INDICACIONES PRÁCTICA 3

* Nombre de Archivo Java:

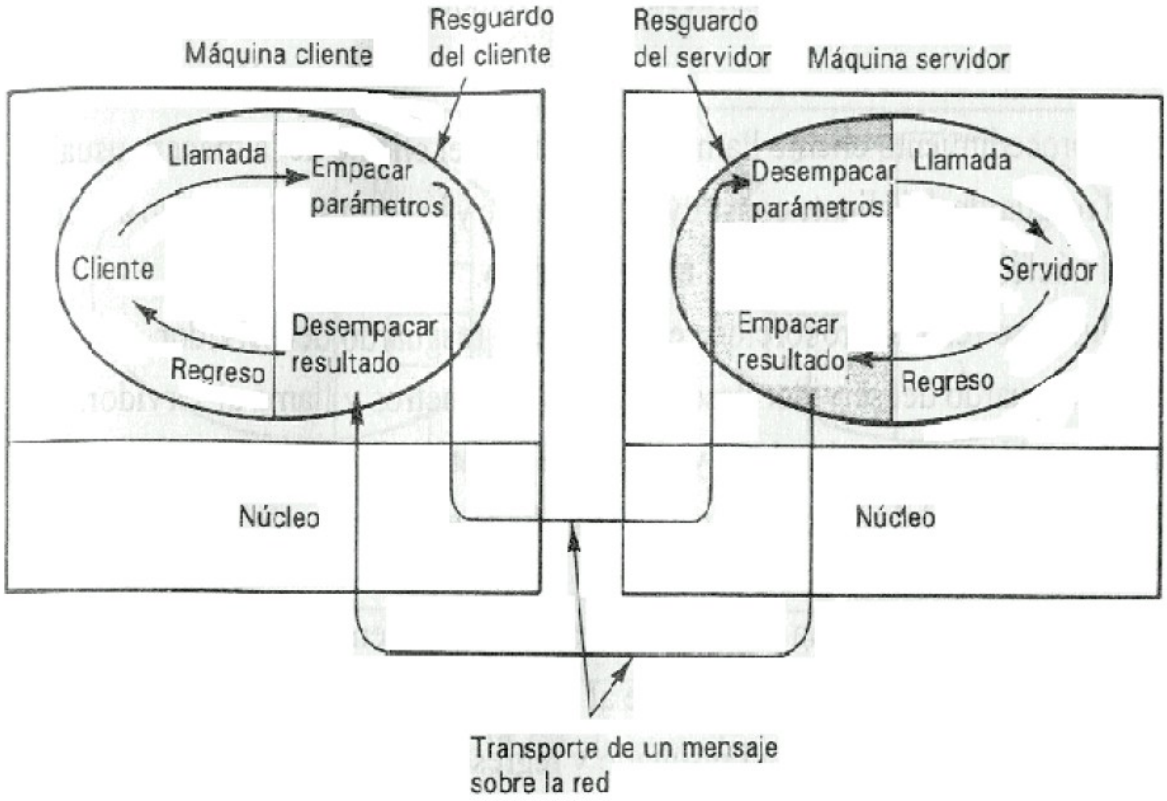
Mandar dentro del archivo comprimido la carpeta sistemaDistribuido

* Ejemplo Nombre Archivo Comprimido:

