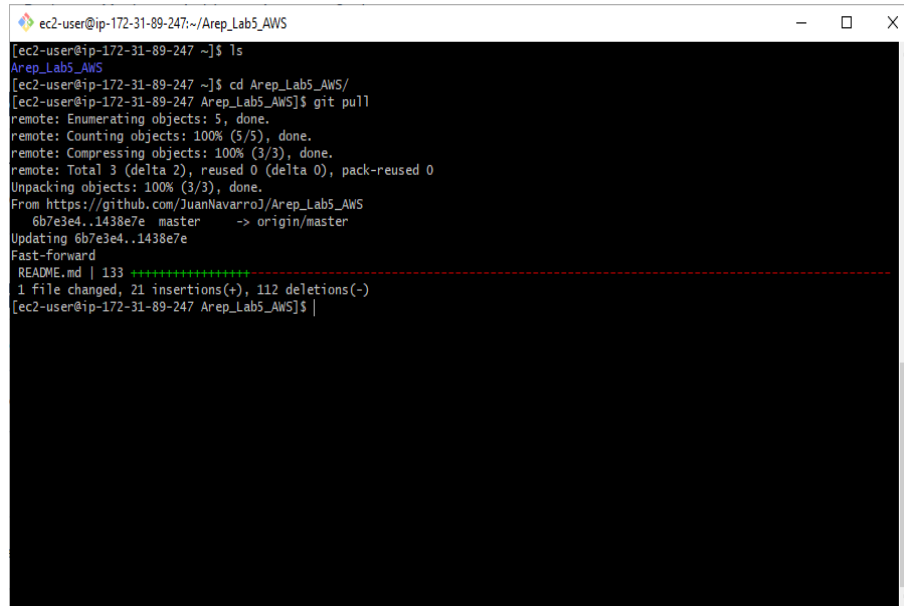
    public static void main(String[] args) throws Exception {
        url = new URL(args[0]);
        numThread = Integer.parseInt(args[1]);
        poolThread = new ArrayList<Thread>();
        for (int i = 0; i < numThread; i++) {
            poolThread.add(new ClientAws(url));
        }
        int cant = 0;
        for (Thread th : poolThread) {
            th.start();
            cant++;
        }
        System.out.println("Se ejecutaron " + cant + " solicitudes concurrentes.");
    }
}
```

Figura 2: Clase Cliente y su Main.

```
@Override
public void run() {
    try {
        BufferedReader reader = new BufferedReader(
            new InputStreamReader(url.openStream()));
        String inputLine = null;
        while ((inputLine = reader.readLine()) != null) {
            System.out.println(inputLine);
        }
    } catch (IOException x) {
        System.err.println(x);
    }
}
```

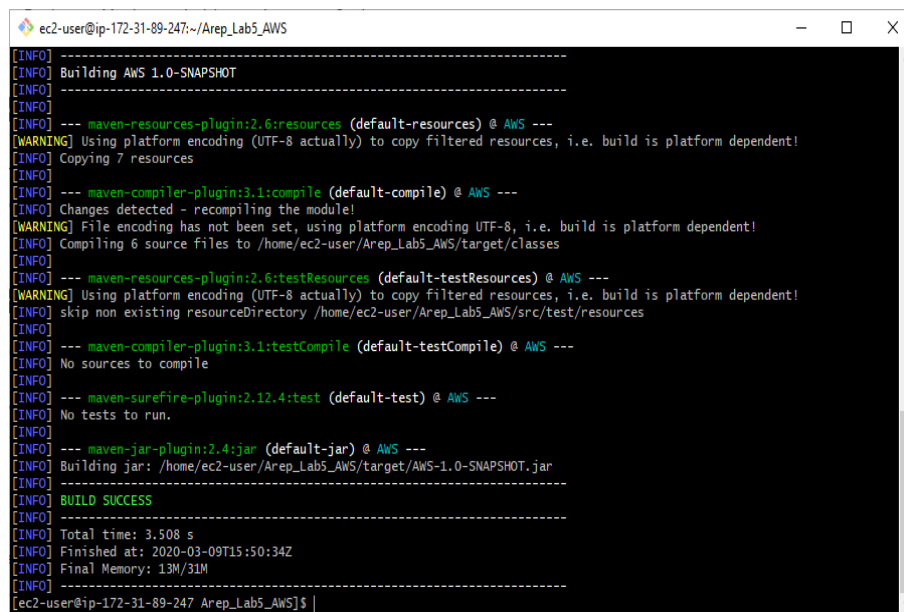
Figura 3: Metodo Run.

Para verificar el funcionamiento del cliente abrimos el proyecto en la maquina virtual de AWS y ejecutamos el cliente por medio del comando `mvn exec:java -Dexec.mainClass=com.example.MainDexec.args=.argument1`



