

Universidad de Granada

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y
DE TELECOMUNICACIÓN

PRÁCTICA 3
REMOTE METHOD INVOCATION
(RMI)

Desarrollo de Sistemas Distribuidos

Autor:
Raquel Molina Reche

Mayo 2021

Índice

I	Implementación de los ejemplos	1
1.	Ejemplo 1	1
2.	Ejemplo 2: Multihebra	3
3.	Ejemplo 3	6
II	Servidor replicado	9
4.	Planteamiento de la solución	9
5.	Métodos	12
5.1.	Peticiones Cliente-Servidor	12
5.2.	Peticiones Servidor-Servidor	13
6.	Ejemplos de ejecución	14
6.1.	Ejemplo	15

Introducción

Los contenidos de esta práctica comprenden el manejo de mecanismos para la implementación de programas distribuidos en Java utilizando la API RMI (Remote Method Invocation).

RMI es un mecanismo ofrecido por Java para la invocación de métodos de forma remota, en el que un programa exporta un objeto a la red y se mantiene a la espera de peticiones, y un cliente se puede conectar e invocar los métodos del objeto exportado.

Esta práctica consta de dos partes: una para la prueba de ejemplos e iniciación a RMI y una segunda parte que consiste en la implementación de una aplicación Cliente-Servidor de un servidor replicado para la gestión de donaciones de entidades (clientes) para una causa humanitaria.

Parte I

Implementación de los ejemplos

1. Ejemplo 1

En este ejercicio cuando el servidor recibe una petición del cliente imprime el argumento que acompaña a la petición, y en caso de terminar este en “0” espera 5 segundos antes de imprimir el mensaje.

Para ello el servidor *Ejemplo* implementa la interfaz *Ejemplo_I* que será invocada por el cliente y exporta un objeto de tipo *Ejemplo* para la comunicación.

La interfaz únicamente tiene el método *escribir_mensaje* que comprueba el valor del argumento recibido y muestra el mensaje , para el caso en el que el argumento es 0 espera 5 segundos antes de imprimir por pantalla el mensaje.

El cliente se conecta al canal de comunicación instanciando la interfaz y llama al método *escribir_mensaje* de la misma.

Compilación `$javac *.java`

Ejecución

1. `$rmiregistry`
2. Ejecución del servidor: `$java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Ejemplo`
3. Ejecución del cliente: `$java -cp . -Djava.security.policy=server.policy Cliente.Ejemplo localhost 0`

Para la ejecución del servidor se especifica:

- La ubicación desde la que poder descargar definiciones de clases desde el servidor.
- El nombre de la máquina donde colocar los stubs para los objetos remotos.
- El fichero de políticas de seguridad que se pretenden seguir o conceder.

Para la ejecución del cliente se especifica:

- El fichero de políticas de seguridad que se pretenden seguir o conceder.
- El nombre de la máquina (host) donde se encuentra el servidor.
- El valor del argumento de la llamada.

En la ejecución se observa que cuando el cliente hace una llamada con el argumento “0” este tarda el tiempo que el servidor está en espera antes de terminar. Para el resto la llamada termina de forma inmediata pues el servidor no se pone en espera.

```

raquelmolire@LAPTOP-TC16343C:/mnt/d/3° GII/DSD/Practicas/P3_RMI/ejemplo1
$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostna
me=localhost -Djava.security.policy=server.policy Ejemplo
Ejemplo bound
Recibida peticion de proceso: 0
Empezamos a dormir
Terminamos de dormir

Hebra 0
Recibida peticion de proceso: 1

Hebra 1
Recibida peticion de proceso: 2

Hebra 2
Recibida peticion de proceso: 0
Empezamos a dormir
Terminamos de dormir

Hebra 0

```

Figura 1: Ejecución servidor ejemplo 1

```

raquelmolire@LAPTOP-TC16343C:/mnt/d/3° GII/DSD/Practicas/P3_RMI/ejemplo1$ java -cp .
-Djava.security.policy=server.policy Cliente_Ejemplo localhost 0
Buscando el objeto remoto
Invocando el objeto remoto
raquelmolire@LAPTOP-TC16343C:/mnt/d/3° GII/DSD/Practicas/P3_RMI/ejemplo1$ java -cp .
-Djava.security.policy=server.policy Cliente_Ejemplo localhost 1
Buscando el objeto remoto
Invocando el objeto remoto
raquelmolire@LAPTOP-TC16343C:/mnt/d/3° GII/DSD/Practicas/P3_RMI/ejemplo1$ java -cp .
-Djava.security.policy=server.policy Cliente_Ejemplo localhost 2
Buscando el objeto remoto
Invocando el objeto remoto
raquelmolire@LAPTOP-TC16343C:/mnt/d/3° GII/DSD/Practicas/P3_RMI/ejemplo1$ java -cp .
-Djava.security.policy=server.policy Cliente_Ejemplo localhost 0
Buscando el objeto remoto
Invocando el objeto remoto

```

Figura 2: Ejecución cliente ejemplo 1

2. Ejemplo 2: Multihebra

En este ejercicio es igual al anterior cambiando que el papel de clientes lo realizan hebras. Por lo que el código sigue el mismo objetivo, se diferencian en que el cliente es una hebra y en su método run invoca al método de la interfaz *Ejemplo_I* y en el que el argumento es el nombre de la hebra. Esta clase tendrá un main en el que se crearán y ejecutarán tantas hebras como se indique como argumento en la ejecución.

