Funcionamiento

Cada cliente genera mensajes y lo envía al buffer. El número de clientes y el número de mensajes que envían deben ser configurables. Para cada mensaje, el thread cliente debe generar un objeto de tipo Mensaje e inicializarlo, después lo envía. Para cada servidor hay un thread, todos estos threads deben terminar para cuando ya no hayan más clientes. Los threads servidores estarán continuamente solicitando mensajes al buffer y respondiendo las respectivas consultas (responder consiste en incrementar el valor del mensaje y avisarle al cliente que puede continuar desde el mismo objeto mensaje). El número de servidores también debe ser configurable. El número de clientes, el número de servidores, el número de consultas de cada uno de los clientes y el tamaño del buffer deben estar en un archivo, el cual será procesado por el main del programa.

Diseño

En el sistema tendremos: clientes, servidores, buffer y mensajes. Los clientes y servidores son los antes descritos. En cuanto al buffer, es un sitio donde los clientes almacenan los mensajes para que sean recogidos por los servidores; este buffer debe tener una cierta capacidad limitada, y funcionar en esquema productor-consumidor. Por su parte, los mensajes son objetos con la consulta que hace el cliente, y donde el servidor deja la respuesta. El funcionamiento es el siguiente: un cliente genera un mensaje, e intenta depositarlo en el buffer; si no es posible queda en espera activa intentando guardar mensajes, si no lo logra, cede procesador con el método yield(). Una vez depositado el mensaje, el cliente debe quedar en espera pasiva de la respuesta del servidor; pero esta vez la espera sobre el mismo objeto mensaje. Cada servidor, por su parte, está continuamente intentando retirar mensajes del buffer; si está vacío, se queda en espera activa. Sin embargo, el servidor debe ceder el procesador después de cada intento con el método yield(). Una vez retirado el mensaje, genera una respuesta, y procede a despertar al cliente que se encuentra a la espera dormido en el mensaje.

Note que el cliente no se comunica directamente con el buffer, sino con el mensaje, y es este último quien se comunica con el buffer. El servidor se comunica con el buffer; no se comunica directamente con el cliente. El buffer debe recibir la información de cuántos clientes hay, pero no de cuántos mensajes van a circular. El tamaño del buffer debe ser configurable.