Memoria Donaciones con RMI

En esta práctica se nos pedía crear un servidor de Donaciones que tuviera 2 réplicas comunicadas entre sí y que sirviesen a los usuarios con distintas operaciones. Estas réplicas se deberían "equilibrar" para que ambas tengan más o menos el mismo número de usuarios.

Empecé creando el gestor de donaciones. Este sería una clase `Donaciones` que implemmenta una interfaz `Donaciones_I` en la que se declaran todos los métodos.

La clase `Donaciones` se instanciará dos veces, una por cada réplica.
Esta tendrá los siguientes atributos:

- Una lista con los usuarios de dicha réplica
- Su nombre en los servidores
- El nombre de la otra réplica
- El subtotal donado en esa réplica
- El host del servidor

De esta manera cada réplica conocerá el nombre de la otra y podrá acceder a ella.

Los usuarios serán instancias de una clase `Usuario` que poseerá su nombre, password, numero de donaciones realizadas y cantidad total donada. Tendrá operaciones básicas como los "gets" y "sets" y un método para realizar una donación que aumentará el numero de donaciones y la cantidad total donada.

Volviendo a la clase `Donaciones`, los principales métodos que implementa son:

- `identificarUsuario`: Este método comprueba que el usuario que se le pasa como parametro existe y coincide con la contraseña que se le pasa.

- `intentoRegistrar`: Comprueba que el usuario no esté ya registrado y si no lo está, lo registra en la réplica con menos usuarios.

- `getServidorUsuario`: Devuelve el nombre de la réplica en la que está registrado el usuario que se le pasa por parámetro.

- `donar`: Realiza una donación de parte del usuario que se le pasa por parámetro e incrementa el subtotal de la réplica.

- `getTotal`: Devuelve el total donado a los servidores, para esto consulta cuál es el subtotal de la otra réplica y devuelve la suma de los subtotales.

- `getTotalUsuario`: Devuelve la cantidad que ha donado un usuario concreto.

- `getNumDonaciones`: Devuelve el número de donaciones de un usuario pasado por parámetro.

La clase `Cliente` actuará como programa que utiliza el usuario del sistema. En esta se obtiene inicialmente la Replica1 y se le pregunta al usuario qué operación desea realizar:

- Identificarse: Inicia sesión con una contraseña y usuario concreto. Si tiene éxito, el cliente buscará en qué réplica se había registrado al usuario y a partir de entonces se trabajará en dicha réplica.

- Registrarse: El usuario podrá registrarse en el sistema.

- Salir: Se cerrará el cliente.

Una vez el usuario se haya registrado aparecerá otro menú con otras opciones distintas:

- Donar: El usuario podrá realizar una donación de un importe mayor que 0.

- Consultar total: Si el usuario ya ha realizado al menos una donación podrá consultar cuánto dinero se ha recaudado para la causa.

- Consultar donado personal: El usuario podrá consultar cuánto ha donado él/ella a la causa.

- Número donaciones: Consulta el número de donaciones que ha realizado el usuario.

- Cerrar sesión: El usuario cierra sesión y se le vuelve a mostrar el menú inicial.

Errores que me he encontrado

El único problema que me he encontrado es que la clase `Usuario` me daba un error `java.io.NotSerializableException` busqué por internet y para solucionarlo tenía que hacer que la clase extendiera de la interfaz `Serializable`, esto permite que el objeto de la clase se pueda recuperar cuando se envía a través de la red.