INDICACIONES PRÁCTICA 4

* Nombre de Archivo Java:

Mandar dentro del archivo comprimido la carpeta sistemaDistribuido

* Ejemplo Nombre Archivo Comprimido:

GutierrezAvilesLuisP4TSOAD0413B.zip

Nota: Al igual que en la práctica 1,2 y 3, 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.

El objetivo de la práctica 4 es lograr la conexión dinámica de la que se ha hablado en la teoría en al apartado RPC, y se refiere a la forma de direccionamiento por medio del cual se logra la búsqueda de un servidor por medio del cliente.

El programa conector es un elemento centralizado que registra a todos los servidores para que los clientes le pregunten a él si hay alguien disponible para darle servicio.

En la práctica 4 simplemente es implementar tres operaciones o funciones por las cuales da servicio el programa conector estas son las que se describen en la siguiente tabla:

Llamada	Entrada	Salida
Registro	Nombre, versión, asa,	identificación única
De registro	Nombre, versión, identificación única	
Búsqueda	Nombre, versión	Asa, identificador único

Figura 2-23. La interfaz del conector

¿Cómo implemento el método Registro?

Para esta práctica vamos a estar trabajando con el mismo paquete de la practica 3, es decir sistemaDistribuido.sistema.rpc.modoUsuario y además con el paquete sistemaDistribuido.sistema.rpc.modoMonitor que contiene el archivo RPC.java que es donde se encuentran los métodos a los que van a convocar los clientes y servidores.

En ProcesoServidor.java debemos hacer una llamada al método registro al inicio del run para que antes de que el resguardo del servidor convoque a receive en espera de mensajes se registre en el programa conector y que de esta forma se ubicado por los clientes.

El Programa Conector es lanzado desde la ventana del micro núcleo por lo que los resguardos del cliente y del servidor que son quienes invocan los métodos del programa conector deben de hacerlo mediante los métodos que se encuentran dentro de RPC.java que ya se encuentran declarados sólo hay que rellenarlos.

El método de RPC.java para el registro al programa conector es exportarInterfaz, asi pues según dice la tabla debemos mandarle 3 parámetros que son: Nombre, Versión y asa , donde el nombre y la versión son inventados por ustedes sólo

asegúrense que los clientes utilicen estos mismos nombres y versiones para la búsqueda.

El asa es un ParMaquinaProceso, es decir, una clase que contiene los datos de la IP del proceso que convoca el método registro y el número de ID del mismo proceso, estos datos se obtienen de forma similar que en prácticas pasadas.

Ya con estos datos hacemos la llamada de la siguiente forma:

```
identificacionUnica=RPC.exportarInterfaz(SERVIDOR,VERSION,asa);
```

El cual nos debe de regresar un número con el que quedo registrado en la Hashtable del Programa Conector. Es importante que des comenten la línea que dice para práctica 4 y que importa el paquete donde se encuentra RPC.java.

Nos brincamos a RPC.java

Hay que corregir el método exportarInterfaz, y lo único que se hace es convocar el método registro del Programa Conector que ya esta instanciado en la variable conector y el valor que nos regrese este método es el valor que a su vez regresara el método exportarInterfaz.

Nos brincamos a ProgramaConector.java

No existe el método registro por lo cual debemos crearlo con su tipo de dato de retorno y sus 3 parámetros, ya hay instanciada una Hashtable llamada conexiones que es donde vamos a registrar estos datos, puede que sea necesario que tengan que especificar qué tipos de datos se ingresan en la Hashtable. Algo asi:

```
private Hashtable<Integer,Object> conexiones;
```

Se recomienda crear una Clase que contenga todos los datos del servidor para ser guardada como Object en la Hashtable, la key es la identificación única que la obtenemos de la interfaz gráfica a la hora de escribir los datos en la ventana de la siguiente manera:

```
int
idUnica=desplegante.agregarServidor(servidor,version,asa.dameIP(),Integer
.toString(asa.dameID()));
```

desplegante es la que muestra en la ventana los servidores registrados y el método agregarServidor nos regresa un entero que utilizaremos como key para el registro de los datos en la Hashtable, este mismo entero de idUnica será el valor de retorno de la función.

Y listo se ha registrado el servidor, el resguardo del servidor (ProcesoServidor.java) recibe esta idUnica y la guardará para el momento de deregistrar al servidor.

¿Cómo implemento el método Deregistro?

