2DA PARTE CONCURRENCIA

PREGUNTAS DE V O F

1- Los recursos reutilizables en serie son aprovechados con ventaja por el modelo de servidor multihilo

-Falso. Los servidores pueden ser secuenciales o en serie como podrían ser la cola de impresión de una impresora.

2- En RMI un método invocado sobre un objeto remoto que devuelve un objeto remoto, devuelve en realidad una referencia al objeto y no una copia serializada

-Verdadero.

3- Una interfaz es un conjunto bien definido de operaciones ofrecidas como los métodos en el caso de objetos distribuidos

-Verdadero

4- El registro RMI no puede ser invocado más de una vez en la misma maquina

-Falso. Si lo lanzamos como servicio" RMI registry", siempre intenta lanzarse en el mismo puerto (en el 1099) entonces cuando lo vuelvas a lanzar, ya estará lanzado. Pero desde programación con "Create Registry " indicamos el puerto en el que queremos lanzar el registro.

5- Los algoritmos de Cristian y Berkeley son algoritmos de sincronización que se basan en establecer una misma referencia temporal en todos los equipos del sistema distribuido

-Verdadera

6- La clase "Server Sockets" debe recibir el constructor como parámetro la IP del servidor que establece el Socket

-Falso. El que necesita la IP es el cliente no el servidor. El servidor lo que necesita es el puerto.

PREGUNTAS DE RESPUESTA CORTA

1-Que es serializar un objeto?

-Convertir el mismo a una cadena de bytes

2-En relación a los relojes de los ordenadores, ¿Qué hay que hacer para trabajar de manera distribuida?

-Sincronizarlos

3-Cual es el patrón de comunicación clásico a emplear para un grupo de robots coordinados?

-Difusión.

EN RELACIÓN A RMI

4- Cual es el patrón sintáctico de la URL pasada como parámetro a “naming.lookup”?

-El formato sintactico es el siguiente:

(rmi://ip/nombre_del_objeto)

-Un ejemplo podría ser:

(rmi://localhost/Hola_str)

5- Que es exactamente lo que se transfiere cuando un objeto se pasa por referencia?

-Un puntero al objeto.

6-Que métodos deben incluir obligatoriamente el tratamiento de “Remote exception”?

-Todos los métodos del objeto remoto

EN RELACIÓN A SISTEMAS DISTRIBUIDOS

7- Caracteristica básica y fundamental que dispone los sistemas no distribuidos y que no pueden aprovechar de ninguna manera los sistemas distribuidos

-La memoria compartida

8- Desafíos a resolver de los sistemas distribuidos en relación a la seguridad

-Integridad, confidencialidad y disponibilidad.

9- Un par de ejemplos que usan el patrón de difusión escucha en sistemas distribuidos

- (Grupo de amigos) WhatsApp, (Suscriptores) listas de correos, (Grupo Servidor) redundancia o reparto de carga.

10- A que hacen referencia las siglas RPC?

-Remote
-Procedure
-Calls

11- Dos ejemplos de programación móvil en sistemas distribuidos donde el código se transfiere totalmente al cliente para su ejecución. (Seguramente no salga)

-JavaScript o PHP

12- Dada la existencia de RCC porque algunos programadores decidieron crear RMI

-Cuando aparecieron los lenguajes de programación orientados a objetos como por ejemplo Java, decidieron ir un paso más para poder convertir esos objetos en remotos.

13- Se desarrolla un código que entre otras cosas tiene que actualizar usando en base a un movimiento de entrada o salida y también tiene que consultar el saldo de una cuenta. Programado bajo java y utilizando hilos, conteste debidamente a las siguientes preguntas:

14.1- Si hay un método para actualizar y otro para consultar, explique si alguno de los dos, debería ser “ Sincronized ” y porque.

-Ambos métodos tienen que ser “ Sincronized ” para poder sincronizar las mismas variables. Si no hiciéramos esto, se podría producir “una condición de carrera” donde la variable podía quedarse en estado de inconsistencia.

14.2- Indique la misma línea los nombres de los métodos de “thread” a invocar por tal de actuar sobre los hilos según las operaciones que se den a conocer, para dormir a uno y a todos y para despertar a uno y a todos.

- Para bloquear un método utilizamos “Wait()“
-Para despertar a un método utilizamos “Notify()”

PREGUNTAS DE CODIGO

```
A1 interface HolaRmiIextendsA2{…}

public class HolaMonRmi() B1 UnicastRemoteObject implements B2{

        public class HolaMonRmiS{

                public static void main(String [] args){

                        try{

                        HolaMonRmiO C1 = new HolaMonRmiO();
                        Naming.rebind(C2, SaludaRemot);

                        }

                }

        }

        public class HolaMonRmiC{

                public static void main(String [] args){


                        try{

                        HolaMonRmiI hm = (D1) D2(UrlRemot + “Salida”);


                        }

                }

        }
```

}

-A1: public

-A2: remote

-B1: extends

-B2: HolaMonRmiI

-C1: saludaRemote

-C2: “saluda” (Es de tipo String)

-D1: HolaMonRmiI

-D2: naming.lookup()