UAM UNIVERSIDAD AUTONOMA DE MADRID		Escuela Politécnica Superior Ingeniería Informática Prácticas de Sistemas Informáticos 2			
Grupo	2401	**Práctica**	2	**Fecha**	8/04/2015
Alumno		Kasner Tourné, Cristina			
Alumno		Guridi Mateos, Guillermo			

Ejercicio 1

Siguiendo todos los pasos anteriores, defina el plan completo de pruebas para realizar las tres ejecuciones secuenciales sobre los tres proyectos definidos hasta ahora (P1-base, P1-ws-cli, P1-ejb). Adjunte el fichero generado P2.jmx al entregable de la práctica.

No hemos tenido ningún problema en definir el plan de pruebas tal y como se nos indicaba en el enunciado. Adjuntamos una imagen con el árbol de resultados.

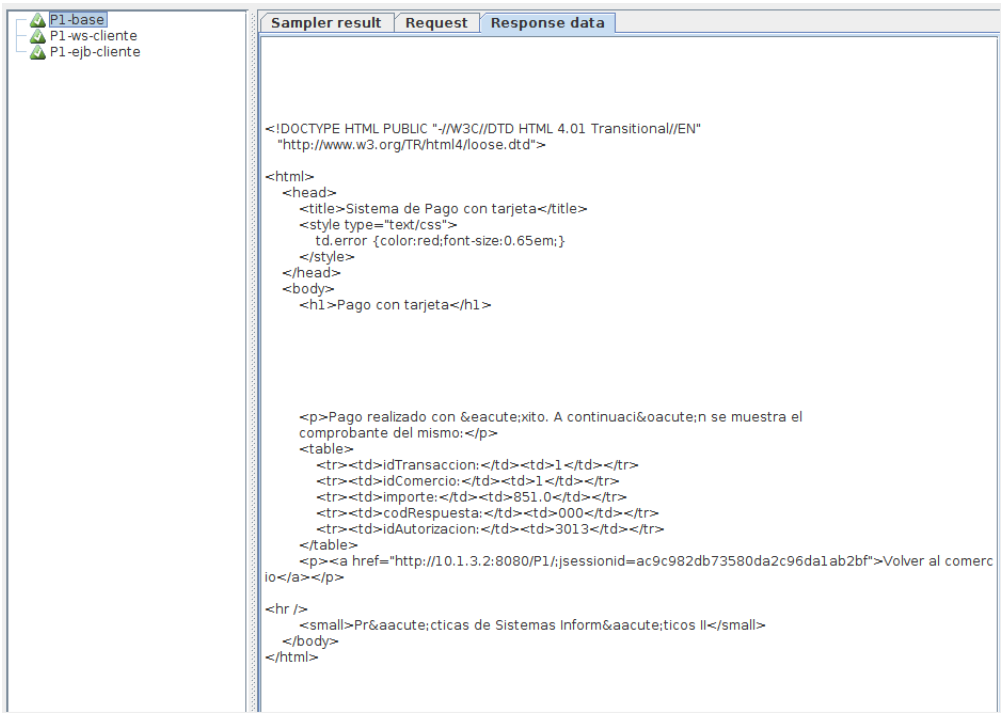


Figura 1: Árbol de resultados

Ejercicio 2

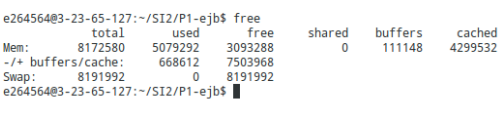
Preparar los PCs con el esquema descrito. Para ello:

