# Introducción

Los test de rendimiento evalúan el rendimiento de nuestra aplicación con el fin de poder transmitirle al cliente cual es la carga máxima que soporta su sistema. Puesto que no se puede probar directamente en el sistema, el lugar donde se debería probar es en la máquina de Pre-Producción ya que es el lugar más parecido al entorno real. Debido a que la máquina de Pre- Producción es una máquina virtual ejecutada desde distintos ordenadores, el rendimiento varía entre ellas y por esto hemos decidido recopilar los dos test que, tras ejecutarlo varias veces en distintas máquinas, obtienen un peor rendimiento (y destacable) entre los otros Test.

A continuación se van a ir detallando los citados test, ya que suponen el cuello de botella de la aplicación y se van a ir proponiendo opciones para mejorar el rendimiento de las acciones realizadas por los test consiguiendo así mejorar el rendimiento de la aplicación en general.

El primer test será denominado de aquí en adelante Test\_A y hace referencia al caso de uso creado por el requisito funcional de nivel A 2.1 (Acme-Barter 2.0) el cual implica manejar las descripciones de los atributos de los objetos. El segundo test (de aquí en adelante Test\_B) hace referencia al caso de uso creado por el requisito funcional de nivel C 12.2 (Acme-Barter 1.0) el cual implica manejar los términos legales.

# Detalles del entorno

Antes de continuar debemos detallar en que máquina se han lanzado dichos tests y que configuraciones tenía tanto la máquina virtual como los propios test:

## Máquina física

Portátil HP-Pavilion con I7, 8GB de RAM y disco HDD

## Máquina virtual

2 Núcleos al 100% con las características PAE/NX habilitadas y con 1536 MB de RAM

## Detalles JMeter

A ambos test se le han introducido un retraso de campana gaussiana de 1500 ms en cada acción que sería realizada por un usuario real (hacer click o rellenar un formulario) con el fin de que sea lo más real posible.

Respecto al número de usuarios, el “ramp-up period” y el número de iteraciones por cada usuario los datos en cada test han sido los siguientes:

* Test\_A: 75 usuarios, en 1 segundo (ramp-up period) y cada usuario se ejecuta 50 veces.
* Test\_B: 100 usuarios, en 1 segundo (ramp-up period) y cada usuario se ejecuta 100 veces.

# Desarrollo de los test

Ambas pruebas tienen como cuello de botella el mismo error, realizar muchas peticiones a la base de datos de escritura/lectura del tipo save() o findAll() para comprobar los datos introducidos. Como podemos observar en ambas gráficas, esto es apreciable en las columna “90% Line” en las líneas listar y crear en el Test\_A y en listar, crear y editar en el Test\_B.

