Memoria de práctica

Servidor Donaciones (DSD P3S2)

Román Larrosa Lewandowska

marzo 2020

Introducción

En el siguiente documento se detalla el proceso de creación de un servidor replicado de donaciones.

Prueba

En el fichero se añade un archivo en el que se explica el proceso para probar el servidor con un cliente preparado para efectuar algunas operaciones y comprobar su funcionamineto y resultados. Para ejecutarlo:

IMPORTANTE EJECUTAR TODO **EN EL MISMO DIRECTORIO**

```
rmiregistry &
javac *.java
```

Terminal 1

```
java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost
-Djava.security.policy=server.policy Servidor_Principal servidor1 servidor2
```

Terminal 2

```
java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost
-Djava.security.policy=server.policy Servidor_Principal servidor2 servidor1
```

Terminal 3 (Cliente)

```
java -cp . -Djava.security.policy=server.policy Cliente
```

Contenidos

Para el desarrollo de la práctica he creado los siguientes archivos:

- server.policy: archivo donde se determinan los permisos que tendrán los ejecutables sobre el directorio de trabajo.
- Interfaz_cliente.java: Conjunto de operaciones del servidor que estarán disponibles para el cliente que use el servicio.

- Interfaz_replicas.java: Conjunto de operaciones del servidor que estarán disponibles para que otros servidores puedan relacionarse con él.
- Servidor.java: Implementación de los métodos declarados en Interfaz_cliente.java y en Interfaz_replicas.java.
- Servidor_Principal: proceso que pone en el registry un objeto de la clase Servidor.
- Cliente: este proceso, busca en el registry el objeto remoto con el que operará, y ejecuta las operaciones de prueba.

Interfaz_cliente.java

```
public interface Interfaz_cliente extends Remote {

    //OPERACIONES

    public boolean registrar (String entidad) throws RemoteException;

    public double getTotalDonaciones (String entidad) throws RemoteException;

    public boolean donar(String entidad, double donacion) throws
RemoteException;

}
```

Se declaran los métodos que podrán utilizarse desde el cliente, que son las operaciones principales que implementa el servicio: registrar una entidad en el servidor, conseguir el total de donaciones, y donar.

Interfaz_replicas.java

```
public interface Interfaz_replicas extends Remote {
    //OPERACIONES

    public double getTotalDonaciones (String entidad) throws RemoteException;
    //Devuelve el numero de registrados
    public int getRegistrados() throws RemoteException;
    //Devuelve el subtotal de donaciones de una replica
    public double getDonaciones() throws RemoteException;
    //Registra a un usuario en una réplica
    public boolean registrar(String entidad) throws RemoteException;
    //Realiza una donación en una réplica
    public boolean donar(String entidad, double donacion) throws RemoteException;
    //Comprueba si un usuario está registrado en una réplica
```

```
    public boolean registrado(String user) throws RemoteException;
    //Comprueba si un usuario ha donado en una réplica
    public boolean donado(String user) throws RemoteException;
}
```

Establece las operaciones que se podrán ejecutar entre los distintos servidores. Además de las mismas que el cliente, para poder reenviar operaciones de un servidor a otro, se añaden las operaciones para conocer si un usuario está registrado en un servidor, si ha donado algo, cuántos usuarios hay registrados en la réplica y conocer el subtotal de las donaciones en la réplica.

Servidor.java

Aquí se especifica la implementación de los métodos declarados en ambas interfaces. En los métodos en los que hay que modificar el estado del servidor (registrar un usuario, realizar una donación, etc.) y no solo informar sobre su estado, éste puede redirigir la acción a la réplica asignada. Por ejemplo:

```
 @Override
    public boolean registrar(String entidad) throws RemoteException {
        if(stub_replica == null){
            this.conectarReplica();
        }
        if(!this.registrado(entidad) && !stub_replica.registrado(entidad))
{
            if(this.getRegistrados() <= stub_replica.getRegistrados()){
                //Se registra en esta replica
                entidades.put(entidad, 0.0);
                System.out.println("Se registra a la entidad " + entidad);
                return true;
            }
            else{
                //Se registra en la otra replica
                return stub_replica.registrar(entidad);
            }
        }
        return false;
    }
```

Para ello, existe un método para conectarse a la réplica si no se había conectado con anterioridad que se le asigna en el momento de inicializar la variable del servidor. Esta réplica podría convertirse en un array de réplicas para que hubiese un servidor “principal” y otros subordinados para realizar la gestión del servicio de donaciones de manera replicada en más de dos servidores.