Compilación \$javac *.java

Ejecución

1. \$rmiregistry
2. Ejecución del servidor: \$java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Ejemplo

3. Ejecución del cliente: `$java -cp . -Djava.rmi.server.codebase=file:///./ -Djava.security.policy=server.policy Cliente_Ejemplo_Multi_Threaded localhost 11`

Para la ejecución del servidor se especifica:

- La ubicación desde la que poder descargar definiciones de clases desde el servidor.
- El nombre de la máquina donde colocar los stubs para los objetos remotos.
- El fichero de políticas de seguridad que se pretenden seguir o conceder.

Para la ejecución del cliente se especifica:

- El fichero de políticas de seguridad que se pretenden seguir o conceder.
- El nombre de la máquina (host) donde se encuentra el servidor.
- El número de hebras que se quieren crear.

En la ejecución se observa como en el servidor se ejecutan las hebras y en el caso del “0” y del “10” ambos inician su tiempo de dormir pero durante ese tiempo otras hebras se ejecutan siguiendo un modo asíncrono.

```
raquelw@laptop-T16343C:/mnt/d/3* GII/OSD/Practicas/P3_RMI/ejemplo2$ java -c
p . -Djava.rmi.server.codebase=file:/// -Djava.rmi.server.hostname=localhost -Djav
a.security.policy=server.policy Ejemplo
Ejemplo bound
Entra Hebra Cliente 2
Sale Hebra Cliente 2
Entra Hebra Cliente 1
Sale Hebra Cliente 1
Entra Hebra Cliente 0
Empezamos a dormir
Entra Hebra Cliente 6
Sale Hebra Cliente 6
Entra Hebra Cliente 10
Empezamos a dormir
Entra Hebra Cliente 3
Sale Hebra Cliente 3
Entra Hebra Cliente 5
Sale Hebra Cliente 5
Entra Hebra Cliente 8
Sale Hebra Cliente 8
Entra Hebra Cliente 4
Sale Hebra Cliente 4
Entra Hebra Cliente 7
Sale Hebra Cliente 7
Entra Hebra Cliente 9
Sale Hebra Cliente 9
Terminamos de dormir
Sale Hebra Cliente 0
Terminamos de dormir
Sale Hebra Cliente 10
```

Figura 3: Ejecución servidor ejemplo 2

[illegible]

Figura 4: Ejecución cliente ejemplo 2

Aplicando la modificación de la [Figura 5](#), en la que se incluye al método *escribir_mensaje* la característica *synchronized* se ve como en la ejecución todas las hebras se ejecutan coordinadas en el tiempo, de manera que para las hebras “0” y “10” hasta que estas no terminan de dormir y por tanto de ejecutarse no entra otra hebra.

```
//Implementacion de la interfaz
public synchronized void escribir_mensaje(String mensaje){
    //Se recibe una peticion
    System.out.println("\nEntra Hebra "+mensaje);

    //Si termina por 0 (0, 10, 20...)
    if(mensaje.endsWith("0")){
        //Se duerme durante 5 segundos
        try{
            System.out.println("Empezamos a dormir");
            Thread.sleep(5000);
            System.out.println("Terminamos de dormir");
        }
        catch(Exception e){
            System.err.println("Ejemplo de exception:");
            e.printStackTrace();
        }
    }

    //Se muestra el la hebra
    System.out.println("\nSale Hebra "+mensaje);
}
```

Figura 5: Modificación para la ejecución síncrona

La interfaz tiene los métodos para poner el contador a 0, sumar un valor al contador e incrementar el contador.

El cliente se conecta al canal de comunicación instanciando la interfaz y llama a los métodos correspondientes de la misma para poner el contador al valor inicial 0 y realizar los 1000 incrementos, calculando el tiempo de respuesta entre el inicio de la petición al servidor y la obtención de los resultados.

Por lo que como se ve en este caso se mantiene una separación entre clases teniendo por un lado la interfaz que llamará el cliente (*icontador*) el objeto que exportará el servidor para la comunicación (*contador*) y las clases correspondientes *cliente* y *servidor*.

Compilación `$javac *.java`

Ejecución

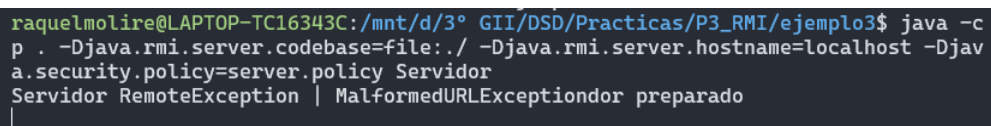
1. Ejecución del servidor: `$java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Servidor`
2. Ejecución del cliente: `$java -cp . -Djava.security.policy=server.policy Cliente`

Para la ejecución del servidor se especifica:

- La ubicación desde la que poder descargar definiciones de clases desde el servidor.
- El nombre de la máquina donde colocar los stubs para los objetos remotos.
- El fichero de políticas de seguridad que se pretenden seguir o conceder.

