Práctica 3. Segunda Parte: Servidor Replicado

Noelia Escalera Mejías

24 de abril de 2020

1. Método de comunicación

Para que las dos réplicas cooperen entre sí, se ha creado un método buscar réplica, que devolverá la otra réplica o null si no la encuentra. Se llamará a este método cada vez que nos queramos comunicar con otra réplica.

2. Funcionamiento del cliente

1. Lo primero que haremos en el cliente será buscar una réplica del servidor (se ha elegido la réplica 1 por defecto).

```
String remoteObjectName = "Replica1";

System.out.println("Searching remote object");
Registry registry = LocateRegistry.getRegistry("localhost",1099);
Replica_I localObject = (Replica_I) registry.lookup(remoteObjectName); // Buscamos la réplica 1
```

2. Preguntamos al cliente si quiere registrarse o iniciar sesión

```
do{
    System.out.println("1: Registrarse\n2: Iniciar sesión\n");
    opcion = s.next();
    s.nextLine();
} while (!opcion.equals("1") && !opcion.equals("2")); // Mientas la opción elegida no esté disponible
```

3. Si el cliente elige registrarse, pediremos un nombre de usuario (no puede estar ya escogido) y una contraseña.

```
if (opcion.equals("1")){ // Registro
    do{
        System.out.print("Elige un nombre de usuario: "); // Pedimos un nombre de usuario
        nombre_us = s.next();
        s.nextLine();
        existe = localObject.existeUsuario(nombre_us); // Comprobamos que el usuario no exista
    }while (existe);

    System.out.print("Elige una contraseña: "); // Pedimos la contraseña
    pass_us = s.next();
    s.nextLine();
    localObject.registrarUsuario(nombre_us,pass_us);
}
```

4. Si el usuario elige inicio de sesión, se le pedirán los credenciales y no se le dejará entrar hasta que sean correctos.

```
else if (opcion.equals("2")){ // Inicio de sesión
    do{
        System.out.print("Introduce nombre de usuario: "); // Pedimos el nombre de usuario
        nombre_us = s.next();
        s.nextLine();
        System.out.print("Introduce tu contraseña: "); // Pedimos la contraseña
        pass_us = s.next();
        s.nextLine();
        credenciales = localObject.credencialesCorrectos(nombre_us,pass_us); // Comprobamos que estos
    }while (!credenciales);                                                  // credenciales son  correctos
}
```

5. Una vez hecho este paso, comprobaremos la réplica en la que está registrada el usuario y a partir de ahora haremos todas las consultas a ella.

```
server = localObject.getServerUsuario(nombre_us); // Comprobamos en qué réplica está registrado el usuario
if (server.equals("2")){ // Conectamos con una réplica u otra, dependiendo de cuál sea
    serverUsuario = (Replica_I) registry.lookup("Replica2");
}
else{
    serverUsuario = (Replica_I) registry.lookup("Replica1");
}
```

6. Luego preguntaremos qué acción se quiere realizar, esto se repetirá hasta que el usuario quiera salir

```java
do{
    System.out.println("1: Donar\n2: Consultar donaciones\n3: Salir"); // Preguntamos la acción a realizar
    opcion = s.next();
    s.nextLine();
    if (opcion.equals("1")){ // Donar
        System.out.print("Elige la cantidad a donar: ");
        String dinero = s.next();
        s.nextLine();
        serverUsuario.aniadeDonacion(Float.parseFloat(dinero),nombre_us);
    }
    if (opcion.equals("2")){ // Consultar donaciones
        float valor = serverUsuario.getTotalDonado(nombre_us);
        if (valor == -1){ // getTotalDonado devuelve -1 si el usuario aún no ha hecho ninguna donación
            System.out.println("Tienes que haber realizado una donación para realizar esta acción\n");
        }
        else{
            System.out.println("Hay "+valor+" euros donados\n");
        }
    }
} while (opcion.equals("1") || opcion.equals("2") || !opcion.equals("3")); // hasta que la opción sea salir

s.close(); // Cerramos el escáner
```

