Juan Nicolás Galvis Ortiz

David Felipe Sabogal

**Caso 1**

1. **Diseño**

De acuerdo a las solicitudes realizadas en el enunciado se ha decidido la realización de un sistema de manejo de threads con las siguientes clases:

* NovaSoftFinancieroOnline: Clase principal, la cual controla la gestión realizada en los clientes y en el servidor.
* Cliente: Clase que representa a los usuarios quienes están encargados de escribir los mensajes. Este genera los mensajes y los almacena en el buffer.
* Servidor: Clase encargada de leer y/o atender los mensajes enviados por el cliente.
* Buffer: Clase dedicada a permitir el traspaso de mensajes entre un usuario y el servidor. Este recibe un mensaje del cliente, lo almacena para que posteriormente sea recogido por el servidor. Como restricción, el Buffer solo puede almacenar a un mensaje en espera, sino no permite enviar el mensaje.
* Mensaje: Objeto que es enviado por el cliente al servidor a través del buffer.

Por motivos prácticos se utilizarán valores preestablecidos, para probar el correcto funcionamiento del programa, en el número de clientes, servidores y los mensajes a enviar.

1. **Funcionamiento**

Para este caso se usaron threads de sincronización de eventos por medio de monitores, los cuales ayudaban a que se realizara de manera concurrente este proceso. Todo esto se puede evidenciar en los siguientes pasos:

1. Se carga el archivo de propiedades, para crear los clientes, los servidores y volver especifico el buffer
2. Se genera un mensaje desde el cliente y de manera inmediata se envía al buffer.
3. El buffer intentara almacenar el mensaje.
   1. Si no es posible, se duerme antes de entrar al buffer ya que solo puede recibir un mensaje en cola
   2. Si lo logra, el buffer verifica con el servidor si debe de ponerle wait o no al mensaje.
4. El servidor lee los mensajes y recién lo haga despierta al que le sigue en la cola, si no hay ningún mensaje en cola el servidor permanecerá en espera activa