Para la ejecución del cliente se especifica:

- La ubicación desde la que poder descargar definiciones de clases desde el cliente.
- El fichero de políticas de seguridad que se pretenden seguir o conceder.



```
raquelmolire@LAPTOP-TC16343C:/mnt/d/3º GII/DSD/Practicas/P3_RMI/ejemplo3$ java -c
p . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djav
a.security.policy=server.policy Servidor
Servidor RemoteException | MalformedURLExceptionor preparado
```

Figura 8: Ejecución servidor ejemplo 3


```
raquelmolire@LAPTOP-TC16343C:/mnt/d/3º GII/DSD/Practicas/P3_RMI/ejemplo3$ java
-cp . -Djava.rmi.server.codebase=file:///./ -Djava.security.policy=server.polic
y Cliente
Poniendo contador a 0
Incrementando...
Media de las RMI realizadas = 0.19 msecs
RMI realizadas = 1000
raquelmolire@LAPTOP-TC16343C:/mnt/d/3º GII/DSD/Practicas/P3_RMI/ejemplo3$ java
-cp . -Djava.rmi.server.codebase=file:///./ -Djava.security.policy=server.polic
y Cliente
Poniendo contador a 0
Incrementando...
Media de las RMI realizadas = 0.136 msecs
RMI realizadas = 1000
raquelmolire@LAPTOP-TC16343C:/mnt/d/3º GII/DSD/Practicas/P3_RMI/ejemplo3$ java
-cp . -Djava.rmi.server.codebase=file:///./ -Djava.security.policy=server.polic
y Cliente
Poniendo contador a 0
Incrementando...
Media de las RMI realizadas = 0.167 msecs
RMI realizadas = 1000
```

Figura 9: Ejecución cliente ejemplo 3

Parte II

Servidor replicado

Esta parte consiste en desarrollar, en RMI, un sistema Cliente-Servidor para la gestión de donaciones de entidades (clientes) para una causa humanitaria.

El servidor es un servidor replicado que recibirá peticiones de entidades clientes, tanto para el registro como para realizar donaciones u otro tipo de consultas que se detallan a continuación.

Los requisitos a tener en cuenta engloban:

- La solicitud de registro se realiza en cualquier réplica, pero se lleva a cabo, de forma transparente, en aquella en la que tenga menos entidades registradas.
- Una vez el cliente está registrado las peticiones se realizan al servidor en el que ha sido registrado.
- Cada réplica del servidor mantendrá el subtotal de las donaciones realizadas en dicha réplica.
- Un cliente no podrá registrarse más de una vez, ni siquiera en réplicas distintas.
- La consulta del total donado en un momento dado sólo podrá llevarse a cabo si el cliente previamente se ha registrado y ha realizado al menos un depósito. En este momento las réplicas deberán comunicarse para obtener la información necesaria y devolvérsela al cliente.

4. Planteamiento de la solución

Para la realización de esta aplicación se ha tenido en cuenta la separación de dos canales de comunicación diferentes, uno para la comunicación Cliente-Servidor y otro para la comunicación Servidor-Servidor. De esta manera se han implementado dos interfaces distintas que ofrecen los métodos que serán requeridos para sendos tipos de comunicación, manteniendo la separación entre las necesidades de los clientes respecto a las de los servidores o réplicas. Teniendo en cuenta que los servidores actúan de servidores, de cara a las entidades clientes, y también actúan de clientes del resto de servidores para la comunicación del sistema completo.

Para la comunicación Cliente-Servidor se ha implementado la interfaz *idonacion* que contiene las operaciones que los servidores proporcionan a los clientes. Estas operaciones se pueden ver en la [Figura 10](#).

```

public interface idonacion extends Remote{
    public HashMap<String, String> registrar(String cliente) throws RemoteException;
    public HashMap<String, String> iniciarSesion(String cliente) throws RemoteException;
    public boolean donar(String cliente, double cantidad) throws RemoteException;
    public double getCantidadCliente(String cliente) throws RemoteException;
    public double getCantidadReplica() throws RemoteException;
    public double getCantidadSistema() throws RemoteException;
    public int getNumDonacionesReplica() throws RemoteException;
    public int getNumDonacionesSistema() throws RemoteException;
    public int getNumClientesReplica() throws RemoteException;
    public int getNumClientesSistema() throws RemoteException;
    public String getNombreMayorDonacionReplica() throws RemoteException;
    public String getNombreMayorDonacionSistema() throws RemoteException;
    public double getCantidadMayorDonacionReplica() throws RemoteException;
    public double getCantidadMayorDonacionSistema() throws RemoteException;
    public boolean darDeBaja(String cliente) throws RemoteException;
    public void registrarDefinitiva(String cliente) throws RemoteException;
}

```

Figura 10: Interfaz comunicación Cliente-Servidor

Para la comunicación Servidor-Servidor se ha implementado la interfaz *iservidor* que contiene las operaciones que necesitan los servidores para conocer la información del sistema total. Estas operaciones se pueden ver en la Figura 11.