```
ec2-user@ip-172-31-89-247:~/Arep_Lab5_AWS
[ec2-user@ip-172-31-89-247 ~]$ ls
Arep_Lab5_AWS
[ec2-user@ip-172-31-89-247 ~]$ cd Arep_Lab5_AWS/
[ec2-user@ip-172-31-89-247 Arep_Lab5_AWS]$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/JuanNavarro3/Arep_Lab5_AWS
   6b7e3e4..1438e7e  master    -> origin/master
Updating 6b7e3e4..1438e7e
Fast-forward
 README.md | 133 ++++++
 1 file changed, 21 insertions(+), 112 deletions(-)
[ec2-user@ip-172-31-89-247 Arep_Lab5_AWS]$
```

Figura 4: Actualizamos el proyecto por medio del comando git pull



```
ec2-user@ip-172-31-89-247:~/Arep_Lab5_AWS
[INFO] -----
[INFO] Building AWS 1.0-SNAPSHOT
[INFO] -----
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ AWS ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 7 resources
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ AWS ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 6 source files to /home/ec2-user/Arep_Lab5_AWS/target/classes
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ AWS ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/ec2-user/Arep_Lab5_AWS/src/test/resources
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ AWS ---
[INFO] No sources to compile
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ AWS ---
[INFO] No tests to run.
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ AWS ---
[INFO] Building jar: /home/ec2-user/Arep_Lab5_AWS/target/AWS-1.0-SNAPSHOT.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 3.508 s
[INFO] Finished at: 2020-03-09T15:50:34Z
[INFO] Final Memory: 13M/31M
[INFO] -----
[ec2-user@ip-172-31-89-247 Arep_Lab5_AWS]$
```

Figura 5: Compilamos el proyecto por medio de mvn package

```
ec2-user@ip-172-31-89-247:~/Arep_Lab5_AWS
[INFO] -----
[INFO] Building AWS 1.0-SNAPSHOT
[INFO] -----
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ AWS ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 7 resources
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ AWS ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 6 source files to /home/ec2-user/Arep_Lab5_AWS/target/classes
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ AWS ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/ec2-user/Arep_Lab5_AWS/src/test/resources
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ AWS ---
[INFO] No sources to compile
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ AWS ---
[INFO] No tests to run.
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ AWS ---
[INFO] Building jar: /home/ec2-user/Arep_Lab5_AWS/target/AWS-1.0-SNAPSHOT.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 3.508 s
[INFO] Finished at: 2020-03-09T15:50:34Z
[INFO] Final Memory: 13M/31M
[INFO] -----
[ec2-user@ip-172-31-89-247 Arep_Lab5_AWS]$ mvn exec:java -Dexec.mainClass="edu.eci.arep.Client.ClientAws" -Dexec.args="https://arep-lab5-aws.herokuapp.com/WebService/Web.html 10"
```

Figura 6: Ejecutamos el cliente por medio del comando mvn exec:java y pasamos como parámetro la URL de nuestro servidor desplegado en heroku y el numero de solicitudes concurrentes que deseamos.