- Anote en la memoria de prácticas las direcciones IP asignadas a cada PC.
- Detenga el servidor de GlassFish de los PCs físicos
- Inicie los servidores GlassFish en las máquinas virtuales
- Repliegue todas las aplicaciones o pruebas anteriores (P1-base, P1-ws, etc), para limpiar posibles versiones incorrectas.
- Revise y modifique si es necesario los ficheros build.properties (propiedad "nombre") de cada versión, de modo que todas las versiones tengan como URL de despliegue las anteriormente indicadas (P1-base, P1-ws, P1-ejb).
- Despliegue las siguientes prácticas: P1-base, P1-ws, P1-ejb, con el siguiente esquema:
  - El destino del despliegue en todos los casos será PC2VM con IP 10.X.Y.2 (as.host o as.host.client en P1-ws)
  - La base de datos en todos ellos será la de PC1VM con IP 10.X.Y.1 (db.host)
  - En el caso particular de P1-ws, el servidor SOAP estará en 10.X.Y.1 (variable as.host.server)

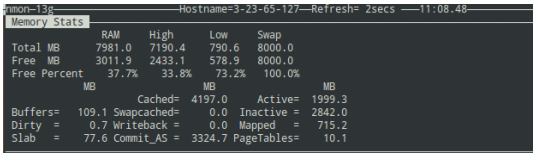
Tras detener / iniciar todos los elementos indicados, anotar la salida del comando "free" así como un pantallazo del comando "nmon" (pulsaremos la tecla "m" para obtener el estado de la RAM) tanto en las máquinas virtuales como los PCs físicos. Anote sus comentarios en la memoria. Pruebe a ejecutar un pago "de calentamiento" por cada uno de los métodos anteriores y verifique que funcionan (comprobar resultados en el árbol de resultados).

Dirección IP asignada al PC 1 : 10.1.3.1
Dirección IP asignada al PC 2 : 10.1.3.2

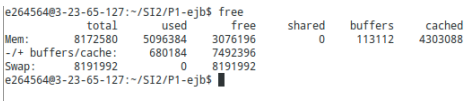
Tras desplegar y ejecutar free y nmon obtenemos los siguientes resultados:



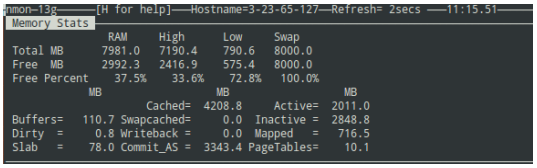
(a) free PC1 antes



(b) nmon PC1 antes



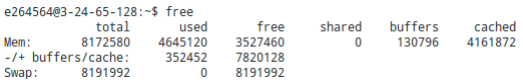
(c) free PC1 después



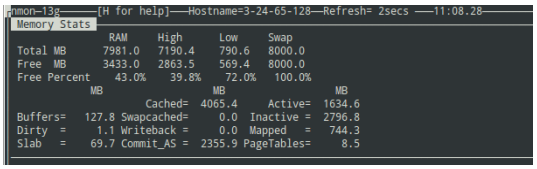
(d) nmon PC1 después

Figura 2: PC 1

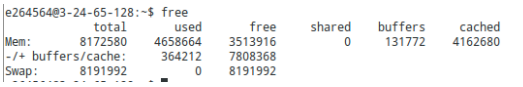
El comando free nos muestra el uso de memoria actual en el sistema operativo en el que se ejecute. Para este ejercicio nos hemos fijado sobre todo en el porcentaje de memoria libre, comparando las máquinas y PCs sin haber desplegado nada (para ver el consumo base de memoria) con los valores obtenidos tras desplegar toda la aplicación.