3. Explicación de los métodos del servidor

Estos son los métodos implementados en el servidor:

```java
public interface Replica_I extends Remote {
    public float getDinero () throws RemoteException;
    public void registrarUsuario (String nombre, String pass) throws RemoteException;
    public void aniadeDonacion (float valor, String usuario) throws RemoteException;
    public Replica_I buscarReplica() throws RemoteException;
    public ArrayList<Usuario> getUsuarios () throws RemoteException;
    public ArrayList<String> getNombresUsuario () throws RemoteException;
    public Boolean existeUsuario (String nombre) throws RemoteException;
    public Boolean credencialesCorrectos (String nombre, String pass) throws RemoteException;
    public String getServerUsuario (String nombre) throws RemoteException;
    public float getTotalDonado (String usuario) throws RemoteException;
}
```

Se pondrá de ejemplo la implementación en la réplica 1, la implementación en la réplica 2 es similar

1. **getDinero.** Devuelve el dinero donado en esa réplica.

```java
public float getDinero(){
    return dinero; // Devolvemos el dinero donado en esta réplica
}
```

2. **registrarUsuario.** Registramos un usuario en el sistema.

```java
public void registrarUsuario(String nombre, String pass){
    Replica_I localObject = buscarReplica(); // Buscamos la réplica
    ArrayList<String> usu_rep2 = new ArrayList<>();
    Usuario user = new Usuario (); // Creamos un usuario con el nombre y
    user.setNombre(nombre);        // la contraseña que nos han pasado
    user.setPass(pass);
    try{
        if (localObject != null){ // Si la réplica existe
            usu_rep2 = localObject.getNombresUsuario(); // Buscamos los nombres de usuario de la otra réplica
            if (usu_rep2.size()>nombres_usuario.size()){ // Si esta réplica tiene menos usuarios
                usuarios.add(user); // Añadimos el usuario
                nombres_usuario.add(nombre); // Añadimos también su nombre a la lista
                System.out.println("Registro contabilizado en esta réplica"); // Informamos del registro
            }
            else{ // Si esta réplica no tiene menos usuarios
                localObject.registrarUsuario(nombre,pass); // Lo registramos en la otra
            }
        }
        else{ // Si la otra réplica no existe registramos al usuario en esta
            usuarios.add(user);
            nombres_usuario.add(nombre);
            System.out.println("Registro contabilizado en esta réplica");
        }
    } catch (Exception e){
        System.err.println("RegistrarUsuario exception:");
        e.printStackTrace();
    }
}
```

3. **aniadeDonacion.** Un usuario dona al servidor.

```java
public void aniadeDonacion(float valor, String usuario){
    try{
        Replica_I localObject = buscarReplica(); // Buscamos la otra réplica
        Usuario user = new Usuario();
        int indice = nombres_usuario.indexOf(usuario); // Buscamos el usuario
        user = usuarios.get(indice);                   // que nos ha donado
        user.setDonacion(); // Informamos de que el usuario ha hecho una donacion
    } catch (Exception e){
        System.err.println("aniadeDonacion exception:");
        e.printStackTrace();
    }
    dinero += valor; // Sumamos el dinero de la donación a la réplica
    donaciones++; // Contabilizamos otradonación en la réplica
    System.out.println("He recibido una donación de "+valor+" euros");
    System.out.println("Me han hecho "+donaciones+" donaciones");
    System.out.println("Tengo "+dinero+" euros");
}
```

4. **buscarReplica.** Buscamos la otra réplica.

```java
public Replica_I buscarReplica(){
    Registry registry;
    Replica_I localObject = null;
    try{
        registry = LocateRegistry.getRegistry("localhost");
        localObject = (Replica_I) registry.lookup("Replica2");
    } catch (Exception e){
        System.err.println("buscarReplica exception:");
        e.printStackTrace();
    }

    if (localObject!=null){
        return localObject;
    }
    else
        return null;
}
```

5. **getUsuarios.** Consultamos la lista de entidades usuario de una réplica.

```java
public ArrayList<Usuario> getUsuarios (){
    return usuarios;
}
```

6. **getNombresUsuario.** Consultamos la lista de nombres de usuario de una réplica.

```java
public ArrayList<String> getNombresUsuario(){
    return nombres_usuario;
}
```