GutierrezAvilesLuisP3TSOAD0413B.zip

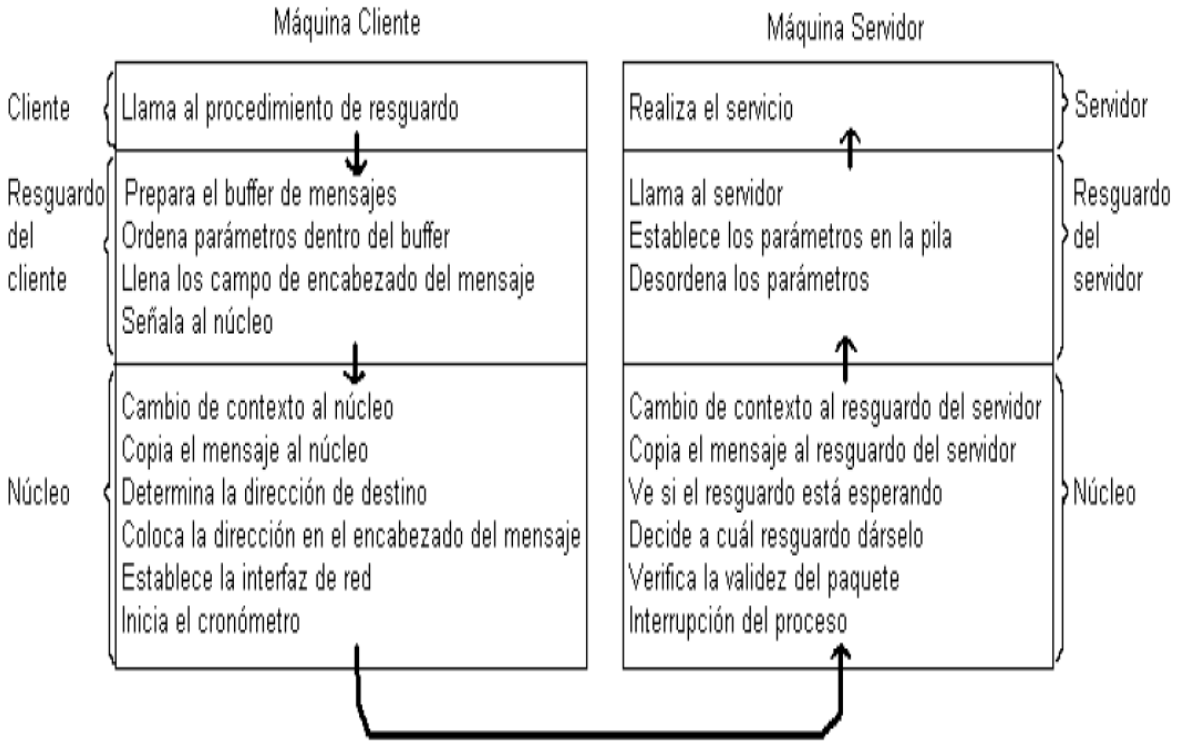
Nota: Al igual que en la práctica 1 y 2, hay que seguir las indicaciones del archivo .pdf y se debe de poner en todos los archivos .java que sean modificados su nombre completo, código y Número de práctica para la cual se hicieron estas correcciones.

La práctica 3 tiene que hacer lo que en teoría se vio como RPC (Remote Procedure Calls). Las siguientes imágenes de la teoría nos ayudarán a recordar un poco el cómo funciona:


Máquina cliente
Resguardo del cliente
Resguardo del servidor
Máquina servidor
Llamada
Empacar parámetros
Cliente
Desempacar resultado
Regreso
Desempacar parámetros
Llamada
Servidor
Empacar resultado
Regreso
Núcleo
Núcleo
Transporte de un mensaje sobre la red

Ruta crítica

Es la serie de instrucciones que se ejecutan con cada RPC



Los archivos a modificar en esta práctica se encuentran dentro del paquete

sistemaDistribuido.sistema.**rpc**.modoUsuario

¿Qué correcciones debo de hacer en procesoCliente.java?

El código que va dentro de este archivo es lo que corresponde hacer al Cliente, es decir simplemente elaborar la solicitud.

Se debe corregir este archivo para que se haga el llamado a 4 procedimientos matemáticos, los que ustedes quieran, tratando de cumplir los requisitos detallados en el archivo pdf de esta práctica.

