

		<b>Escuela Politécnica Superior</b> <b>Ingeniería Informática</b> <b>Prácticas de Sistemas Informáticos 2</b>			
<b>Grupo</b>	<b>2401</b>	<b>Práctica</b>	<b>4</b>	<b>Fecha</b>	<b>04/04/2017</b>
<b>Alumno/a</b>		Hernando, Bernabé, Amanda			
<b>Alumno/a</b>		Pérez, Manso, Pablo			

## Práctica 2: Rendimiento

### Ejercicio número 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, P1-ejb).**

El fichero generado *P2.jmx* se adjunta en la entrega. Contiene tres planes de pruebas, muy similares, para cada uno de los proyectos. Para configurar el plan de pruebas se han ido añadiendo una serie de atributos en el siguiente orden:

1. Parámetros por defecto para los generadores de peticiones.
2. Variable de usuario "samples" que nos permite indicar el número de muestras que queremos tomar en todas las pruebas.
3. Grupo de hilos que simula el conjunto de usuarios que accede a la aplicación.
4. Variable aleatoria "importe" que permite que el importe de cada pago tenga un valor comprendido entre 1 y 1000.
5. Variable contador: el identificador de la transacción es un parámetro clave que no se puede repetir en cada pago de la tienda
6. Conjunto de datos aleatorios: el resto de los datos del pago (número de tarjeta, titular, fecha de emisión, caducidad y código de verificación de la tarjeta) se toman aleatoriamente de un archivo de datos .csv.
7. Generador de peticiones HTTP.
8. Aggregate Report para medir los resultados.

### Ejercicio número 2:

**Preparar los PCs con el esquema descrito en la Figura 21. Para ello:**

- **Anote en la memoria de prácticas las direcciones IP asignadas a cada PC.**  
Con el comando ifconfig obtenemos las direcciones IP de los Pc's:
  - PC1: 150.244.64.216
  - PC2: 150.244.64.215
 Las direcciones de las máquinas virtuales las asignamos nosotros:
  - MV1: 10.1.4.1
  - MV2: 10.1.4.2
- **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: nombre=P1-base  
 P1-ejb:
  - Cliente: nombre=P1-ejb-cliente-remoto
  - Servidor: nombre=P1-ejb
 P1-ws: nombre=P1-ws
- **Revise y modifique si es necesario el fichero glassfish-web.xml, para indicar la IP del EJB remoto que usa P1-ejb-cliente.**

La IP se modifica a 10.1.4.1 ya que ahora, como se explica a continuación, el servidor se despliega en la máquina virtual 1.

- **Despliegue las siguientes prácticas: P1-base, P1-ws, P1-ejb-servidor-remoto y P1-ejb-cliente-remoto, con el siguiente esquema:**
  - El destino de despliegue de la aplicación P1-base será PC2VM con IP 10.X.Y.2 (as.host).
  - El destino del despliegue de la parte cliente de P1-ws y de P1-ejb-cliente-remoto será PC2VM con IP 10.X.Y.2 (as.host.client).
  - El destino del despliegue de la parte servidor de P1-ws y de P1-ejb-servidor-remoto será PC1VM con IP 10.X.Y.1 (as.host.server).
  - La base de datos en todos ellos será la de PC1VM con IP 10.X.Y.1 (db.host).

La configuración final se encuentra en los respectivos ficheros build.properties y postgresql.properties.

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.

#### PC1:

```
e302505@3-12-65-79:~$ free
              total        usado        libre        compart.        búffers        almac.
Mem:          8049900      7771888      278012      1102064        60612      5923292
-/+ buffers/cache:      1787984      6261916
Intercambio:    4095992          16      4095976

nmon-14g-----Hostname=3-12-65-79-----Refresh= 2secs
Memory Stats
RAM      High      Low      Swap      Page Size=4 KB
Total MB  7861.2    -0.0    -0.0    4000.0
Free MB   251.9    -0.0    -0.0    4000.0
Free Percent 3.2%  100.0%  100.0%  100.0%
MB
Cached= 5793.5      Active= 2884.9
Buffers= 59.3 Swapcached= 0.0 Inactive = 4425.6
Dirty = 0.5 Writeback = 0.0 Mapped = 1087.9
Slab = 183.7 Commit_AS = 5105.4 PageTables= 30.2
```

