**Luis Alfonso Ruiz Botero - 201112453**

**Juan Daniel Carrillo Ramírez - 201613501**

**Infraestructura Computacional**

**Caso 3**

**1) Descripción de la Implementación**

Dado que se quería implementar ahora un pool de threads, se procedió a crear una constante con el número máximo de threads ejecutándose y se importaron las librerías ExecutorService y Executors.

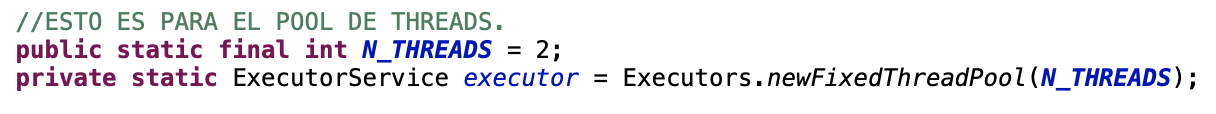


Imagen 1. Muestra la declaración para la inicialización del pool de threads.

Adicionalmente en el main, se eliminó la ejecución automatica del thread creado, haciendo que fuera el ExecutorService quien manejara los threads.



Imagen 2. Muestra la declaración para la inicialización del pool de threads.

Del lado del cliente, se llevaron a cabo varias modificaciones para la implementación de los escenarios de carga. La primera, fue añadir timeStamps en los lugares donde era necesario calcular tiempos. Esto se llevó a cabo utilizando variables de tipo long con ayuda de System.CurrentTimeMillis() en los lugares donde se debían calcular los tiempos.

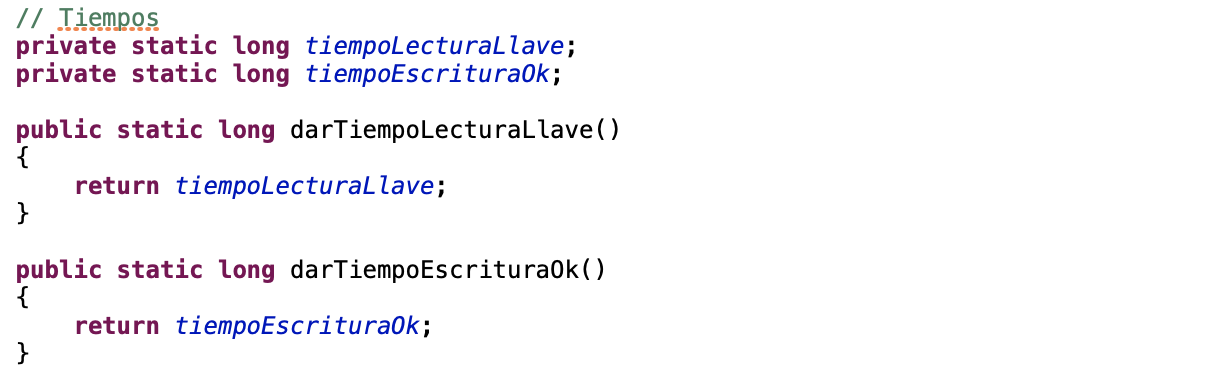


Imagen 3. Muestra la declaración para la inicialización de los tiempos.



Imagen 4. Muestra la declaración para la inicialización de los tiempos.

Adicionalmente, se realizaron cambios en la estructura de los protocolos. En la entrega pasada, era el usuario quien seleccionaba los algoritmos a utilizar, así como la información que mandaba con su ubicación. Debido a que se necesitaba simular usuarios concurrentes, estas selecciones se hicieron de manera automática (seleccionando siempre el primer elemento posible), y los mensajes con identificación y ubicación se enviaron con Strings por defecto.



Imagen 5. Muestra la declaración para la selección de algoritmos.

