Práctica 3: RMI

Normas Generales

- La práctica se realiza de forma individual.
- La práctica durará 3 sesiones: consultar la fecha de entrega de cada grupo en PRADO.
- La entrega de la práctica estará formada por un fichero .zip con todos los archivos utilizados y una memoria en .pdf explicando la solución (se aconseja usar diagramas y capturas de pantalla mostrando su funcionamiento).
- La detección de copia implica un SUSPENSO en toda la práctica. Se usará el sistema Turnitin para detección de plagios.
- Se valorará la participación en el foro para resolver las dudas de los compañeros.

Parte 1: Implementación de los ejemplos (1 semana). Esta parte es OBLIGATORIA para corregir la segunda parte.

La primera parte consiste en seguir los pasos para implementar y probar los ejemplos del guión. Demostrar que funcionan y explicar qué ocurre y qué diferencias hay entre ellos. Es muy importante que entendáis bien el guión y sigáis los pasos que en él se indican para no tener problemas más tarde. Documentad todo en la memoria a entregar.

Parte 2: Servidor replicado (2 semanas)

La segunda parte consiste en implementar el servidor replicado que se especifica en el guión, con una serie de salvedades.

- Explicar en la memoria qué técnica habéis usado para que los dos servidores cooperen entre sí (descubrimiento, actualización de datos). Podéis usar diagramas para ayudaros a describir el sistema.
- Explicar los métodos del cliente y del servidor.
- Extender la funcionalidad del servidor con cualquier cosa que se os ocurra:
    - Añadir más operaciones al servidor
    - Probar con más de 2 réplicas
    - Probar en ordenadores distintos
    - Aplicar alguno de los algoritmos para exclusión mutua distribuida (basada en relojes lógicos o anillo) que se han visto recientemente en teoría para desarrollar la solución para garantizar consistencia en la variable replicada del total de donaciones recibidas.

Algunas cosas a tener en cuenta:

- Por defecto *rmiregistry* coge las clases que hay en el classpath por defecto de vuestra máquina virtual de java, por lo que si no tocáis nada, tendréis que lanzarlo desde la carpeta donde estén los .class compilados (si usáis javac como en los ejemplos, pues desde el directorio donde habéis compilado como sale en el script de compilación, pero por ejemplo en NetBeans los .class se crean en la carpeta build/classes, tendríais que lanzarlo desde ahí).
- Por esa razón, los ficheros .policy y los argumentos de la VM son ligeramente distintos entre lanzar por LC y NB (como viene al final del guión).
- En Netbeans podéis crear el cliente y el servidor en el mismo proyecto, aunque en la vida real se hace en proyectos separados, que comparten una interfaz en un proyecto aparte.
- En NetBeans deberéis crear varias *configuraciones*: una para el cliente, con sus parámetros de la VM y los argumentos del programa, y una para cada servidor replicado (con sus argumentos específicos al servidor, y los parámetros de la VM comunes).
- Si compilais desde línea de comandos, no olvidéis el “-cp .” <--el punto lo soléis olvidar.
- Escribid bien los argumentos de ejecución, si no encuentra el fichero, o está mal escrito, no os avisa.
- Ojo con los argumentos de los ejemplos del guión, dependiendo del ejemplo los argumentos son distintos!

Rúbrica:

- De 0 a 5 puntos: La práctica no funciona o da errores. La memoria es muy incompleta.
- De 5 a 6 puntos: La arquitectura desarrollada funciona con dos servidores correctamente como se pide en el guión. La memoria es correcta.
- De 6 a 8 puntos: Se han implementado más operaciones y se ha extendido más que la parte básica. La memoria es de más calidad.
- De 8 a 10 puntos: Se han implementado muchas más operaciones y se han probado distintos algoritmos y gestión de más de un servidor replicado. La memoria es muy completa. Incluye diagramas y explicaciones de cómo se han abordado los problemas y errores.