#### PC2:

```
e250858@3-11-65-78:~$ free
              total        usado        libre        compart.        búffers        almac.
Mem:          8049900      7476756      573144      1107492        40980      5972132
-/+ buffers/cache:      1463644      6586256
Intercambio:    4095992          0      4095992

nmon-14g-----Hostname=3-11-65-78-----Refresh= 2secs
Memory Stats
RAM      High      Low      Swap      Page Size=4 KB
Total MB  7861.2    -0.0    -0.0    4000.0
Free MB   556.6    -0.0    -0.0    4000.0
Free Percent 7.1%  100.0%  100.0%  100.0%
MB
Cached= 5832.5      Active= 2584.9
Buffers= 40.1 Swapcached= 0.0 Inactive = 4420.2
Dirty = 0.6 Writeback = 0.0 Mapped = 1119.1
Slab = 180.2 Commit_AS = 5348.0 PageTables= 33.6
```

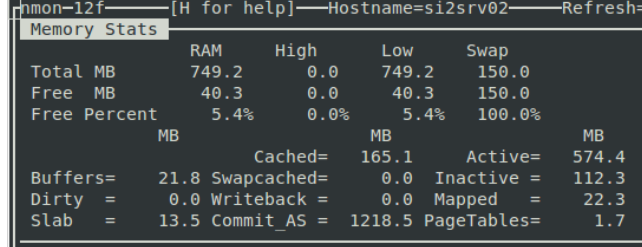
#### MV1:

```
si2@si2srv01:~$ free
              total        used        free        shared        buffers        cached
Mem:          767168      746800      20368          0        30188      195656
-/+ buffers/cache:      520956      246212
Swap:          153592          0      153592

nmon-12f-----Hostname=si2srv01-----Refresh=
Memory Stats
RAM      High      Low      Swap
Total MB  749.2    0.0    749.2    150.0
Free MB   17.9    0.0    17.9    150.0
Free Percent 2.4%  0.0%  2.4%  100.0%
MB
Cached= 191.2      Active= 566.6
Buffers= 29.5 Swapcached= 0.0 Inactive = 141.0
Dirty = 0.1 Writeback = 0.0 Mapped = 26.1
Slab = 14.2 Commit_AS = 1276.9 PageTables= 1.8
```

MV2:

```
si2@si2srv02:~$ free
              total        used        free      shared    buffers     cached
Mem:          767168      724528      42640           0        22288      168972
-/+ buffers/cache:      533268      233900
Swap:         153592           0        153592
```



El comando **free** en Linux muestra la cantidad de memoria libre y usada que tiene el sistema. Por una parte muestra la memoria física y por otra la swap, también muestra la memoria caché y de buffer consumida por el Kernel. A su vez, **nmon** es un sistema de monitoreo para Linux y AIX que nos permite ver en pantalla los diferentes indicadores de nuestro sistema, en el caso de teclear la opción *m*, el uso de la memoria RAM.

Los resultados dados por ambos programas son coherentes. En el caso de la MV2:

- Memoria total:  $749.2 \times 1024 = 767180 \text{ bytes}$  (~767168 bytes). Lo mismo ocurre para la memoria libre, la caché y los buffers.
- Porcentaje de Memoria usada:  $724528 / 767168 = 94.4\%$ , es decir, el porcentaje de memoria libre es  $5.6\%$  (~5.4%).

Además se puede observar que la memoria de ambas máquinas virtuales es del mismo tamaño, usando más capacidad la máquina virtual 1 lo que puede deberse al despliegue de la base de datos. Análogamente, ambos PC's tienen el mismo tamaño de memoria, y el PC1 usa una mayor cantidad. Esto puede ser por el despliegue de aplicaciones ya que en el PC2 simplemente está ejecutándose la máquina virtual.

**Pruebe a ejecutar un pago “de calentamiento” por cada uno de los métodos anteriores y verifique que funcionan a través de la página testbd.jsp.**

1. Pago a través de P1-base:



2. Pago a través de P1-ejb-cliente-remoto:



### 3. Pago a través de P1-ws:

**Pago con tarjeta**

**Proceso de un pago**

Id Transacción:

Id Comercio:

Importe:

Numero de visa:

Titular:

Fecha Emisión:

Fecha Caducidad:

CVV2:

Modo debug: ☐ True ☒ False

Direct Connection: ☐ True ☒ False

Use Prepared: ☐ True ☒ False

### 4. Comprobación de los 3 pagos al comercio 1:

**Pago con tarjeta**

Lista de pagos del comercio 1

idTransaccion	Importe	codRespuesta	idAutorizacion
1	1.0	000	1
2	2.0	000	8
3	3.0	000	9