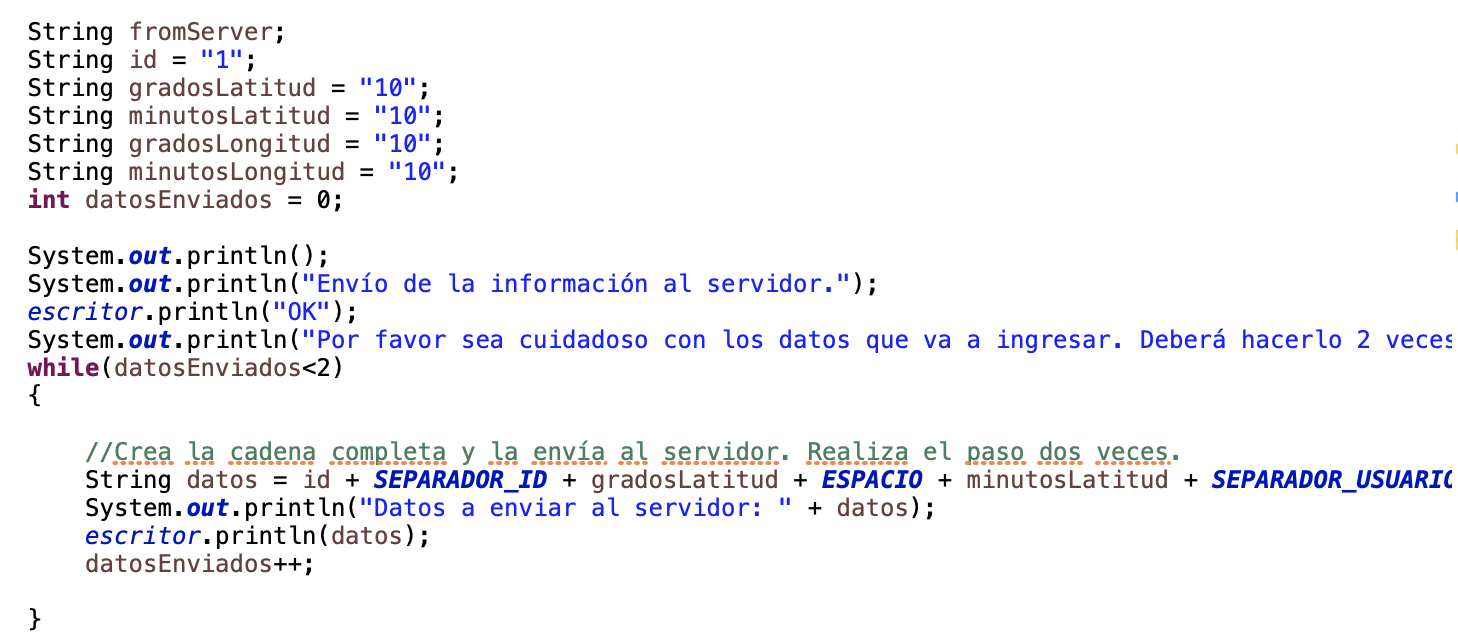


Imagen 6. Muestra la declaración para la selección de identificador y coordenadas.

Para terminar, se utilizaron las implementaciones de Gload y el método de getSystemCpuLoad() tal cual como estaban descritas en las guías.

**2) Identificación de la Plataforma**

Se realizó la identificación de la máquina donde se corrió el servidor. En un principio, se había dispuesto de 2 máquinas diferentes (una en donde se ejecutaba el servidor, y la otra donde se ejecutaba la carga), sin embargo se presentaron algunos inconvenientes con dicha máquina, por lo que se debió ejecutar todo en la segunda. La configuración de la primera y segunda máquina se muestran a continuación:

**Máquina 1:**

**Arquitectura (32 o 64 bits):** 64 bits.

**Número de núcleos (cores):** 4 núcleos. 8 procesadores lógicos

**Velocidad del procesador:** 2.5 GHZ.

**Tamaño de la memoria RAM:** 12 GB.

**Espacio de Memoria asignado a JVM:** 5 Gb.

**Máquina 2:**

**Arquitectura (32 o 64 bits):** 64 bits.

**Número de núcleos (cores):** 2 núcleos. 4 procesadores lógicos

**Velocidad del procesador:** 2.9 GHZ.

**Tamaño de la memoria RAM:** 8 GB.

**Espacio de Memoria asignado a JVM:** 3 Gb.

En principio, la máquina 1 se encargaría del servidor. Sin embargo, después de realizar varias pruebas y obtener diferentes mensajes de error se decidió realizar todo en la segunda máquina.

Un error típico que se encontró era que por ejemplo, el servidor no respondía de manera adecuada al envío de mensajes, como se puede observar en las siguientes imágenes:

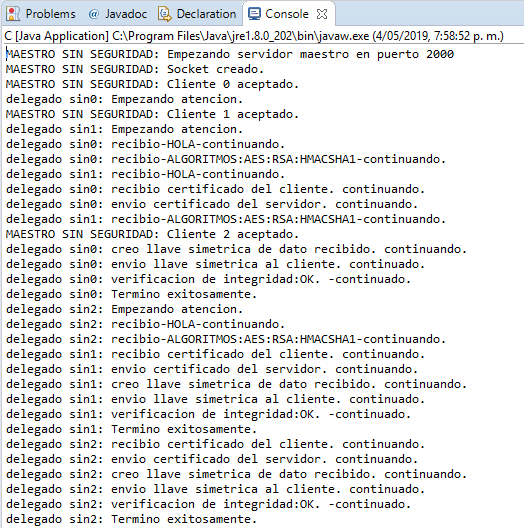


Imagen 7. El servidor funcionando correctamente al ejecutar 3 solicitudes diferentes.

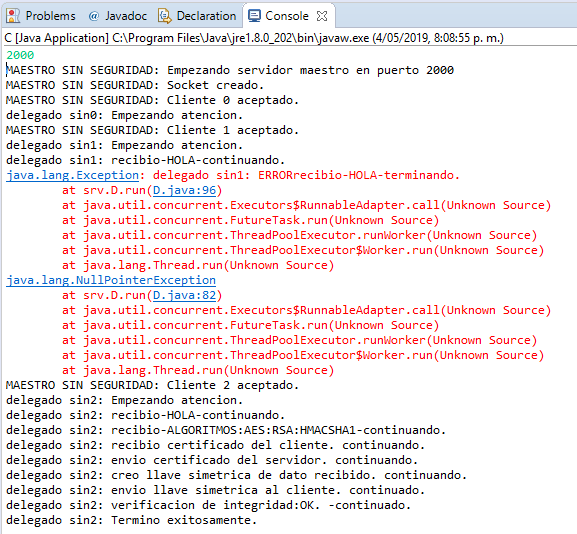


Imagen 7. El servidor funcionando de manera irregular, al ejecutar las mismas 3 solicitudes.