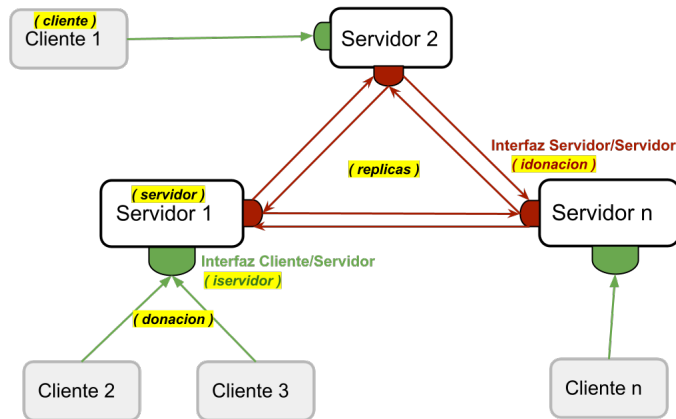
```

public interface iservidor extends Remote{
    public void addReplica(String host, String nombreReplica, int puerto) throws RemoteException;
    public HashMap<String, String> registrar(String cliente) throws RemoteException;
    public boolean yaRegistrado(String cliente) throws RemoteException;
    public void registrarDonacion(String nombreReplica, String cliente, double cantidad) throws RemoteException;
    public HashMap<String, String> iniciarSesion(String cliente) throws RemoteException;
    public double getcantidadTotal() throws RemoteException;
    public int getNumDonacionesTotal() throws RemoteException;
    public int getNumClientesTotal() throws RemoteException;
    public String getNombreMayorDonacionTotal() throws RemoteException;
    public double getCantidadMayorDonacionTotal() throws RemoteException;
    public boolean darDeBaja(String cliente) throws RemoteException;
}

```

Figura 11: Interfaz comunicación Servidor-Servidor

En la Figura 12 se muestra un diagrama del sistema en el que se incluyen los roles de las distintas clases que se han implementado y sus nombres. Y en la Figura 13 se muestra el diagrama con los distintos métodos que lleva asociado cada canal de comunicación.



* El nombre de las interfaces y clases que realizan cada papel están subrayadas en amarillo

Figura 12: Diagrama roles del Sistema

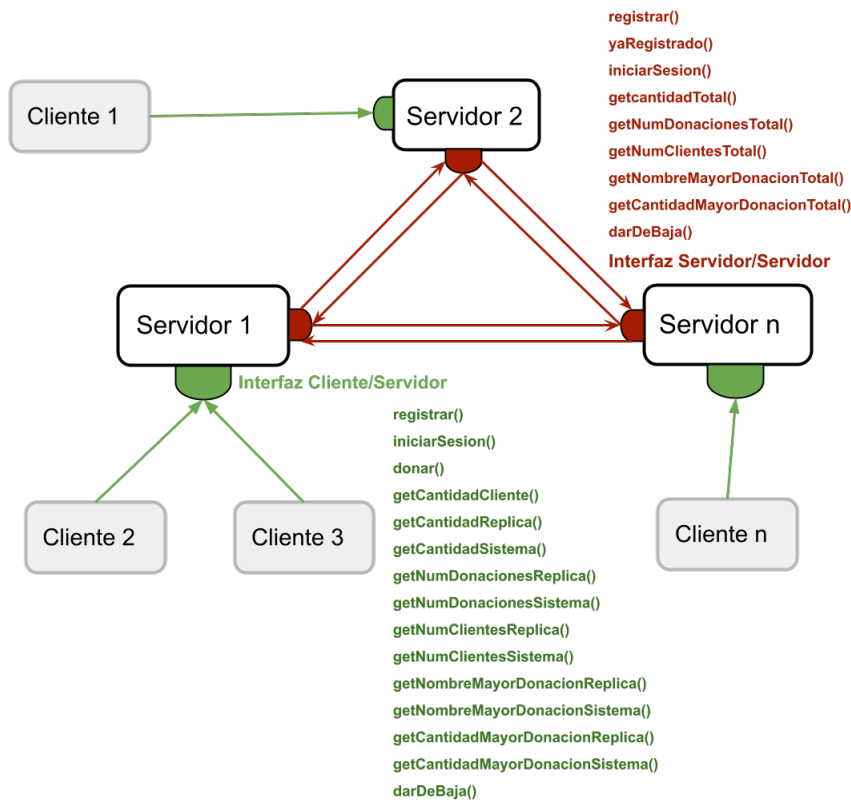


Figura 13: Diagrama métodos del sistema

Cabe destacar que se ha extendido la funcionalidad de modo que se pueden usar un número **ilimitado** de réplicas(o lo que es lo mismo, servidores). Y para ello la técnica que se ha usado consiste en que el primer servidor que se inicia es el encargado de crear y mantener el canal de comunicación entre los distintos servidores que se iniciaran en el sistema. Cada servidor indistintamente crea el canal de comunicación para la recepción de peticiones de los clientes y se conecta al canal de comunicación de los servidores. Para mantener la integridad de la información en el sistema, se realiza una actualización de datos constante.

Cuando el primer servidor se inicia, este crea el canal que comunicará a los servidores y exporta un objeto del tipo *replicas* que implementa los métodos de la interzar *iservidor*, que como se ha dicho anteriormente es la interfaz para la comunicación servidor-servidor.