[Volver al comercio](#)

### 5. Eliminación de los pagos:

**Pago con tarjeta**

Se han borrado 3 pagos correctamente para el comercio 1

[Volver al comercio](#)

Prácticas de Sistemas Informáticos II

### 6. Comprobación de la eliminación de los pagos:

**Pago con tarjeta**

Lista de pagos del comercio 1

idTransaccion	Importe	codRespuesta	idAutorizacion
---------------	---------	--------------	----------------

[Volver al comercio](#)

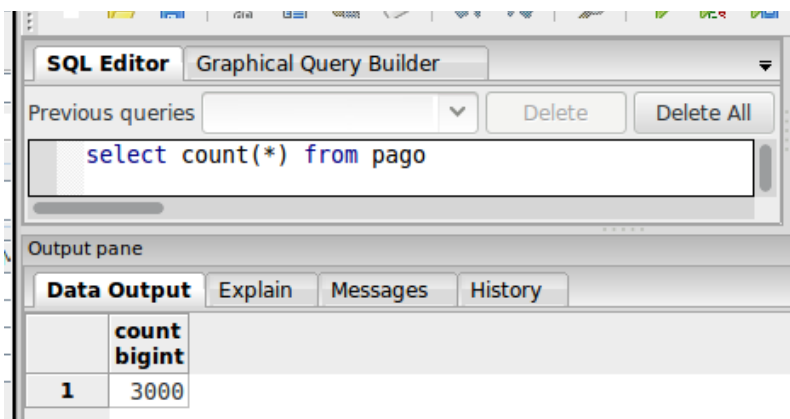
Prácticas de Sistemas Informáticos II

## Ejercicio número 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;`



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

Una vez que los resultados han sido satisfactorios:

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

sampler_label	aggregate_report_count	average	aggregate_report_median	aggregate_report_90%_line	aggregate_report_min	aggregate_report_max	aggregate_report_error%	aggregate_report_rate	aggregate_report_bandwidth
P1-base	1000	9	10	11	7	36	0.0	99.52229299363057	127.68020143934116
P1-ws	1000	58	57	70	39	140	0.0	16.88561683158286	21.861481612618622
P1-ejb	1000	25	24	31	15	68	0.0	39.18188229762558	51.339821404082755
Total	3000	31	24	61	7	140	0.0	31.077777317366262	40.27582822276551

- 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 mejor resultado es el de P1-base ya que tiene los valores más bajos de tiempo de procesamiento (media, mediana, mínimo y máximo). Además, vemos que el rendimiento ha sido mucho mayor que en las dos siguientes pruebas, por tanto, es ésta columna la que da mayor información. La columna del percentil 90 también podría ser de gran ayuda para ver la desviación de los datos.

Repita la prueba de P1-ejb (inhabilite los 'Thread Group' P1-base y P1-ws) con el EJB local incluido en P1-ejb-servidor-remoto. Para ello, cambie su 'HTTP Request', estableciendo su 'Server Name or IP' a 10.X.Y.1 (VM1) y su 'Path' a 'P1-ejb-cliente/procesapago'. Compare los resultados obtenidos con los anteriores.

sampler_label	aggregate_report_count	average	aggregate_report_median	aggregate_report_90%_line	aggregate_report_min	aggregate_report_max	aggregate_report_error%	aggregate_report_rate	aggregate_report_bandwidth
P1-ejb	1000	3	3	4	2	21	0.0	274.34842249657066	355.4545503257888
Total	1000	3	3	4	2	21	0.0	274.34842249657066	355.4545503257888

Observamos que los tiempos de procesamiento han disminuido considerablemente, casi todos en un factor 7, justamente la cantidad por la que se ha multiplicado el rendimiento.

## Ejercicio número 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.xml. 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.

