Programación Distribuida y Tiempo Real Facultad de Informática - Universidad Nacional de La Plata

Prof. Fernando G. Tinetti

Practica 2

1) Utilizando como base el programa ejemplo de RMI:

   a.- Analice si RMI es de acceso completamente transparente (*access transparency*, tal como está definido en Coulouris-Dollimore-Kindberg). Justifique.

   b.- Enumere los archivos .class que deberían estar *del lado del cliente* y *del lado del servidor* y que contiene cada uno.

2) Investigue porque con RMI puede generarse el problema de desconocimiento de clases en las JVM e investigue como se resuelve este problema.

3) Implementar con RMI el *mismo* sistema de archivos remoto implementado con RPC en la práctica anterior:

   a.- Defina e implemente con RMI un servidor cuyo funcionamiento permita llevar a cabo las operaciones desde un cliente enunciadas informalmente como (definiciones copiadas aquí de la práctica anterior):

   leer: dado un nombre de archivo, una posición y una cantidad de bytes a leer, retorna 1) la cantidad de bytes del archivo pedida a partir de la posición dada o en caso de haber menos bytes, se retornan los bytes que haya y 2) la cantidad de bytes que efectivamente se retornan leidos.

   escribir: dado un nombre de archivo, una cantidad de bytes determinada, y un buffer a partir del cual están los datos, se escriben los datos en el archivo dado. Si el archivo existe, los datos se agregan al final, si el archivo no existe, se crea y se le escriben los datos. En todos los casos se retorna la cantidad de bytes escritos.

   b.- Implemente un cliente RMI del servidor anterior que copie un archivo del sistema de archivos del servidor en el sistema de archivos local y genere una copia del mismo archivo en el sistema de archivos del servidor. En todos los casos se deben usar las operaciones de lectura y escritura del servidor definidas en el item anterior, sin cambios específicos del servidor para este item en particular. Al finalizar la ejecución del cliente deben quedar tres archivos en total: el original en el lado del servidor, una copia del original en el lado del cliente y una copia en el servidor del archivo original. El comando diff no debe identificar ninguna diferencia entre ningún par de estos tres archivos

5) Investigue si es posible que varias invocaciones remotas estén ejecutándose concurrentemente y si esto es apropiado o no para el servidor de archivos del ejercicio anterior. En caso de que no sea apropiado, analice si es posible proveer una solución (enunciar/describir una solución, no es necesario implementarla). **Nota**: diseñe un experimento con el que se pueda demostrar fehacientemente que dos o más invocaciones remotas se ejecutan concurrentemente o no. Compare este comportamiento con lo que sucede con RPC.

6) Tiempos de respuesta de una invocación:

a.- Diseñe un experimento que muestre el tiempo de respuesta mínimo de una invocación con JAVA RMI. Muestre promedio y desviación estándar de tiempo respuesta.

b.- Investigue los timeouts relacionados con RMI. Como mínimo, verifique si existe un timeout predefinido. Si existe, indique de cuanto es el tiempo y si podría cambiarlo. Si no existe, proponga alguna forma de evitar que el cliente quede esperando indefinidamente.

Entrega de la práctica (individual o en grupos de dos alumnos como máximo):

Se debe entregar un único informe detallando lo realizado para cada ejercicio. Debe tener un formato bien definido identificando materia, trabajo practico y autor/es. Se debe entregar en formato electrónico con tipo de archivo .pdf, en tamaño de hoja A4.

Para cada programa modificado o generado para resolver los ejercicios, debe explicarse el cambio o la implementación realizada. Si bien el programa fuente puede estar comentado, el cambio o la implementación realizada debe explicarse en el texto del informe (no es aceptable “ver código fuente” en el informe).

Se debe entregar en formato electrónico tanto el informe como todo el código fuente usado/desarrollado.