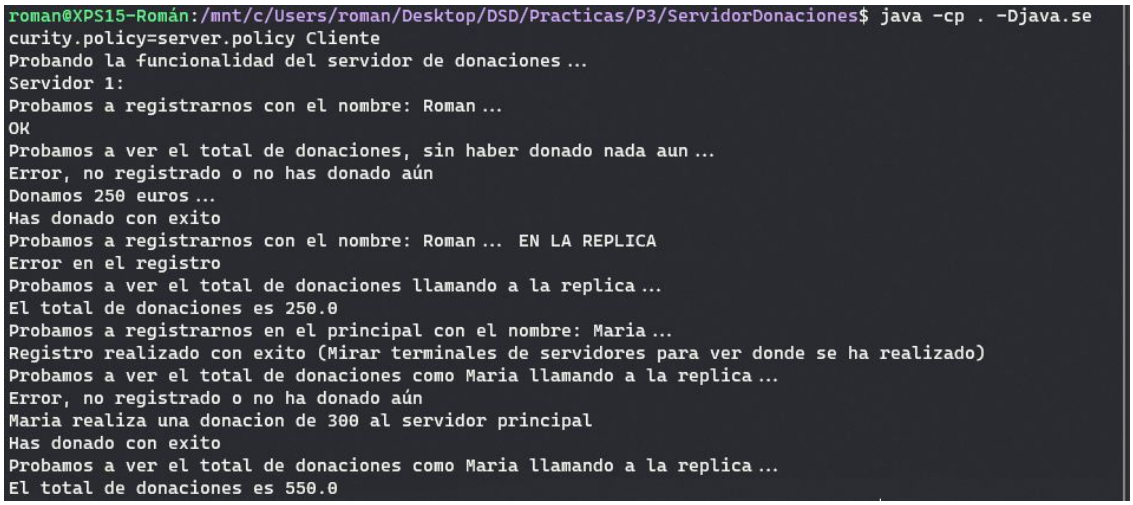
Servidor_Principal.java

Este proceso lee desde la consola el nombre que va a tomar (para subir su Servidor al registry) y el que tiene su réplica para poder buscarla desde su objeto Servidor en el registry. A continuación, toma el registry y hace un rebind de su objeto servidor para que quede a disposición del cliente y del otro servidor.

Cliente.java

El cliente busca en el registry del host indicado, en este caso el localhost, un objeto servidor. A continuación ejecuta un conjunto de operaciones para probar la funcionalidad de los servidores replicados. La traza de su ejecución y la de los servidores, para comprobar que las operaciones se realizan donde deben independientemente de al servidor al que se llame, puede observarse aquí:

Cliente:



Servidor 1:

```
roman@XPS15-Román:/mnt/c/Users/roman/Desktop/DSD/Practicas/P3/ServidorDonaciones$ java -cp . -Djava.rm
i.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Se
rvidor_Principal servidor1 servidor2
servidor1 bind: OK
Se registra a la entidad Roman
El usuario Roman NO ha donado nada
La entidad Romanrealiza una donacion de 250.0
```

Servidor 2:

```
^Croman@XPS15-Román:/mnt/c/Users/roman/Desktop/DSD/Practicas/P3/ServidorDonaciones$ java -cp . -Djava.
i.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Se
rvidor_Principal servidor2 servidor1
servidor2 bind: OK
Se registra a la entidad Maria
El usuario Maria NO ha donado nada
La entidad Mariarealiza una donacion de 300.0
```