El resto de servidores cuando se inician se conectan a ese canal de comunicación interaccionando con ese objeto remoto por medio de la interfaz *iservidor*.

Indistintamente todos los servidores crean su propio canal de comunicación hacia los clientes, exportando un objeto de tipo *donación*.

Los clientes interaccionan con los objetos remotos exportados por los servidores por medio de la interfaz *idonacion*.

5. Métodos

5.1. Peticiones Cliente-Servidor

5.1.1. registrar(String cliente)

El servidor pregunta a todo el sistema si el cliente ya está registrado, en caso de no estarlo pregunta al sistema donde se debe registrar (el sistema se encarga de registrarlo) y devuelve al cliente el nombre y puerto del que será su servidor definitivo para las futuras peticiones.

En caso de que el cliente ya se encuentre registrado en el sistema se devuelve “ERROR” y no se registra.

5.1.2. iniciarSesion(String cliente)

Si el cliente pertenece al servidor al que ha realizado la petición este le devuelve su nombre y su puerto, en caso contrario el servidor pregunta al sistema el nombre y el puerto del servidor en el que el cliente está registrado y se lo devuelve al cliente.

En caso de que el cliente no esté registrado se devuelve “ERROR”.

5.1.3. donar(String cliente, double cantidad)

Si la cantidad a donar es mayor que 0, se incluye la cantidad donada a la cantidad que ya tuviera asociada el cliente anteriormente, se suma la cantidad a la cantidad de la réplica servidor y se incrementa el número de donaciones realizadas en la réplica servidor.

5.1.4. getCantidadCliente(String cliente)

Devuelve la cantidad que ha donado el cliente.

5.1.5. getCantidadReplica()

Devuelve la cantidad que se ha donado en la réplica.

5.1.6. getCantidadSistema()

Se comunica con el sistema para calcular la cantidad total de todo el sistema y se devuelve.

5.1.7. getNumDonacionesReplica()

Devuelve el número de donaciones que se han realizado en la réplica.

5.1.8. getNumDonacionesSistema()

Se comunica con el sistema para calcular el número de donaciones total de todo el sistema y se devuelve.

5.1.9. getNumClientesReplica()

Devuelve el número de clientes que hay actualmente en la réplica.

5.1.10. getNumClientesSistema()

Se comunica con el sistema para calcular el número de clientes que hay en todas las réplicas del sistema y lo devuelve.

5.1.11. getNombreMayorDonacionReplica()

Recorre todos los clientes buscando la mayor donación y devuelve el nombre asociado a la misma.

5.1.12. getNombreMayorDonacionSistema()

Se comunica con el sistema para conocer cuál es el nombre asociados a la mayor donación de todas las réplicas.

5.1.13. getCantidadMayorDonacionReplica()

Recorre todos los clientes buscando la mayor donación y devuelve la cantidad de la misma.

5.1.14. getCantidadMayorDonacionSistema()

Se comunica con el sistema para conocer cuál es la cantidad asociada a la mayor donación de todas las réplicas.

5.1.15. darDeBaja(String cliente)

Elimina al usuario de la réplica y actualiza los datos.

5.2. Peticiones Servidor-Servidor

5.2.1. yaRegistrado(String cliente)

Recorre todas las réplicas del sistema y devuelve true si el cliente ya está registrado en alguna o false si no lo está.

5.2.2. registrar(String cliente)

Recorre todas las réplicas del sistema buscando cual es la que tiene menos clientes registrados, se registra al cliente y devuelve su nombre y puerto de la replica correspondiente.

(cuando se invoca este método ya se ha comprobado que el cliente no existe previamente en el sistema).

5.2.3. iniciarSesion(String cliente)

Recorre todas las réplicas del sistema buscando cual es la que tiene al usuario registrado y devuelve su nombre y su puerto, en el caso de no encontrarlo devuelve ERROR.

5.2.4. getcantidadTotal()

Recorre todas las réplicas del sistema sumando la cantidad donada en cada réplica y devuelve el resultado.

5.2.5. `getNumDonacionesTotal()`

Recorre todas las réplicas del sistema sumando el número de donaciones en cada réplica y devuelve el resultado.

5.2.6. `getNumClientesTotal()`

Recorre todas las réplicas del sistema sumando el número de clientes en cada réplica y devuelve el resultado.

5.2.7. `getNombreMayorDonacionTotal()`

Recorre todas las réplicas del sistema buscando cual es la mayor donación en cada réplica y las va comparando hasta obtener la mayor donación de todo el sistema y devuelve el nombre del cliente asociado a ella.

5.2.8. `getCantidadMayorDonacionTotal()`

Recorre todas las réplicas del sistema buscando cual es la mayor donación en cada réplica y las va comparando hasta obtener la mayor donación de todo el sistema y devuelve la cantidad asociada a ella.

6. Ejemplos de ejecución

La compilación y ejecución tanto del cliente como del servidor se ha realizado con NetBeans. Para ello solo hace falta abrir los proyectos y ejecutar el main de la clase *servidor* tantas veces como servidores se quieran y el main de la clase *cliente* tantas veces como clientes se quieran.