```
ec2-user@ip-172-31-89-247:~/Arep_Lab5_AWS
[INFO] -----
[INFO] --- exec-maven-plugin:1.6.0:java (default-cli) @ AWS ---
Se ejecutaron 10 solicitudes concurrentes.
<!DOCTYPE html> <html lang="es"> <head> <meta charset="iso-8559-1"/> <meta name="description" content="Pagina de prueba Html
5"/> <meta name="keywords" content="html5 , css3 , javaScrip, diseño web"/> <title>Aws</title> </head> <body><
header id="cabeceraweb"> <h1> Laboratorio AWS Juan David Navarro </h1> </header></body></html>
<!DOCTYPE html> <html lang="es"> <head> <meta charset="iso-8559-1"/> <meta name="description" content="Pagina de prueba Html
5"/> <meta name="keywords" content="html5 , css3 , javaScrip, diseño web"/> <title>Aws</title> </head> <body><
header id="cabeceraweb"> <h1> Laboratorio AWS Juan David Navarro </h1> </header></body></html>
java.io.IOException: Server returned HTTP response code: 503 for URL: https://arep-lab5-aws.herokuapp.com/WebService/Web.html
java.io.IOException: Server returned HTTP response code: 503 for URL: https://arep-lab5-aws.herokuapp.com/WebService/Web.html
<!DOCTYPE html> <html lang="es"> <head> <meta charset="iso-8559-1"/> <meta name="description" content="Pagina de prueba Html
5"/> <meta name="keywords" content="html5 , css3 , javaScrip, diseño web"/> <title>Aws</title> </head> <body><
header id="cabeceraweb"> <h1> Laboratorio AWS Juan David Navarro </h1> </header></body></html>
java.io.IOException: Server returned HTTP response code: 503 for URL: https://arep-lab5-aws.herokuapp.com/WebService/Web.html
<!DOCTYPE html> <html lang="es"> <head> <meta charset="iso-8559-1"/> <meta name="description" content="Pagina de prueba Html
5"/> <meta name="keywords" content="html5 , css3 , javaScrip, diseño web"/> <title>Aws</title> </head> <body><
header id="cabeceraweb"> <h1> Laboratorio AWS Juan David Navarro </h1> </header></body></html>
java.io.IOException: Server returned HTTP response code: 503 for URL: https://arep-lab5-aws.herokuapp.com/WebService/Web.html
<!DOCTYPE html> <html lang="es"> <head> <meta charset="iso-8559-1"/> <meta name="description" content="Pagina de prueba Html
5"/> <meta name="keywords" content="html5 , css3 , javaScrip, diseño web"/> <title>Aws</title> </head> <body><
header id="cabeceraweb"> <h1> Laboratorio AWS Juan David Navarro </h1> </header></body></html>
<!DOCTYPE html> <html lang="es"> <head> <meta charset="iso-8559-1"/> <meta name="description" content="Pagina de prueba Html
5"/> <meta name="keywords" content="html5 , css3 , javaScrip, diseño web"/> <title>Aws</title> </head> <body><
header id="cabeceraweb"> <h1> Laboratorio AWS Juan David Navarro </h1> </header></body></html>
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2.293 s
[INFO] Finished at: 2020-03-09T15:52:32Z
[INFO] Final Memory: 12M/29M
[INFO] -----
[ec2-user@ip-172-31-89-247 Arep_Lab5_AWS]$
```

Figura 7: Vemos el resultado de las solicitudes.

### **3. Conclusión**

Aprendimos a configurar algunos servicios de AWS en este caso utilizamos una instancia de una máquina virtual Linux usando el servicio EC2 de AWS.

Se mejoro el funcionamiento del servidor ya que es capaz de resolver solicitudes concurrentemente.

Se mejoro el funcionamiento del cliente ya que tiene la capacidad de realizar x solicitudes concurrentes a una URL dada.