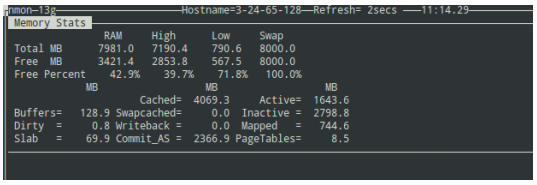
(a) free PC2 antes



(b) nmon PC2 antes

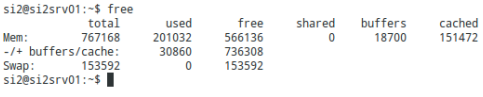


(c) free PC2 después

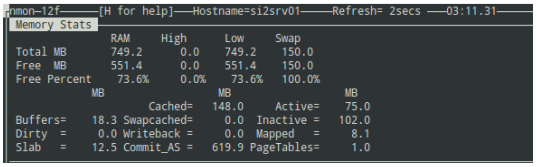


(d) nmon PC2 después

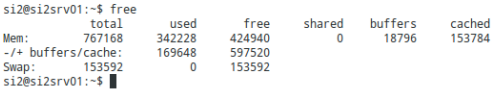
Figura 3: PC 2



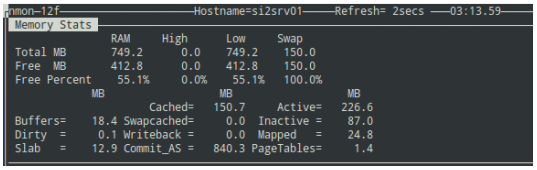
(a) free MV1 antes



(b) nmon MV1 antes

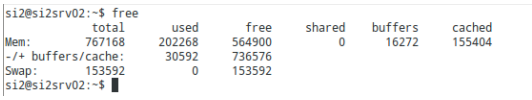


(c) free MV1 después

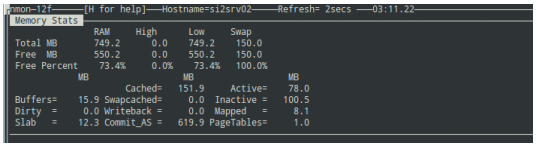


(d) nmon MV1 después

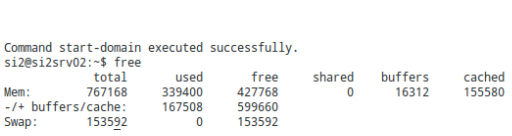
Figura 4: máquina virtual 1



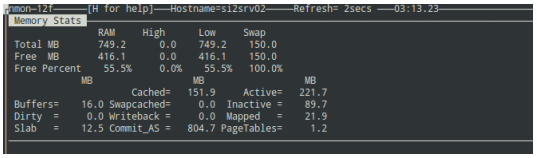
(a) free MV2 antes



(b) nmon MV2 antes



(c) free MV2 después



(d) nmon MV2 después

Figura 5: máquina virtual 2

Podemos observar que al hacer free en los PCs, no hay mucha diferencia en el uso de la memoria estando el glassfish y el postgres arrancados o no. Probablemente esto se deba a que los arrancamos dentro de la maquina virtual que probablemente ya tenga parte de la memoria del sistema virtualizado reservada. Ésto se confirma al comprobar que dentro de las máquinas virtuales si que podemos observar un aumento significativo de la memoria usada (entorno a los 140MB).

Nmon, al usarlo solo con 'm' para obtener el mismo tipo de estadísticas nos ofrece las mismas conclusiones.

Adicionalmente, como era de esperar, los valores iniciales son parecidos en las máquinas virtuales entre ellas y en los PCs entre ellos.

Ejercicio 3

Ejecute el plan completo de pruebas sobre las 3 versiones de la práctica, empleando el esquema de despliegue descrito anteriormente. Realice la prueba tantas veces como necesite para eliminar ruido relacionado con procesos periódicos del sistema operativo, lentitud de la red u otros elementos.

- Compruebe que efectivamente se han realizado todos los pagos. Es decir, la siguiente consulta deberá devolver "3000": SELECT COUNT(*) FROM PAGO;

```
e264564@3-24-65-128:~$ echo "SELECT count(*) FROM pago;" | psql -U alumnodb -h 10.1.3.1 visa
 count
-------
  3000
(1 row)
```

Figura 6: Consulta

- Compruebe que ninguna de las peticiones ha producido un error. Para ello revise que la columna %Error indique 0 % en todos los casos.

Label	# Samples	Average	Median	90% Line	Min	Max	Error %	Throughput	KB/sec
P1-base	1000	9	9	11	7	30	0,00%	100,5/sec	127,6
P1-ws-cliente	1000	71	68	83	55	324	0,00%	14,0/sec	18,1
P1-ejb-cliente	1000	12	12	15	9	35	0,00%	77,5/sec	100,3
TOTAL	3000	31	12	73	7	324	0,00%	31,6/sec	40,6

Figura 7: Comprobación error

Una vez que los resultados han sido satisfactorios:

Anote los resultados del informe agregado en la memoria de la práctica.