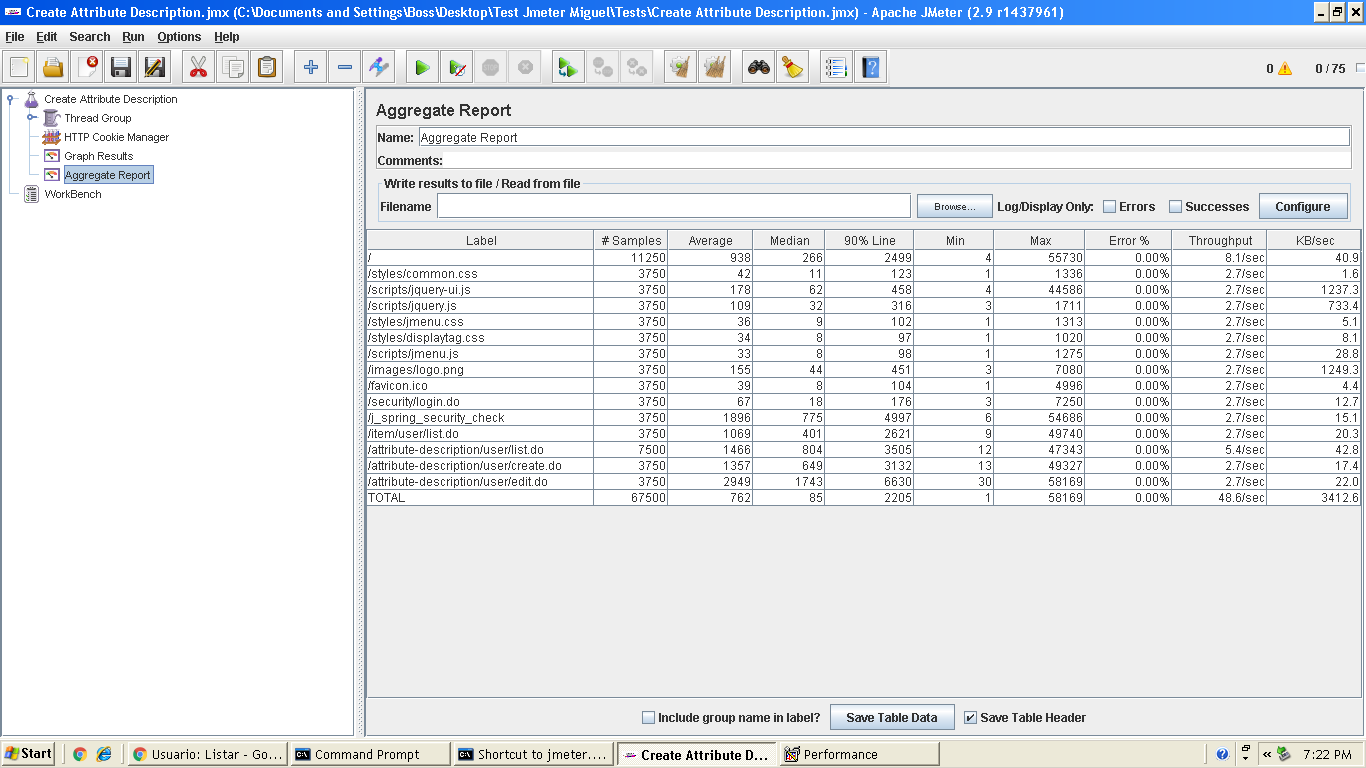


Tabla peticiones web Test\_A

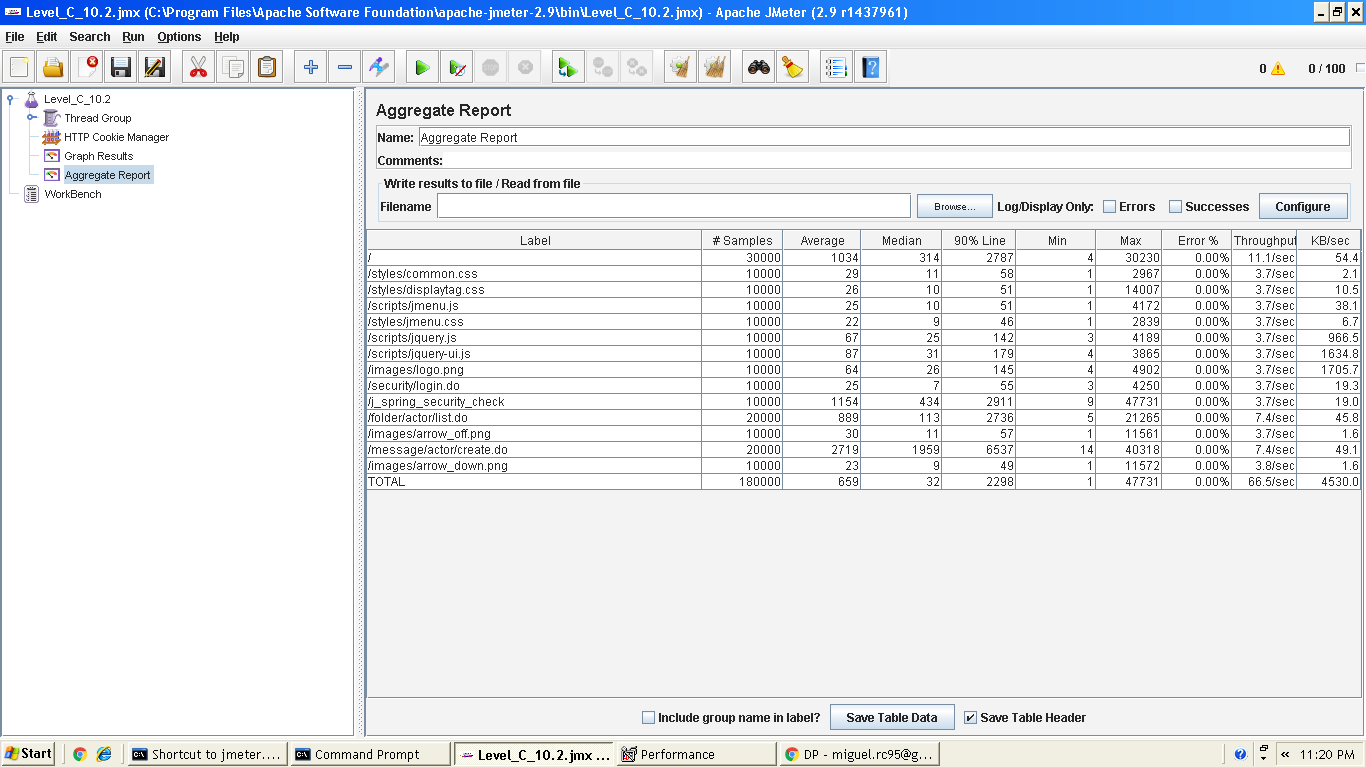


Tabla peticiones web Test\_B

Esto se debe (como podemos observar en las siguientes gráficas) al estado del procesador, la memoria RAM y el acceso al disco del sistema los cuales