**Documento de Informe, Análisis y Entendimiento del Problema**

**Luis Miguel Gómez Londoño – 201729597**

**Infraestructura Computacional**

**Caso #3**

**Profesora: Sandra Rueda**

1. **Descripción detallada de la implementación de los monitores.**

**1.1 Cambios generales**

Para este caso se hicieron diversas modificaciones en el código para poder monitorear el desempeño de una aplicación servidor/cliente utilizando alta demanda de clientes y manejo de respuestas de servidores a través de un pool de threads… Las principales modificaciones que se hicieron a partir del caso #2 fueron la implementación del pool de threads en el servidor, así como el *GLoad* para la carga masiva de clientes. Como había que implementar un servidor inseguro además del que ya teníamos (con protocolo seguro), se modificó la clase del servidor para que cada protocolo tuviera un método y con esto se da la opción al usuario de elegir el tipo de servidor que se iba a utilizar. El servidor es quien decide que archivo se va a crear. Se hizo una adecuación para tener mas agilidad en la serie de pruebas en donde el usuario corre el servidor y luego elige el punto del trabajo que se va a realizar (P3 seguro, P4 Inseguro), el numero de escenario que va a realizar, y el numero de prueba de este que se va a realizar. Dados estos parámetros, por cada ejecución del servidor se crean dos archivos, el [log de la ejecución](../data/logs/) en *data/logs/<P3 ó P4>/Escenario<Numero del escenario ejecutado>R<Numero de prueba de ese escenario>-resultados.txt*, y se crea el archivo de datos de análisis que sigue el formato:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Delegado* | *Tiempo* | *CPU Load Inicial* | *CPU Load Mitad* | *CPU Load Final* | *Estado Final* |

Y están ubicados en *data/datos/<P3 ó P4>/Escenario<Numero del escenario ejecutado>R<Numero de prueba de ese escenario>-datos.csv*

Lo mismo se hizo para el cliente, esta vez, mas personalizado aún… Como ya no se tenia que usar el *main* del cliente porque el *generator* usa es el constructor (objeto), entonces se dejo el *main* por si se quería dejar alguna opción de correr un solo cliente (Como solía ser) y se implemento una personalización de datos para la creación de clientes en el *generator*, y, al igual que el servidor, poder elegir si se va a utilizar un cliente con protocolo seguro o no. Todo esto de la personalización precisamente para que las pruebas se llevaran a cabo mas eficientemente sin tener que cambiar directamente el código.

**1.2 Cambios para la implementación de monitores**

Se realizaron los siguientes cambios para la implementación de los monitores.

1. **Identificación de la plataforma.**

**2.1 Características de la plataforma**

**Arquitectura:** *64 bits*

**Número de núcleos:** *Dual-Core Intel Core i5*

**Velocidad del procesador:** *2,9 GHz*

**Tamaño de memoria RAM:** *8 GB*

**Espacio de memoria asignado a la JVM:**

1. **Comportamiento de la aplicación con diferentes estructuras de administración de la concurrencia.**
2. **Comportamiento de la aplicación ante diferentes niveles de seguridad.**