- Salve el fichero server.log que se encuentra en la ruta glassfish/domains/domain1/logs de Glassfish y adjúntelo con la práctica.
- Añada a la memoria de prácticas la siguiente información: ¿Cuál de los resultados le parece el mejor? ¿Por qué? ¿Qué columna o columnas elegiría para decidir este resultado?

El resultado que nos parece mejor es P1-base ya que tiene el mayor Throughput, para eso nos fijamos en la penúltima columna, que es la que nos indica las peticiones completadas por segundo, también sería importante contemplar la columna del percentil 90 % ya que nos garantiza que la mayoría de las peticiones tardarán menos que ese dato. Una vez más gana P1-base, incluso con un poquito más de margen.

Pierde claramente P1-ws, probablemente por las peticiones http extra que tiene que realizar para procesar cada pago.

Ejercicio 4

Adaptar la configuración del servidor de aplicaciones a los valores indicados. Guardar, como referencia, la configuración resultante, contenida en el archivo de configuración localizado en la máquina virtual en $ J2EE_HOME/domains/domain1/config/domain.xml1. Para obtener la versión correcta de este archivo es necesario detener el servidor de aplicaciones. Incluir este fichero en el entregable de la práctica. Se puede copiar al PC del laboratorio con scp.

A parte de realizar las modificaciones de configuración creamos un pequeño script (con el comando asadmin) que las modifica por nosotros (excepto las de autodeploy y reload), para poder realizar pruebas en los laboratorios más rapidamente.

Ejercicio 5

Registrar en la hoja de cálculo de resultados los valores de configuración que tienen estos parámetros.

Parámetros de configuración

Elemento	Parámetro	Valor
JVM Settings	Heap Máx. (MB)	512
JVM Settings	Heap Mín. (MB)	512
HTTP Service	Max.Thread Count	5
HTTP Service	Queue size	4096
Web Container	Max.Sessions	-1
Visa Pool	Max.Pool Size	32

Figura 8

Ejercicio 6

Tras habilitar la monitorización en el servidor, repita la ejecución del plan de pruebas anterior. Durante la prueba, vigile cada uno de los elementos de monitorización descritos hasta ahora. Responda a las siguientes cuestiones:

- A la vista de los resultados, ¿qué elemento de proceso le parece más costosa? ¿Red? ¿CPU? ¿Acceso a datos? En otras palabras, ¿cuál fue el elemento más utilizado durante la monitorización con nmon en un entorno virtual? (CPU, Memoria, disco,...)
- ¿Le parece una situación realista la simulada en este ejercicio? ¿Por qué?
- Teniendo en cuenta cuál ha sido el elemento más saturado, proponga otro esquema de despliegue que resuelva esa situación.

Estos fueron los resultados obtenidos:

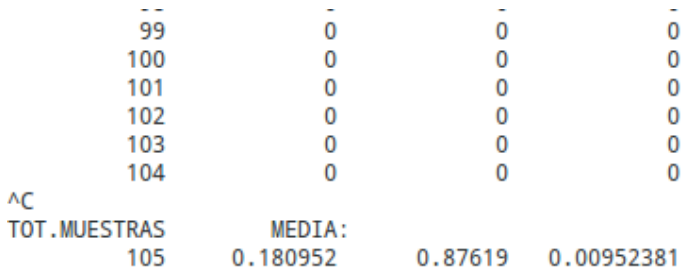


Figura 9

Lo más costoso resulta ser la CPU. Como se puede apreciar en el pantallazo del nmon, java está ocupando el 87 % del procesador y éste ultimo está trabajando en intervalos cercanos al 100 % durante la duración de las pruebas. La memoria se ocupa, pero no parece ser un factor determinante en el rendimiento de este sistema. El uso de red no parece crer ningún cuello de botella ya que los picos máximos de tráfico están en torno a los 100kbps (al exterior y del exterior), lo cual es perfectamente asumible en una red local.