Restricciones a tener en cuenta:

- El primer servidor en iniciarse lleva asociado el puerto 1100 y el nombre “replica1”, por lo que ni ese puerto ni ese nombre podrán ser usados por el resto de servidores.
- El canal de comunicación servidor-servidor lleva asociado el puerto 1099 y el nombre “comunicacionSS”, por lo que ni ese puerto ni ese nombre podrán ser usados por el resto de servidores.
- Los servidores pueden iniciar su ejecución en cualquier momento, es decir no necesariamente deben estar todos iniciados al principio. Pero para mantener la integridad de los datos ningún servidor debe abortar su ejecución.
- Antes de ejecutar el cliente, al menos 1 servidor debe estar activo y será su puerto y su nombre el que se usen para la primera petición del cliente.

Para la ejecución, en el cliente se han manejado las operaciones a partir de un menú en el que se van mostrando las distintas operaciones posibles. La primera petición la realiza al servidor que se inicia primero porque se han incluido los datos de esa réplica por comodidad, pero es sencillo realizar el cambio en el código para que se pueda comunicar al principio con cualquiera.

En los servidores al inicio se pregunta sobre el nombre y el puerto que llevará asociado el servidor y durante la ejecución se muestra en azul las peticiones que le llegan por parte de los

clientes. Además en el primer servidor en iniciarse se muestran en verde las peticiones que han necesitado comunicación con el resto de servidores.

6.1. Ejemplo

A continuación se muestra un supuesto ejemplo de ejecución mostrando el estado del sistema en cada paso.

1. Se inicia el primer servidor.

```
ant -f "D:\3* GII\DSD\Practicas\P3_RMI\Ejercicio\Donaciones\servidor" -Dnb.internal.action.name=run run
init:
Deleting: D:\3* GII\DSD\Practicas\P3_RMI\Ejercicio\Donaciones\servidor\build\build-jar.properties
deps-jar:
Updating property file: D:\3* GII\DSD\Practicas\P3_RMI\Ejercicio\Donaciones\servidor\build\build-jar.properties
compile:
run:
¿Es el primer servidor que se inicia?
0: No
1: Si
--Introduce una opción: 1
=> Servidor replical preparado!
```

Figura 14: Se inicia el primer servidor: Replica 1

2. Se inicia un cliente, se registra y realiza una donación.


```

ant -f "D:\3° GII\DSD\Practicas\P3_RMI\Ejercicio\Donaciones\cliente" -Dnb.internal.action.name=run run
init:
Deleting: D:\3° GII\DSD\Practicas\P3_RMI\Ejercicio\Donaciones\cliente\build\build-jar.properties
deps-jar:
Updating property file: D:\3° GII\DSD\Practicas\P3_RMI\Ejercicio\Donaciones\cliente\build\build-jar.properties
compile:
run:
Buscando al servidor...

Opciones disponibles:
1: Iniciar Sesión
2: Registrarse
3: Salir
--Introduce una opción: 2

----FORMULARIO DE REGISTRO----
Introduce el nombre: Clientel
Registro correcto...

Operaciones disponibles:
1: Donar
2: Darse de baja
3: Cerrar Sesión / Salir
--Introduce una opción: 1

-----DONAR-----
Introduce la cantidad (en decimal con punto): 50.0
La donacion se ha realizado de forma correcta. GRACIAS POR SU APORTACIÓN! :)

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción:

```

Figura 15: Registro y donación de Clientel

```

---replical: Clientes => {Clientel=50.0}

```

Figura 16: Estado del sistema

3. Se inician 3 servidores más.

```

compile:
run:
¿Es el primer servidor que se inicia?
0: No
1: Si
--Introduce una opción: 0
Puerto : 1101
Nombre : replica2

=> Servidor replica2 preparado!

```

Figura 17: Replica 2

```

compile:
run:
¿Es el primer servidor que se inicia?
0: No
1: Si
--Introduce una opción: 0
Puerto : 1102
Nombre : replica3

=> Servidor replica3 preparado!

```

Figura 18: Replica 3

```

compile:
run:
¿Es el primer servidor que se inicia?
0: No
1: Si
--Introduce una opción: 0
Puerto : 1103
Nombre : replica4

=> Servidor replica4 preparado!

```

Figura 19: Replica 4

4. Se registran 8 clientes más y 4 de ellos realizan donaciones.

a) Cliente2.

```

compile:
run:
Buscando al servidor...

Opciones disponibles:
1: Iniciar Sesión
2: Registrarse
3: Salir
--Introduce una opción: 2

----FORMULARIO DE REGISTRO----
Introduce el nombre: Cliente2
Registro correcto...

Operaciones disponibles:
1: Donar
2: Darse de baja
3: Cerrar Sesión / Salir
--Introduce una opción: 1

-----DONAR-----:
Introduce la cantidad (en decimal con punto): 20.50
La donación se ha realizado de forma correcta. GRACIAS POR SU APORTACIÓN! :)

```

Figura 20: Registro y donación Cliente2

```

---replica1: Clientes => {Cliente1=50.0}
---replica2: Clientes => {Cliente2=20.5}
---replica3: Clientes => {}
---replica4: Clientes => {}