Es necesario que cambien la ventana del cliente para cambiar el nombre de los Label que se encuentran antes de los campos de texto y poner el código necesario para tomar los datos de estos campos de texto y enviarlos como parámetros de los procedimientos que se deben de encontrar dentro del archivo Libreria.java.

Por ejemplo:

```
int resProm=lib.promedio(cantidad, arreglo);
```

Donde lib es una instancia de LibreriaCliente que hereda de Librería.

¿Cómo funciona esto de las Librerías?

Hay una clase padre que es Librería donde se deben de encontrar las funciones que se van a implementar con el envío de parámetros, pero no se hace ninguna operación, simplemente se toman el o los parámetros que llegan a la función y se ingresan dentro de una pila (Clase Stack en Java), por medio de la función push para luego convocar a una función del mismo nombre pero sin parámetros.

Esta función sin parámetros dentro de Librería, no se implementa simplemente se pone el prototipo de función esperando que las clases que hereden de Librería las implementen.

Ejemplo de Prototipo de función:

```
protected abstract void cuadrado();
```

Después de convocar a la función sin parámetros se espera que en el tope de la pila se encuentre ya la respuesta de la función y se procede a sacarla por medio de la función pop. Una vez obtenida la respuesta se convierte al tipo de dato que se debe de convertir y se regresa como resultado.

OJO: Dentro de stack no se permite meter tipo de datos primitivos por lo que un entero habrá que convertirlo a Integer

antes de meterlo a la pila y al sacar de la pila se sacará un Integer que habrá que convertirlo a entero.

A esta clase Librería se espera que la hereden la LibreriaCliente y la LibreriaServidor, la primera hace lo que le corresponde al Resguardo del Cliente y la segunda hace lo que le corresponde al Servidor. (Ver ruta crítica).

La “Magia” está en que cuando LibreriaCliente herede de Librería e implemente estos métodos sin parámetro no realiza el servicio en sí, sino que a los parámetros que toma de la pila los acomoda en un mensaje y se envían por la red para que sean atendidos por un servidor. En cambio cuando LibreríaServidor hereda de Librería e implementa estos métodos este si realiza el servicio y obtiene una respuesta.

¿Qué se tiene que corregir en LibreriaCliente.java?

Como se explica en el punto anterior en LibreriaCliente se pone el código que le corresponde a lo que hace el resguardo del cliente en la Ruta Crítica, en el pdf también se encuentra un pseudocódigo de lo que se tiene que implementar.

En este archivo pues hay que implementar las funciones sin parámetros que la clase Librería no implementó. Y lo que debemos hacer es lo siguiente:

Sacar parámetros de la pila: Simplemente por medio de pop se saca el parámetro o parámetros, según la función que hayan implementado, convirtiéndolos al tipo de dato requerido.

Preparar buffer de mensajes: Es repetir el código de la practica 1 donde preparan un arreglo de bytes donde se van a colocar los datos necesarios para la solicitud (Origen, destino, Código de operación y datos).

Ordenamiento de parámetros: Se refiere a colocar los datos con los que se va a trabajar dentro del mensaje de la misma forma que en la práctica 1 y 2.

Llenar campos de encabezado: Solamente se llena el espacio dentro del arreglo de bytes o buffer dedicado al código de operación, los datos de origen y destino son llenados por el nucleo y ya esta implementado en la práctica 2.

Se convoca a send, que debe de tener dos parámetros, el primero un número cualquiera (Como el 248 de las prácticas anteriores) que vendría a ser la forma de localizar al servidor pero como aún no tenemos ningún direccionamiento implementado, los datos del servidor se toman de la interfaz del núcleo y por esto no importa este número.

Se convoca receive, que tiene como parámetros el ID del proceso cliente y el arreglo de bytes donde se espera la respuesta, igual que en la práctica 1 y 2.

….. Aquí entra en juego el nucleo que se encarga de atender las solicitudes de send y receive, no se debe de hacer ningún cambio en Micronucleo.java debe de funcionar de la forma en que esta implementado para la práctica 2.