Cabe destacar, como se puede apreciar en el último pantallazo, que durante la prueba de P1-ws se observa un mayor tráfico en la red.

No consideramos esta prueba una situación realista ya que todos los clientes vienen de uno en uno, y son capaces de hacer el pago directamente, en vez de pasar por todo el proceso de la pasarela. Seróia mejor si vinieran concurrentemente y tardaran algo de tiempo en completar los datos del pago, cosa que no se contempla aquí al hacer la petición directamente a procesapago.

Como el elemento más saturado ha sido la CPU, probablemente se debería, o bien, mejorar la cpu de las máquinas de producción, o desplegar varios servidores en paralelo con un load balancer a la entrada. Como la base de datos está en otra máquina no habría ningún problema en duplicar directamente.

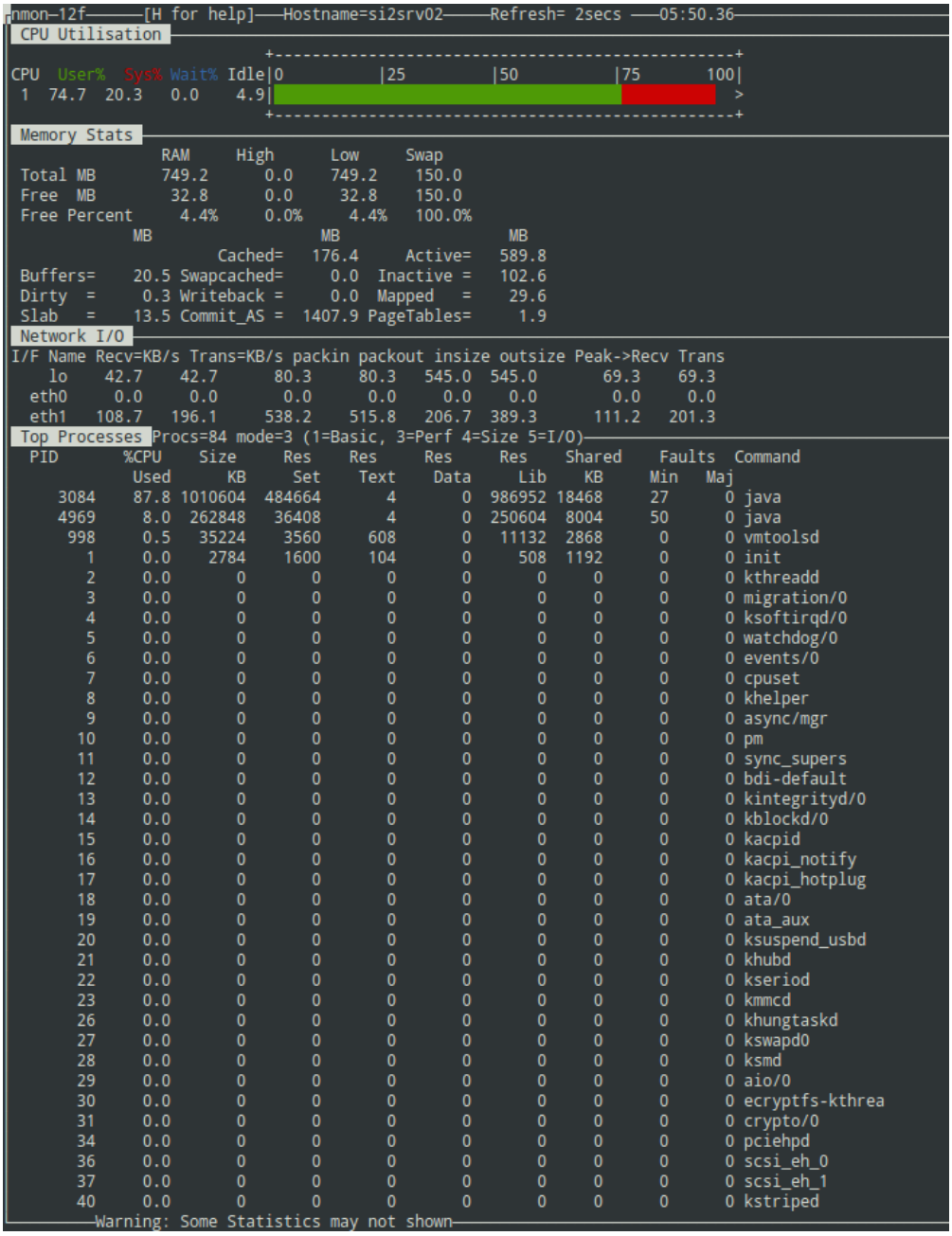


Figura 10: nmon durante la prueba de P1-base

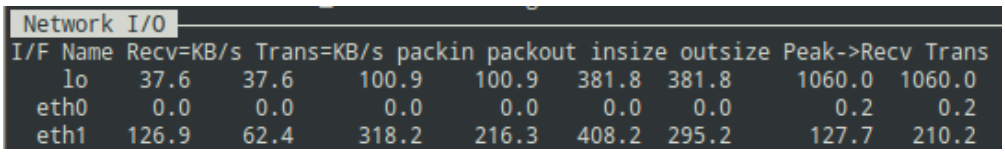


Figura 11: nmon durante la prueba de P1-ws

Ejercicio 7

Preparar el script de JMeter para su ejecución en el entorno de pruebas. Cambiar la dirección destino del servidor para que acceda al host en el que se encuentra el servidor de aplicaciones. Crear también el directorio datagen en el mismo directorio donde se encuentre el script, y copiar en él el archivo listado.csv, ya que de dicho archivo, al igual que en las prácticas anteriores, se obtienen los datos necesarios para simular el pago.

A continuación, realizar una ejecución del plan de pruebas, con un único usuario, una única ejecución, y un think time bajo (entre 1 y 2 segundos) para verificar que el sistema funciona correctamente. Comprobar, mediante el listener View Results Tree que las peticiones se ejecutan correctamente, no se produce ningún tipo de error y los resultados que se obtienen son los adecuados. Una vez comprobado que todo el proceso funciona correctamente, desactivar dicho listener del plan de pruebas para que no aumente la carga de proceso de JMeter durante el resto de la prueba. Este ejercicio no genera información en la memoria de la práctica, realícelo únicamente para garantizar que la siguiente prueba va a funcionar.