```

Figura 21: Estado del sistema

b) Cliente3.

```

Opciones disponibles:
1: Iniciar Sesión
2: Registrarse
3: Salir
--Introduce una opción: 2

----FORMULARIO DE REGISTRO----
Introduce el nombre: Cliente3
Registro correcto...

Operaciones disponibles:
1: Donar
2: Darse de baja
3: Cerrar Sesión / Salir
--Introduce una opción: 1

-----DONAR-----
Introduce la cantidad (en decimal con punto): 100.0
La donación se ha realizado de forma correcta. GRACIAS POR SU APORTACIÓN! :)

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mí
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el número total de donaciones en el sistema
6: Consultar el número total de donaciones en mi servidor concreto
7: Consultar el número de clientes totales del sistema
8: Consultar el número de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 14
La sesión se ha cerrado de forma correcta. HASTA PRONTO! :)

```

Figura 22: Registro y donación Cliente3

```

---replica1: Clientes => {Cliente1=50.0}
---replica2: Clientes => {Cliente2=20.5}
---replica3: Clientes => {Cliente3=100.0}
---replica4: Clientes => {}

```

Figura 23: Estado del sistema

c) Cliente4.

```

Opciones disponibles:
1: Iniciar Sesión
2: Registrarse
3: Salir
--Introduce una opción: 2

----FORMULARIO DE REGISTRO----
Introduce el nombre: Cliente4
Registro correcto...

Operaciones disponibles:
1: Donar
2: Darse de baja
3: Cerrar Sesión / Salir
--Introduce una opción: 3
La sesión se ha cerrado de forma correcta. HASTA PRONTO! :)

```

Figura 24: Registro Cliente4

```

---replica1: Clientes => {Cliente1=50.0}
---replica2: Clientes => {Cliente2=20.5}
---replica3: Clientes => {Cliente3=100.0}
---replica4: Clientes => {Cliente4=0.0}

```

Figura 25: Estado del sistema

d) Cliente5.

```

Opciones disponibles:
1: Iniciar Sesión
2: Registrarse
3: Salir
--Introduce una opción: 2

----FORMULARIO DE REGISTRO----
Introduce el nombre: Cliente5
Registro correcto...

Operaciones disponibles:
1: Donar
2: Darse de baja
3: Cerrar Sesión / Salir
--Introduce una opción: 3
La sesión se ha cerrado de forma correcta. HASTA PRONTO! :)

```

Figura 26: Registro Cliente5

```

---replica1: Clientes => {Cliente5=0.0, Cliente1=50.0}
---replica2: Clientes => {Cliente2=20.5}
---replica3: Clientes => {Cliente3=100.0}
---replica4: Clientes => {Cliente4=0.0}

```

Figura 27: Estado del sistema

e) Cliente6.

```

Opciones disponibles:
1: Iniciar Sesión
2: Registrarse
3: Salir
--Introduce una opción: 2

----FORMULARIO DE REGISTRO----
Introduce el nombre: Cliente6
Registro correcto...

Operaciones disponibles:
1: Donar
2: Darse de baja
3: Cerrar Sesión / Salir
--Introduce una opción: 1

-----DONAR-----:
Introduce la cantidad (en decimal con punto): 60.0
La donación se ha realizado de forma correcta. GRACIAS POR SU APORTACIÓN! :)

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mí
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el número total de donaciones en el sistema
6: Consultar el número total de donaciones en mi servidor concreto
7: Consultar el número de clientes totales del sistema
8: Consultar el número de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 14
La sesión se ha cerrado de forma correcta. HASTA PRONTO! :)

```

Figura 28: Registro y donación Cliente6

```

---replica1: Clientes => {Cliente5=0.0, Cliente1=50.0}
---replica2: Clientes => {Cliente6=60.0, Cliente2=20.5}
---replica3: Clientes => {Cliente3=100.0}
---replica4: Clientes => {Cliente4=0.0}

```

Figura 29: Estado del sistema

f) Cliente7.

```

Opciones disponibles:
1: Iniciar Sesión
2: Registrarse
3: Salir
--Introduce una opción: 2

----FORMULARIO DE REGISTRO----
Introduce el nombre: Cliente7
Registro correcto...

Operaciones disponibles:
1: Donar
2: Darse de baja
3: Cerrar Sesión / Salir
--Introduce una opción: 1

-----DONAR-----:
Introduce la cantidad (en decimal con punto): 150.0
La donacion se ha realizado de forma correcta. GRACIAS POR SU APORTACIÓN! :)

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 14
La sesión se ha cerrado de forma correcta. HASTA PRONTO! :)

```

Figura 30: Registro y donación Cliente7

```

---replica1: Clientes => {Cliente5=0.0, Clientel=50.0}
---replica2: Clientes => {Cliente6=60.0, Cliente2=20.5}
---replica3: Clientes => {Cliente7=150.0, Cliente3=100.0}
---replica4: Clientes => {Cliente4=0.0}