Una vez que el nucleo despierta al proceso Cliente pasa lo siguiente:

Desempacar respuesta: Se toma la respuesta que viene dentro del arreglo de bytes que se recibió de la red

Restauración de parámetros out ó in-out: Se refiere a que en los sistemas operativos distribuidos se trata de evitar los más posible los parámetros por referencia y se utiliza el método copia/restauración, en este paso es copiar los valores cambiados o enviados por el servidor a las variables originales.

Regresar valor izquierdo: Es regresar al cliente el resultado de la operación.

¿Qué se tiene que corregir en ProcesoServidor.java?

Según la ruta critica después de entrar a escena el resguardo del cliente hace el señalamiento al núcleo (núcleo del cliente) que se encarga de enviar el mensaje al núcleo remoto (núcleo servidor) y este le entrega el mensaje al proceso que está esperando la solicitud que en esta caso sería el resguardo del servidor.

El código correspondiente a los núcleos ya esta implementado en la práctica 2 por lo que en esta práctica no se hacen modificaciones a este respecto. Entonces lo siguiente a

modificar en la práctica es lo correspondiente al resguardo del servidor que debe de implementarse en el archivo ProcesoServidor.java.

Aquí hay que implementar código muy parecido al de la práctica 1, donde se convoca a receive en espera de una solicitud, al llegar la solicitud se toma el código de operación de su espacio correspondiente en el arreglo de bytes y según sea este código de operación es la función matemática a ejecutar.

En cada caso habría que sacar los parámetros del mensaje arribado de la red según fueron acomodados por el resguardo del cliente y con esos parámetros mandar a llamar a la función en cuestión de igual forma como lo hizo el Proceso Cliente, es decir así:

```
int resProm=ls.promedio(cantidad, arreglo);
```

En esta caso ls es una instancia de la clase LibreriaServidor que también hereda de Librería, por esto esta función lo que hace es tomar los parámetros y meterlos en una pila y llamar al mismo método pero sin parámetros. -------- Aquí según la ruta crítica continúa lo que hace el servidor, pero eso lo veremos en el siguiente punto.

Una vez que se obtiene la respuesta se tiene que empaquetar dentro del mensaje y convoca a send poniendo como parámetros la ID del proceso que viene dentro del mensaje solicitud como origen y el arreglo de bytes con la respuesta empaquetada (Exactamente igual que en la práctica 2).

¿Qué se tiene que corregir en LibreriaServidor.java?

El código que aquí se implementa es el correspondiente al servidor, y es implementar las funciones sin parámetros que se heredan de la clase Librería, al igual como se hizo en el

archivo LibreriaCliente.java, es decir se sacan los parámetros de la pila pero en vez de empaquetarlos, ahora si se hace la operación solicitada y se ingresa el resultado a la pila.

De esta forma el resguardo del servidor saca el resultado de la pila, lo empaqueta en un arreglo de bytes, señala al núcleo del servidor quien lo envía al núcleo del cliente, quien copia el arreglo de bytes al espacio del proceso que está esperando respuesta, es este caso el resguardo del cliente quien desempaqueta el resultado del mensaje, y lo ingresa en la pila que después es sacado por el proceso de la clase Librería que le entrega el resultado al proceso cliente que no se dio cuenta que su solicitud pudo ser realizada en otra máquina y con esto concluye la práctica 3.

Criterios de Evaluación

- Cumplir con las “Reglas de Operación y Evaluación del Taller de Sistemas Operativos Avanzados”.
- Fecha de asignación: 23 de Octubre del 2013
- Fecha planeación de entrega: 30 de Octubre del 2013
- Fecha límite de entrega (Sólo 55% del total): 20 de Noviembre del 2013
- Observación: _Ninguna_____
- Calificación en base a cobertura de requerimientos y fecha de entrega en horas clase