se encuentran continuamente al límite de sus posibilidades.

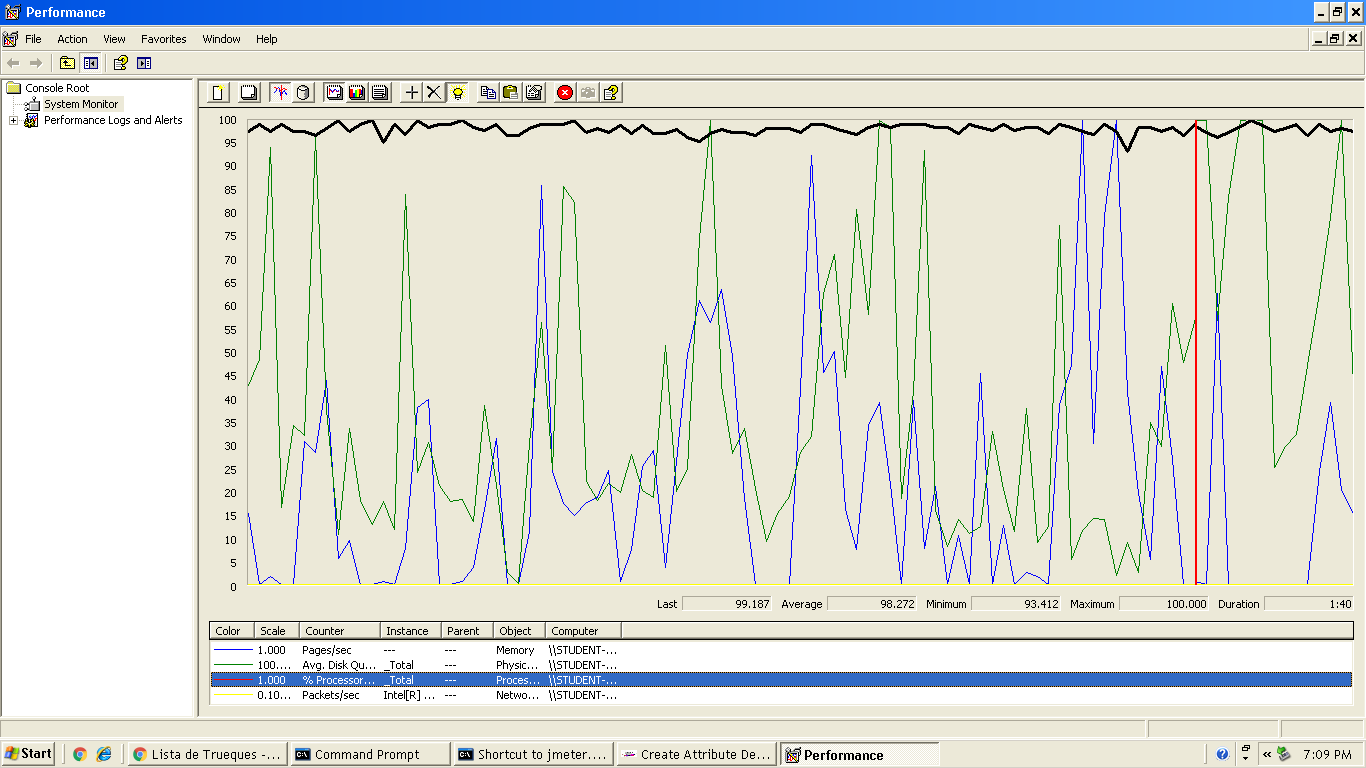


Ilustración carga sistema durante ejecución de Test\_A

Nota: Los colores representan la memoria ram (azul), el disco duro (verde), el procesador (negro) y la tarjeta de red (amarillo).

