**Sistema de Gestión Empresarial y Operativa de una Compañía Transportadora**

**Caso 1**

**Manejo de la concurrencia**

**Carlos Felipe Agudelo Ospina**

**cf.agudelo12**

**201328150**

**Sergio Yodeb Velásquez Yepes**

**sy.velasquez10**

**201315851**

**Infraestructura Computacional**

**Universidad de Los Andes**

**Bogotá D.C.**

**2015-2**

**Organización del programa**

Se tiene un programa con cuatro clases:

* Buffer: Es la clase principal del programa. Desde esta se ejecuta el main y además se crean tanto los threads que modelan a los clientes como los threads que modelan los servidores. Esta clase tiene el buffer de mensajes, un atributo del tamaño del buffer, un contador de la cantidad de mensajes dentro del programa y la cantidad de clientes pendientes de atender.
* Servidor: Extiende de la clase thread. Modela los servidores que han de leer los mensajes de los clientes y responder a estos. Cuenta con una referencia al buffer.
* Cliente: Extiende de la clase thread. Modela los clientes que enviarán mensajes al buffer. Cuenta con referencia al buffer y la cantidad de mensajes que deben enviar.
* Mensaje: Representa un mensaje enviado por un cliente al buffer.

**Funcionamiento del programa**

El main del buffer se encarga de leer el tamaño que tendrá el buffer, la cantidad de servidores con los que contará el programa, la cantidad de clientes que realizarán consultas y la cantidad de consultas que harán estos a partir de un archivo de propiedades. Se encarga además de crear la cantidad de threads de clientes y de servidores correspondientes y ejecutarlos a todos.

El run del cliente se encarga de crear iterativamente los mensajes del cliente, enviarlos al buffer y dormir al thread de cliente dentro de la cola del mensaje creado. Una vez este run termine de iterar sobre todos los mensajes a enviar, el cliente reporta al buffer que ya fue atendido y se puede disminuir la cantidad de clientes pendientes.

El run del servidor se encarga de hacer que el servidor se ejecute mientras que hayan clientes pendientes de atención y dentro de este se trata de leer un mensaje del buffer; si el mensaje se pudo leer se responde y se pasa a la siguiente iteración, si no pudo leerse un mensaje se pasa a la siguiente iteración. Después de cada iteración el servidor se encarga de ceder su procesador para que otros servidores puedan ejecutar (yield()).

El envío de mensajes por parte de los clientes y la lectura de los mensajes por parte de los servidores se realizan a través del buffer. El envío de mensajes funciona bajo la siguiente algorítmica: El cliente envía el mensaje al servidor; si el buffer está lleno el cliente se duerme sobre el buffer, si no se agrega el mensaje al buffer. La lectura de mensajes funciona de la siguiente forma: Si no hay mensajes dentro del buffer se retorna null y el servidor vuelve a intentar leer después de ceder su procesador; si hay mensajes se retira el primero, se notifica a la cola del buffer para despertar a un posible cliente dormido esperando por espacio en el buffer y se retorna el mensaje para que el servidor pueda responderlo.

Dentro del mensaje se tienen dos métodos importantes para la lógica del programa:

* Dormir(): El método utilizado para dormir el thread de ejecución del cliente.
* Responder(): El método encargado de hacer que un servidor dé el mensaje de respondido y además despierte al cliente dormido dentro del mensaje para que pueda continuar enviando mensajes.