Sistemas distribuidos
Parcial 2

Descripción del sistema a desarrollar

Se debe implementar un programa mediante el modelo de invocación a métodos remotos y servidores concurrentes, que permita a los clientes comprar hamburguesas pequeñas, medianas y grandes, y asignarle una cantidad de ingredientes extra. El cliente debe enviar un objeto DTO al servidor de pedidos, el cual genera una factura con el costo parcial de la hamburguesa y costo total del pedido. La factura generada debe ser enviada a un servidor concurrente de facturas el cual esta implementado con sockecs. Cuando llega una factura se debe notificar automáticamente a un conjunto de cocineros, los cuales previamente se deben registrar. La notificación al cocinero debe mostrarse mediante una GUI.

Una factura se ve de esta manera

===== Listado de hamburguesas compradas =====

Hamburguesa no: 1
Identificador: mi hamburguesa favorita
Tipo: pequeño
Cantidad de ingredientes extra: 2
Costo: $7000

Hamburguesa no: 2
Identificador: extreme burguer
Tipo: grande
Cantidad de ingredientes extra: 5
Costo: $19000

Hamburguesa no: 3
Identificador: solo para mí
Tipo: pequeño
Cantidad de ingredientes extra: 7
Costo: $12000

Costo con IVA del pedido: $ 39750

El valor de cada tipo de hamburguesa y de un ingrediente extra puede ser quemado en el servidor de pedidos.

Preguntas a responder

a) (v 1.5) Ejecute el sistema, compruebe que las 4 funcionalidades han sido desarrolladas correctamente y explique brevemente cada una de las clases e interfaces que constituyen el sistema distribuido.

b) (v 1.0) Explicar los 14 pasos que permiten el funcionamiento del sistema distribuido al realizar los Callback. Durante la explicación debe señalar las líneas de código donde se encuentra implementado el paso. Debe basarse en la explicación de los 14 pasos que constituyen el chat grupal.

c) (v 0.5) Justifique en que líneas de código se pueden ver reflejadas operaciones bloqueantes y no bloqueantes desde la perspectiva de la interoperabilidad del lado del cliente y servidor, y cuantas y cuales sesiones se generan al ejecutar la aplicación. Considere los hilos que se generan.

d) (v 1.0) Realice el diagrama de nodos del sistema implementado y la arquitectura solo de java RMI, considerando que cada cliente de notificaciones antes de comenzar a recibir notificaciones debe registrarse e iniciar sesión en un servidor de usuarios.

e) (v 1.0) Explique mediante un gráfico una forma de controlar los objetos remotos huérfanos en el servidor de pedidos.

Mostrar notificación mediante una GUI

Para mostrar la notificación mediante una GUI, debe tener en cuenta el siguiente código donde se muestra la implementación de la interface notificarClienteNotificacionesCallbackInt.

```
Public class notificarClienteNotificacionesCallbackImpl extends UnicastRemoteObject implements notificarClienteNotificacionesCallbackInt
{
        Private GUICliente objGUI;

        Public notificarClienteNotificacionesCallbackImpl (GUICliente objGUI) thows RemoteException
        {
                super();
                this.objGUI=objGUI;
        }

        Public void notificar (IndicadoresDTO objIndicadores)
        {
                this.ObjGUI.mostrarNotificacion(objIndicadores);
        }
}
```

Sustentación

Para la sustentación del segundo parcial debe seguir las siguientes instrucciones:

- Los dos estudiantes deben participar en la respuesta de cada punto
- Debe registrarse en un horario asignado para la sustentación
- Debe sustentar el código fuente de la aplicación, el funcionamiento de la aplicación y la respuesta a las preguntas en un tiempo máximo de 20 minutos.
- En la presentación puede mostrar los diagramas creados, partes de código que realizan funciones por ejemplo registrar el objeto remoto en el NS, y responder a preguntas teóricas.
- Al momento de responder a cada punto del parcial sea ordenado y si es posible coloque un slide que indique cual pregunta se va a responder.
- Al iniciar la presentación agregue una diapositiva inicial que contenga el título del trabajo a desarrollar, el autor, docente, asignatura y programa.