Este método es también convocado por el *Resguardo* después del ciclo while(continuar), el método de RPC.java que nos servirá en esta ocasión es deregistrarInterfaz que tiene como valor de retorno un booleano y que según la tabla hay que enviarle tres parámetros que son: Nombre, Versión e identificación Única, todos estos datos ya son conocidos por el resguardo del servidor por lo que simplemente se usan.

Nos brincamos a RPC.java

Aquí es necesario rellenar el método deregistrar interfaz, primeramente de la misma manera como se hizo en el método registro , es decir se agarran los parámetros y se convoca al método deregistro del conector.

Como lo que nos regresa es un booleano que nos indicará si se logro o no el deregistro en este método simplemente se muestra en la ventana el resultado de el mètodo Nucleo.imprimeln(“Algo”);

Nos brincamos a ProgramaConector.java

El método deregistro no está implementado por lo cual deberíamos crearlo poniendo el valor de retorno y los parámetros que se requieren.

Con la Idunica que recibimos de parámetro hacemos la búsqueda en la Hashtable y nos debería regresar un Objeto/Clase que contiene los datos de Nombre, Version y asa del servidor, para asegurarnos de que si estamos deregistrando el servidor solicitado tenemos que validar que el nombre del servidor de lo sacado de la Hashtable coincida con el nombre recibido como parámetro, así como la versión, si ambas cosas coinciden procedemos a la eliminación del registro de la Hashtable y de una vez también lo borramos de la ventana por medio del desplegante:

```
desplegante.removerServidor(idUnica);
```

Y como valor de retorno ponemos un true, ya que si se logro el deregistro, si algunas de las validaciones falla o no se encuentra nada en la Hashtable con la idUnica recibida entonces se regresa false.

Y listo, llevamos dos de tres.

¿Cómo implemento el método Búsqueda?

Este método es convocado por el resguardo del Cliente (LibreriaCliente.java), en cada una de las operaciones matemáticas que realizamos en la práctica 3, ya que cada operación se maneja como un RPC distinto.

En vez de poner una asa con valor 0, ese valor lo obtendremos de lo que nos mandé el programa conector.

El método de RPC.java a convocar es importarInterfaz, que según la tabla debe de tener como parámetros el nombre de Servidor y la versión (Los mismos que se utilizaron el resguardo de servidor), y como valor de retorno pondremos un entero que será el número que pondremos en la llamada a send.

Nos brincamos a RPC.java

De nueva cuenta el método importarInterfaz esta vacio, por lo cual deberemos de llenarlo convocando al método búsqueda del programa conector con los mismos parámetros que se reciben pero como valor izquierdo recibiremos una ParMaquinaProceso.

En caso de que después de convocar el método búsqueda la instancia de la clase ParMaquinaProceso sea null entonces se supone que no se encontró ningún servidor y se procede a informale al cliente, es libre como lo implementen, pero se sugiere el uso de números negativos.

En caso de que no sea nulo, este método de importarInterfaz debería registrar el ParMaquinaProceso obtenido en la TablaEmisión, aquella que se creó en la práctica 2 dentro del archivo Micronucleo.Java y que nos regresaba el valor de la key con la que fue registrado y que será el valor de retorno de importarInterfaz.

Aquí se las tienen que ingeniar para crear un método en MicroNucleo.Java y que sea posible ser convocado desde RPC.java para hacer este registro en la TablaEmision.

Aquí nos brincamos a ProgramaConector.Java

Hay que implementar el método búsqueda que regrese una ParMaquinaProceso y que reciba Nombre del servidor y versión.

Como en las Hashtable obtenemos los elementos dentro de ella por medio de la key, que en este caso no tenemos no queda más que por medio de Enumeration analicemos uno a uno los elementos dentro de la Hashtable comparando primero el nombre del servidor con el parámetro recibido y posteriormente la versión, si ambos coinciden regresamos el asa (ParMaquinaProceso del elemento en cuestión),

Si se acaban los elementos y ninguno coincidió en nombre y versión regresamos null y listo hemos terminado la práctica 4.

Criterios de Evaluación

- Cumplir con las “Reglas de Operación y Evaluación del Taller de Sistemas Operativos Avanzados”.
- Fecha de asignación: 30 de Octubre del 2013
- Fecha planeación de entrega: 13 de Noviembre del 2013
- Fecha límite de entrega (Sólo 55% del total): 04 de Diciembre del 2013
- Observación: _Ninguna_____
- Calificación en base a cobertura de requerimientos y fecha de entrega en horas clase