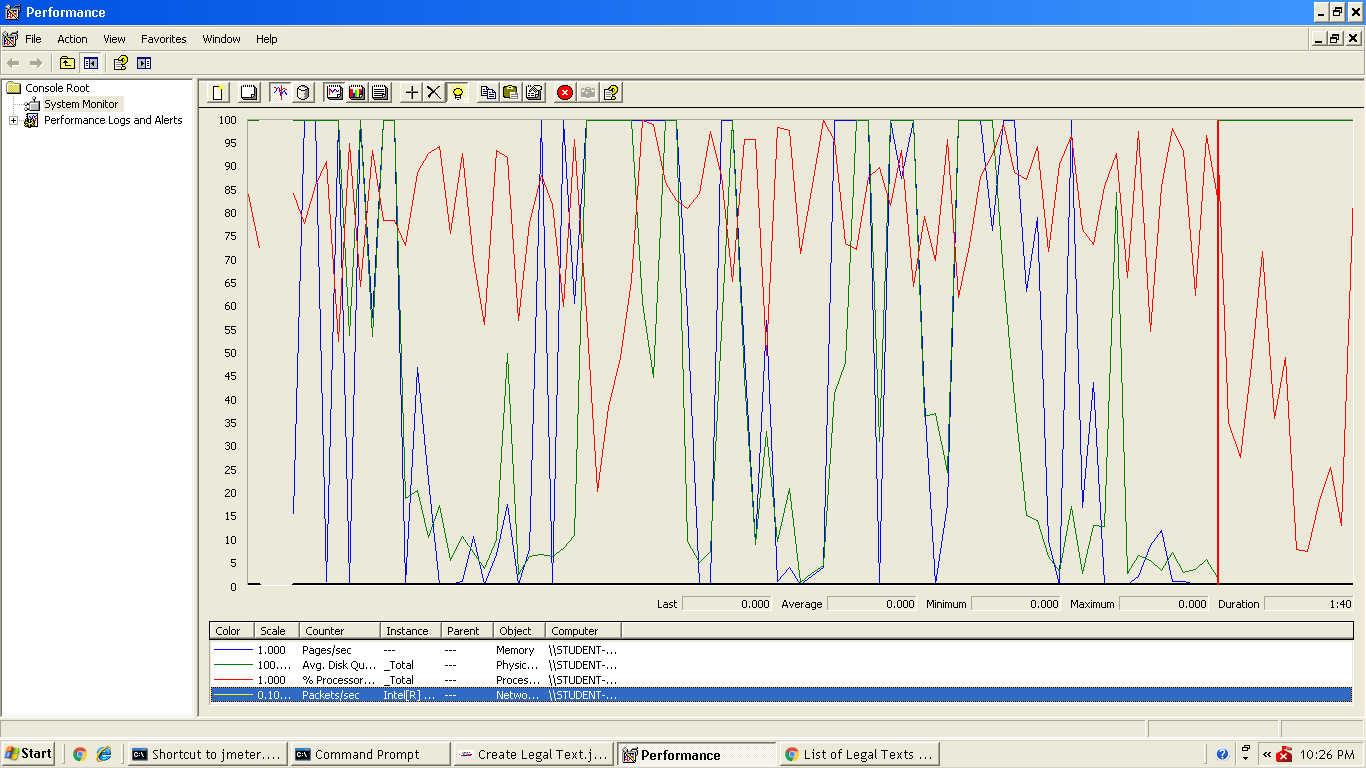


Ilustración carga sistema durante ejecución de Test\_B

Nota: Los colores representan la memoria ram (azul), el disco duro (verde), el procesador (rojo) y la tarjeta de red (negro).

Para mejorar estos resultados, la opción por parte del cliente es comprar procesadores con más potencia, discos duros cuyo acceso de lectura escritura sea más rápido y memorias RAM más rápidas y con más capacidad (lo que podría evitar tener que mejorar el disco duro temporalmente) y por parte de los desarrolladores de la aplicación (nosotros) usar estas llamadas “lentas” el menor número de veces posible lo que conllevaría a programar de manera más eficiente en general.

Con todo esto podemos concluir que el límite de nuestro sistema serían unos 75 usuarios por segundo.