7. **existeUsuario.** Comprobamos si un usuario existe.

```java
public Boolean existeUsuario(String nombre){
    Boolean existe = false;
    try{
        if (nombres_usuario.size()>0){ // Comprobamos si hay algún usuario registrado en esta réplica
            if (nombres_usuario.contains(nombre)) // Comprobamos si la lista de nombres contiene al usuario
                existe = true;
            else{
                Replica_I localObject = buscarReplica(); // Buscamos la réplica
                ArrayList<String> usu_rep2 = new ArrayList<>();
                if (localObject!=null){ // Si la réplica existe
                    usu_rep2 = localObject.getNombresUsuario(); // Cogemos la lista de nombres de usuario de la réplica
                    if (usu_rep2.contains(nombre)){ // Comprobamos si la réplica contine ese nombre de usuario
                        existe = true;
                    }
                }
            }
        }
        else{ // Si no buscamos en la réplica
            Replica_I localObject = buscarReplica();
            ArrayList<String> usu_rep2 = new ArrayList<>();
            if (localObject!=null){
                usu_rep2 = localObject.getNombresUsuario();
                if (usu_rep2.contains(nombre)){
                    existe = true;
                }
            }
        }
    } catch (Exception e){
        System.err.println("existeUsuario exception:");
        e.printStackTrace();
    }

    return existe;
}
```

8. **credencialesCorrectos.** Comprueba si los credenciales introducidos al iniciar sesión son correctos.

```java
public Boolean credencialesCorrectos (String nombre, String pass){
    Boolean correctos = false;
    int indice = 0;
    try{
        if (existeUsuario (nombre)){ // Comprobamos que existe el usuario en alguna de las réplicas
            if (nombres_usuario.size()>0){ // Nos aseguramos de que tenemos algún usuario en esta réplica
                if (nombres_usuario.contains(nombre)){ // Si tenemos ese usuario registrado
                    indice = nombres_usuario.indexOf(nombre);
                    if (usuarios.get(indice).getPass().equals(pass)) // Miramos si la contraseña coincide
                        correctos = true;
                }
                else { // Si no lo tenemos registrado, buscamos en la otra réplica
                    Replica_I localObject = buscarReplica(); // Buscamos la réplica
                    if (localObject!=null){ // Si existe
                        correctos = localObject.credencialesCorrectos(nombre,pass); // Miramos en la otra réplica
                    }
                }
            }
            else { // Si no tenemos usuarios en esta réplica miramos en la otra
                Replica_I localObject = buscarReplica();
                if (localObject!=null){
                    correctos = localObject.credencialesCorrectos(nombre,pass);
                }
            }
        }
    } catch (Exception e){
        System.err.println("credencialesCorrectos exception:");
        e.printStackTrace();
    }

    return correctos;
}
```

9. **getServerUsuario.** Comprobamos en qué réplica está registrado un usuario.

```java
public String getServerUsuario(String nombre){
    String server = "";
    try{
        if (nombres_usuario.size()>0){ // Nos aseguramos de que tenemos algún usuario registrado
            if (nombres_usuario.contains(nombre)) // Comprobamos si tenemos ese usuario en esta réplica
                server = "1";
            else{ // Si no lo tenemos en esta réplica
                Replica_I localObject = buscarReplica(); // Buscamos la otra réplica
                ArrayList<String> usu_rep2 = new ArrayList<>();
                if (localObject!=null){ // Comprobamos que existe
                    usu_rep2 = localObject.getNombresUsuario(); // Pedimos los nombres de los usuarios de la otra réplica
                    if (usu_rep2.contains(nombre)){ // Comprobamos que está en la otra réplica
                        server = "2";
                    }
                }
            }
        }
        else{ // Si no tenemos ningún usuario registrado aquí, miramos en la otra réplica
            Replica_I localObject = buscarReplica();
            ArrayList<String> usu_rep2 = new ArrayList<>();
            if (localObject!=null){
                usu_rep2 = localObject.getNombresUsuario();
                if (usu_rep2.contains(nombre)){
                    server = "2";
                }
            }
        }
    } catch (Exception e){
        System.err.println("existeUsuario exception:");
        e.printStackTrace();
    }

    return server;
}
```

10. **getTotalDonado.** Comprobamos el total de dinero donado a las 2 réplicas.

```java
public float getTotalDonado(String usuario){
    float total = 0;
    try{
        Usuario user = new Usuario();
        int indice = nombres_usuario.indexOf(usuario); // Buscamos al usuario que ha donado, sabemos que existe,
        user = usuarios.get(indice);                   // ya que al llegar a este punto se ha debido de identificar
        if(user.getDonacion()){ // Comprobamos si ese usuario ya ha donado
            Replica_I localObject = buscarReplica(); // Buscamos la otra réplica
            total = localObject.getDinero() + dinero; // Sumamos el dinero de esta réplica y de la otra
        }
        else{ // Si no ha realizado ninguna donación, no puede ver el total donado
            total = -1;
        }
    } catch (Exception e){
        System.err.println("getTotalDonado exception:");
        e.printStackTrace();
    }

    return total;
}
```