```

Figura 31: Estado del sistema

g) Cliente8 y Cliente9.

```

Opciones disponibles:
1: Iniciar Sesion
2: Registrarse
3: Salir
--Introduce una opción: 2

----FORMULARIO DE REGISTRO----
Introduce el nombre: Cliente8
Registro correcto...

Operaciones disponibles:
1: Donar
2: Darse de baja
3: Cerrar Sesion / Salir
--Introduce una opción: 3
La sesión se ha cerrado de forma correcta. HASTA PRONTO! :)

Opciones disponibles:
1: Iniciar Sesion
2: Registrarse
3: Salir
--Introduce una opción: 2

----FORMULARIO DE REGISTRO----
Introduce el nombre: Cliente9
Registro correcto...

Operaciones disponibles:
1: Donar
2: Darse de baja
3: Cerrar Sesion / Salir
--Introduce una opción: 3
La sesión se ha cerrado de forma correcta. HASTA PRONTO! :)

```

Figura 32: Registro Cliente8 y Cliente9

```

---replica1: Clientes => {Cliente9=0.0, Cliente5=0.0, Clientel=50.0}
---replica2: Clientes => {Cliente6=60.0, Cliente2=20.5}
---replica3: Clientes => {Cliente7=150.0, Cliente3=100.0}
---replica4: Clientes => {Cliente8=0.0, Cliente4=0.0}

```

Figura 33: Estado del sistema

5. Cliente1 realiza las consultas

```

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesion / Salir
--Introduce una opción: 2

-----CONSULTAR EL DINERO TOTAL DONADO POR MI-----:
El dinero total que has donado ha sido 50.0

```

Figura 34:

```

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 3

-----CONSULTAR EL DINERO TOTAL DONADO EN EL SISTEMA-----:
El dinero total donado en el sistema es 380.5

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 4

-----CONSULTAR EL DINERO TOTAL DONADO EN MI SERVIDOR CONCRETO-----:
El dinero total donado en tu replica es 50.0

```

Figura 35:


```

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 5

-----CONSULTAR EL NUMERO TOTAL DE DONACIONES EN EL SISTEMA-----:
El numero total de donaciones en el sistema es 5

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 6

-----CONSULTAR EL NUMERO TOTAL DE DONACIONES EN MI SERVIDOR CONCRETO-----:
El numero total de donaciones en tu servidor concreto es 1

```

Figura 36:

```

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 7

-----CONSULTAR EL NUMERO DE CLIENTES TOTALES DEL SISTEMA-----:
    El numero de clientes totales del sistema es 9

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 8

-----CONSULTAR EL NUMERO DE CLIENTES DE MI SERVIDOR CONCRETO-----:
    El numero de clientes de tu servidor concreto es 3

```

Figura 37:

```

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 9

-----CONSULTAR EL NOMBRE DEL CLIENTE QUE HAYA DONADO MÁS DINERO EN TODO EL SISTEMA-----:
El nombre del cliente que haya donado más dinero en todo el sistema es Cliente7

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 10

-----CONSULTAR EL NOMBRE DEL CLIENTE QUE HAYA DONADO MÁS DINERO EN MI SERVIDOR CONCRETO-----:
El nombre del cliente que haya donado más dinero en esta replica es Clientel

```

Figura 38:

```

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 11

-----CONSULTAR LA MAYOR CANTIDAD DONADA EN TODO EL SISTEMA-----:
La mayor cantidad donada en todo el sistema es 150.0

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el numero total de donaciones en el sistema
6: Consultar el numero total de donaciones en mi servidor concreto
7: Consultar el numero de clientes totales del sistema
8: Consultar el numero de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 12

-----CONSULTAR LA MAYOR CANTIDAD DONADA EN MI SERVIDOR CONCRETO-----:
La mayor cantidad donada en mi servidor concreto 50.0

```

Figura 39:

6. Cliente9 inicia sesión y se da de baja

```

Opciones disponibles:
1: Iniciar Sesión
2: Registrarse
3: Salir
--Introduce una opción: 1

-----INICIO SESIÓN-----
Introduce el nombre: Cliente9
Inicio de sesion correcto...

Operaciones disponibles:
1: Donar
2: Darse de baja
3: Cerrar Sesión / Salir
--Introduce una opción: 2

-----DARSE DE BAJA-----:
La operacion se ha realizado de forma correcta. :)

```

Figura 40:

7. Cliente6 realiza consultas los clientes actuales del sistema, que deben ser 8 porque Cliente9 se ha dado de baja

```

Opciones disponibles:
1: Iniciar Sesión
2: Registrarse
3: Salir
--Introduce una opción: 1

----INICIO SESIÓN----
Introduce el nombre: Cliente6
Inicio de sesión correcto...

Operaciones disponibles:
1: Donar
2: Consultar el dinero total donado por mi
3: Consultar el dinero total donado en el sistema
4: Consultar el dinero total donado en mi servidor concreto
5: Consultar el número total de donaciones en el sistema
6: Consultar el número total de donaciones en mi servidor concreto
7: Consultar el número de clientes totales del sistema
8: Consultar el número de clientes de mi servidor concreto
9: Consultar el nombre del cliente que haya donado más dinero en todo el sistema
10: Consultar el nombre del cliente que haya donado más dinero en mi servidor concreto
11: Consultar la mayor cantidad donada en todo el sistema
12: Consultar la mayor cantidad donada en mi servidor concreto
13: Darse de baja
14: Cerrar Sesión / Salir
--Introduce una opción: 7

-----CONSULTAR EL NUMERO DE CLIENTES TOTALES DEL SISTEMA-----:
El número de clientes totales del sistema es 8

```

Figura 41: