Universidad de los Andes

Ingeniería de Sistemas y Computación

ISIS 2203 Infraestructura Computacional

Taller de comunicación

El propósito de este taller es implementar en Java algunos esquemas de comunicación. En estos puntos solo debe usar `synchronized` y `wait`, `notify` y `notifyAll`.

1. *Comunicación sincrónica por canal.*

Este esquema sirve para comunicar dos threads usando un intermediario (el canal). La comunicación es solo entre dos y es sincrónica. Uno de ellos solo envía y el otro solo recibe (los mensajes serán sencillamente números enteros).

Tendremos 3 elementos: productor, consumidor y canal. El canal está compuesto de dos partes: el contenido y un indicador de que hay alguien en espera (cierto, hay alguien en espera; falso, el canal está desocupado).

El primero que llegue al canal, productor o consumidor, marca el canal como ocupado y se queda a la espera del segundo. El segundo que llegue se encarga de desencadenar la comunicación, y de despertar al primero.

El canal debe tener métodos para enviar y recibir, y es en estos donde se implementa el anterior comportamiento. El productor y el consumidor se limitan a intercambiar un cierto número predeterminado de mensajes invocando para ello los anteriores métodos.

Nota: para simplificar el problema, los `wait` no deben estar encerrados en un `try-catch`; en vez de esto, es mejor que el método lance excepciones.

2. *Comunicación sincrónica por nombre de proceso.*

Este esquema es parecido al anterior pero sin intermediario: no hay canal; en lugar de eso, el productor envía los mensajes directamente al consumidor. Por ende, solo tendremos 2 elementos: productor y consumidor.

En este caso, el contenido y el indicador de que hay alguien en espera se encuentran en el consumidor, así como los métodos para enviar y recibir.

3. ¿Qué ventajas y desventajas les ve a los anteriores métodos?