Caso 1

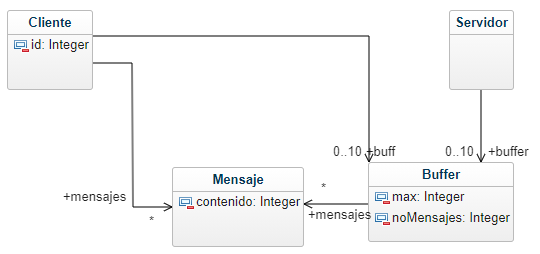
**Cliente:** El cliente es asignado a un buffer y cuando se ejecuta (***run***) crea un mensaje y agrega el mensaje al Buffer

**Mensaje:** Tiene un *contenido*

**Buffer:** Tiene un límite de mensajes. Cuando llega un mensaje directo del cliente se verifica que el número de mensajes en el buffer no sea máximo, sino lo pone a esperar (wait()). Para eliminar un mensaje, verifica que haya mensajes que eliminar, en caso contrario hace esperar al servidor (wait()).

Hay exclusión mutua para agregar un mensaje al buffer (addMensaje(Mensaje m)) o para borrar un mensaje por orden del servidor (removeMensaje()), para resolver el indeterminismo.

**Servidor:** Esta constantemente eliminando mensajes de Buffer (buff.removeMensaje()).



**Flujo de acción:**

1. El programa inicia
2. Se crean n clientes y n servidores y un buffer
3. Clientes generan colecciones de mensajes, y los envían al Buffer.
4. El Buffer puede agregar o no los mensajes dependiendo del cupo de mensajes faltante (en exclusión mutua).
5. El servidor removerá el último mensaje del Buffer (si existen mensajes en el Buffer), e imprime el contenido de los mensajes.
6. El programa termina