Decidimos ejecutar el monitor directamente en la maquina-2 y recoger los resultados de ella cada vez que terminamos una prueba.

Ejercicio 8

Obtener la curva de productividad, siguiendo los pasos que se detallan a continuación:

- Previamente a la ejecución de la prueba se lanzará una ejecución del script de pruebas (unas 10 ejecuciones de un único usuario) de la que no se tomarán resultados, para iniciar el sistema y preparar medidas consistentes a lo largo de todo al proceso.

  Borrar los resultados de la ejecución anterior. En la barra de acción de JMeter, seleccionar Run → Clear All.

- Borrar los datos de pagos en la base de datos VISA.
- Seleccionar el número de usuarios para la prueba en JMeter (parámetro C de la prueba)
- Conmutar en JMeter a la pantalla de presentación de resultados, Aggregate Report.
- Ejecutar la prueba. En la barra de accioón de JMeter, seleccionar Run → Start.
- Ejecutar el programa de monitorización si2-monitor.sh.
  - Arrancarlo cuando haya pasado el tiempo definido como rampa de subida de usuarios en JMeter.
  - Detenerlo cuando esté a punto de terminar la ejecución de la prueba.
  - Registrar los resultados que proporciona la monitorización en la hoja de cálculo.
- Durante el periodo de monitorización anterior, vigilar que los recursos del servidor si2srv02 y del ordenador que se emplea para realizar la prueba no se saturen, mediante inspección del programa de monitorización nmon que se ejecuta en ambas máquinas.
- Finalizada la prueba, salvar el resultado de la ejecución del Aggregate Report en un archivo, y registrar en la hoja de cálculo de resultados los valores Average, 90 % line y Throughput para las siguientes peticiones:
  - ProcesaPago.