4. Ejemplo de ejecución

Lanzamos la réplica 1:

```
noella@noella-HP-ENVY-17-Notebook-PC:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 2$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=
server.policy Replica1
Replica 1 lista
```

Y la réplica 2:

```
noella@noella-HP-ENVY-17-Notebook-PC:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 2$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=
server.policy Replica2
Replica 2 lista
```

Lanzamos el cliente y nos registramos:

```
noelia@noelia-HP-ENVY-17-Notebook-PC:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 2$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=
server.policy Cliente localhost
Searching remote object
1: Registrarse
2: Iniciar sesión

1
Elige un nombre de usuario: noelia
Elige una contraseña: pass
1: Donar
2: Consultar donaciones
3: Salir
```

La réplica 2 nos avisa de que se han registrado en ella:

```
noelia@noelia-HP-ENVY-17-Notebook-PC:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 2$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=
server.policy Replica2
Replica 2 lista
Registro contabilizado en esta réplica
```

Si ahora intentamos registrar otro usuario con el mismo nombre no nos dejará, habrá que elegir otro:

```
^Cnoelia@noelia-HP-ENVY-17-Notebook-PC:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 2$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.polic
server.policy Cliente localhost
Searching remote object
1: Registrarse
2: Iniciar sesión

1
Elige un nombre de usuario: noelia
Elige un nombre de usuario: user
Elige una contraseña: pass
1: Donar
2: Consultar donaciones
3: Salir
```

Ahora será la réplica 1 la que ha contabilizado el registro:

```
noelia@noelia-HP-ENVY-17-Notebook-PC:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 2$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=
server.policy Replica1
Replica 1 lista
Registro contabilizado en esta réplica
```

Si intentamos iniciar sesión con unos credenciales incorrectos no nos dejará:

```
noelia@noelia-HP-ENVY-17-Notebook-PC:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 2$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=
server.policy Cliente localhost
Searching remote object
1: Registrarse
2: Iniciar sesión

2
Introduce nombre de usuario: noexisto
Introduce tu contraseña: pass
Introduce nombre de usuario: noelia
Introduce tu contraseña: mal
Introduce nombre de usuario: noelia
Introduce tu contraseña: pass
1: Donar
2: Consultar donaciones
3: Salir
```

Probamos ahora a hacer una donación:

```
1: Donar
2: Consultar donaciones
3: Salir
1
Elige la cantidad a donar: 40
1: Donar
2: Consultar donaciones
3: Salir
```

La réplica 2 avisa de la donación que ha recibido, ya que es donde está registrado este usuario.

```
^Cnoelia@noelia-HP-ENVY-17-Notebook-PC:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 2$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.polic
server.policy Replica2
Replica 2 lista
Registro contabilizado en esta réplica
He recibido una donación de 40.0 euros
He han hecho 1 donaciones
Tengo 40.0 euros
```

Si ahora intentamos consultar el total donado con el otro usuario no nos va a dejar, ya que este usuario no ha hecho ninguna donación:

```
noelia@noelia-HP-ENVY-17-Notebook-PC:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 2$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=
server.policy Cliente localhost
Searching remote object
1: Registrarse
2: Iniciar sesión

2
Introduce nombre de usuario: user
Introduce tu contraseña: pass
1: Donar
2: Consultar donaciones
3: Salir
2
Tienes que haber realizado una donación para realizar esta acción

1: Donar
2: Consultar donaciones
3: Salir
```

Probamos a donar con este usuario:

```
1: Donar
2: Consultar donaciones
3: Salir
1
Elige la cantidad a donar: 3
1: Donar
2: Consultar donaciones
3: Salir
```

Ahora será la réplica 1 la que recogerá la donación:

```
noelia@noelia-HP-ENVY-17-Notebook-PC:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 2$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=
server.policy Replica1
Replica 1 lista
Registro contabilizado en esta réplica
He recibido una donación de 3.0 euros
Me han hecho 1 donaciones
Tengo 3.0 euros
```

Ya sí nos dejará consultar el total donado:

```
1: Donar
2: Consultar donaciones
3: Salir
2
Hay 43.0 euros donados

1: Donar
2: Consultar donaciones
3: Salir
```