Andrés Felipe Losada 201631453  
Daniel Rengifo 201533922  
Santiago Rangel 201632011

Caso 1

Para solucionar el problema propuesto decidimos implementar cuatro clases: Cliente, Servidor, Buffer, y Mensaje. Decidimos crear estas cuatro clases porque el modelo Productor-Consumidor tiene Cliente, Servidor, y Buffer, y porque el mensaje tiene que conocer al cliente que lo tiene, es una relación bidireccional.

La clase cliente tiene como atributos una lista de mensajes (ArrayList) llamada mensajes, y una referencia al buffer, los getters-setters respectivos, el método run y un método extra llamado almacenar. También tiene un método constructor que recibe por parámetro el buffer y la capacidad del cliente. El método run llama al método almacenar con cada mensaje que tiene el cliente. Luego, almacenar revisa si el buffer tiene espacio para almacenar el mensaje. Si no tiene espacio el cliente entra en una espera activa que después de cada intento cede el procesador. Si el buffer tiene espacio, se llama al método almacenar del buffer, pasando por parámetro el mensaje a ser almacenado. Después de almacenarlo el cliente entra en una espera pasiva, vista con el uso del método wait.

La clase servidor tiene como atributos, un mensaje (el mensaje que está procesando), una referencia al buffer, el método constructor (le entra por parámetro el buffer e inicializa el mensaje en null), el método run y un método extra llamado procesarMensaje. El método run revisa si el buffer tiene espacio, o si hay clientes, si el buffer no tiene espacio entra en una espera activa y en cada iteración cede el procesador. Si el buffer tiene espacio se llama al método pedir del buffer y el mensaje dado se pasa como parámetro al método procesarMensaje. Este método le suma uno al mensaje, cambia la variable finalización que está en el buffer y notifica a un cliente que esté esperando. La variable finalización es importante para entender si se procesan todos los mensajes.

La clase buffer tiene como atributos: el tamaño del buffer, una lista de mensajes, el número de clientes, el número de servidores, y una variable llamada finalización. Esta variable se cambia cada vez que el servidor lee un mensaje y se usa para comparar los mensajes totales con los leidos, si son iguales esto implica que el programa ha terminado. Esta clase tiene los métodos: constructor (le entra por parámetro el tamaño, el número de clientes y el número de servidores), almacenar, y pedir. El método almacenar agrega el mensaje a la lista de mensajes almacenados y le resta a la variable tamaño del buffer, esta variable representa el espacio disponible del buffer. El método pedir coge el primer elemento de la lista, lo elimina, le suma uno a la varible tamaño y retorna el mensaje. Esta clase también tiene el método main, este método corre el programa. La información para inicializar cada objeto está en un archivo llamado datos.txt que tiene, en la primera línea, el tamaño del buffer, el número de clientes, y el número de servidores. Luego cada línea representa el número de mensajes que tiene cada cliente, cada línea es un cliente diferente.

La clase mensaje solamente tiene dos atributos y sus respectivos getters-settters. Los atributos son: un entero llamado mensaje, y una referencia al cliente que tiene este mensaje. El constructor recibe por parámetro ambos atributos y los iniclializa.

Para revisar que la aplicación funcione tenemos varios impresiones en consola, cada una se imprime justo después que se toma acción descrita en ella. Al finalizar el programa se imprime una línea que es: “Al final el número de clientes es 0”. Esta línea sirve representa que todos los clientes almacenaron sus mensajes y que estos mensajes fueron procesados por los servidores. Esto se demuestra comparando la variable de buffer, finalización, con el número total de mensajes, si ambos son iguales implica que se procesaron todos los mensajes.