- Total.

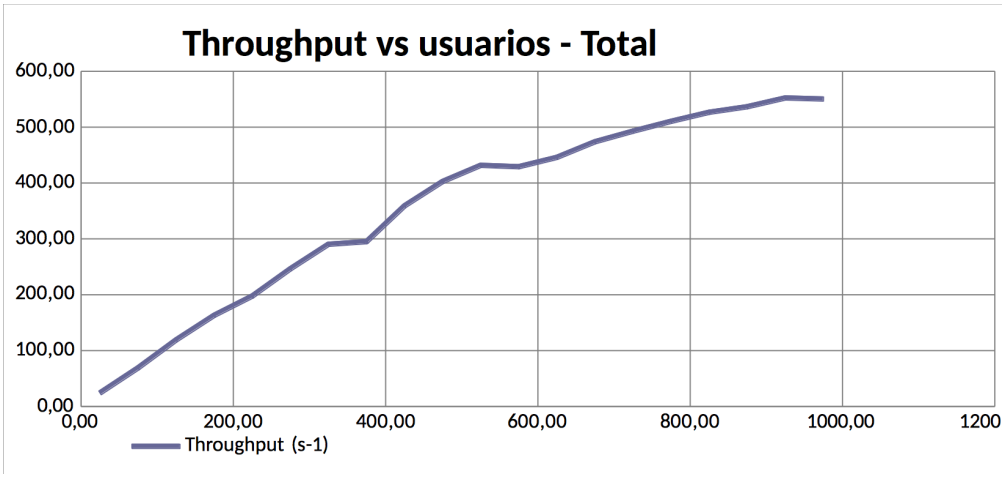
Una vez realizadas las iteraciones necesarias para alcanzar la saturación, representar la curva de Throughput versus usuarios.

Realizamos las pruebas pedidas empezando en 25 y subiendo de 50 en 50 hasta 775, con la ayuda de un pequeño script para minimizar la influencia del factor humano en los resultados.

Ejercicio 9

Responda a las siguientes cuestiones:

- A partir de la curva obtenida, determinar para cuántos usuarios conectados se produce el punto de saturación, cuál es el máximo throughput que se alcanza en el mismo, y el throughput máximo que se obtiene en zona de saturación.
- Analizando los valores de monitorización que se han ido obteniendo durante la elaboración de la curva, sugerir el parámetro del servidor de aplicaciones que se cambiaría para obtener el punto de saturación en un número mayor de usuarios.
- Realizar el ajuste correspondiente en el servidor de aplicaciones, reiniciarlo y tomar una nueva muestra cercana al punto de saturación. ¿Ha mejorado el rendimiento del sistema? Documente en la memoria de prácticas el cambio realizado y la mejora obtenida.



La curva obtenida es bastante suave, pero estimamos que el punto de saturación se encuentra en torno a los 900 usuarios. A partir de ahí el trhoughput del servidor aumenta bastante más lentamente y hasta decrece un poco.

El throughput máximo que hemos alcanzado en el punto de saturación está en torno a las 530 peticiones por segundo. En la zona de saturación aumenta hasta las 550 peticiones por segundo.

Viendo los parámetros del servidor, vemos que la cpu está parada la mayoría del tiempo y que lo que aumenta más rapidamente es el número de peticiones encoladas a la entrada del servidor. Así que decidimos aumentar el número de hilos para recibir las peticiones.

Parámetros de configuración		
Elemento	**Parámetro**	**Valor**
JVM Settings	Heap Máx. (MB)	512
JVM Settings	Heap Mín. (MB)	512
HTTP Service	Max.Thread Count	10
HTTP Service	Queue size	4096
Web Container	Max.Sessions	-1
Visa Pool	Max.Pool Size	32

Figura 12: nuevos parámetros

Con los nuevos prámetros obtuvimos estos resultados, mucho mejores. Se puede ver como el punto de saturación aumenta más o menos hasta los 1000 usuarios y throughput en saturación hasta 600-610 peticiones por segundo.