Para adaptar la configuración hemos cambiado el modo a servidor y hemos establecido establecido los valores de memoria mínimo y máximo asignados:

<input checked="" type="checkbox"/>	-server
<input type="checkbox"/>	-DANTLR_USE_DIRECT_CLASS_LOADING=true
<input type="checkbox"/>	-Dfelix.fileinstall.bundles.startTransient=true
<input type="checkbox"/>	-Djavax.net.ssl.trustStore=\${com.sun.aas.instanceRoot}/config/cacerts.jks
<input type="checkbox"/>	-Dosgi.shell.telnet.ip=127.0.0.1
<input type="checkbox"/>	-Dfelix.fileinstall.log.level=2
<input type="checkbox"/>	-XX:+UnlockDiagnosticVMOptions
<input type="checkbox"/>	-Djava.security.auth.login.config=\${com.sun.aas.instanceRoot}/config/login.conf
<input type="checkbox"/>	-Dfelix.fileinstall.disableConfigSave=false
<input type="checkbox"/>	-Djava.awt.headless=true
<input checked="" type="checkbox"/>	-Xmx512m
<input type="checkbox"/>	-Djdbc.drivers=org.apache.derby.jdbc.ClientDriver
<input type="checkbox"/>	-Dosgi.shell.telnet.port=6666
<input type="checkbox"/>	-Dosgi.shell.telnet.maxconn=1
<input type="checkbox"/>	-Djava.ext.dirs=\${com.sun.aas.javaRoot}/lib/ext\${path.separator}\${com.sun.aas.javaRoot}/jre/lib/ext\${path.separator}\${com.sun.aas.instanceRoot}/lib/ext
<input type="checkbox"/>	-Djava.security.policy=\${com.sun.aas.instanceRoot}/config/server.policy
<input type="checkbox"/>	-Dgosh.args=--nointeractive
<input checked="" type="checkbox"/>	-Xms512m

Además, eliminamos el despliegue automático y recarga automática de aplicaciones y activamos la precompilación de JSPs:

**Reload:** ☐ **Enabled**  
Enables dynamic reloading of applications.

**Reload Poll Interval:**  **Seconds**  
Frequency for checking reload requests.

**Admin Session Timeout:**  **Minutes**  
A value of 0 means the session never times out.

#### Auto Deploy Settings

**Auto Deploy:** ☐ **Enabled**  
Automatically deploys applications in the autodeploy directory.

**Auto Deploy Poll Interval:**  **Seconds**  
Frequency at which the autodeploy directory is checked for applications; interval.

**Auto Deploy Retry Timeout:**  **Seconds**  
Time to report failure after a file remains stable in size but cannot be opened.

**Auto Deploy Directory:**   
Directory to monitor for autodeploy applications.

**XML Validation:**   
Type of deployment descriptor validation.

**Verifier:** ☐ **Enabled**  
Performs detailed verification before deployment.

**Precompile:** ☒ **Enabled**  
Precompiles JSPs, deploys only resulting class files.

Por último, activamos un nivel de motorización adecuado:

**Monitoring Service:** ☒ **Enabled**  
Enable monitoring for GlassFish Server

**Monitoring MBeans:** ☒ **Enabled**  
Deploy all MBeans needed for monitoring

Component Level Settings (16)		
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Level: <input type="text"/> <a href="#">Change Level</a>
Select	Module	Monitoring Level
<input type="checkbox"/>	Jvm	HIGH
<input type="checkbox"/>	Transaction Service	OFF
<input type="checkbox"/>	Connector Service	OFF
<input type="checkbox"/>	Jms Service	OFF
<input type="checkbox"/>	Security	OFF
<input type="checkbox"/>	Web Container	HIGH
<input type="checkbox"/>	Jersey(RESTful Web Services)	OFF
<input type="checkbox"/>	Web Services Container	OFF
<input type="checkbox"/>	Java Persistence	OFF
<input type="checkbox"/>	Jdbc Connection Pool	HIGH
<input type="checkbox"/>	Thread Pool	HIGH
<input type="checkbox"/>	Ejb Container	OFF
<input type="checkbox"/>	ORB (Object Request Broker)	OFF
<input type="checkbox"/>	Connector Connection Pool	OFF
<input type="checkbox"/>	Deployment	OFF
<input type="checkbox"/>	Http Service	HIGH

Revisar el script si2-monitor.sh e indicar los mandatos asadmin que debemos ejecutar en el Host PC1 para averiguar los valores siguientes, mencionados en el Apéndice 1, del servidor P1VM1:

**1. Max Queue Size del Servicio HTTP**

```
asadmin --host 10.1.4.1 --user admin --passwordfile ./passwordfile get -m server.http-service.connection-pool.max-pending-count
```

**2. Maximum Pool Size del Pool de conexiones a nuestra DB**

```
asadmin --host 10.1.4.1 --user admin --passwordfile ./passwordfile get -m domain.resources.jdbc-connection-pool.myjdbc_oracle-pool
```

## Ejercicio número 5:

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

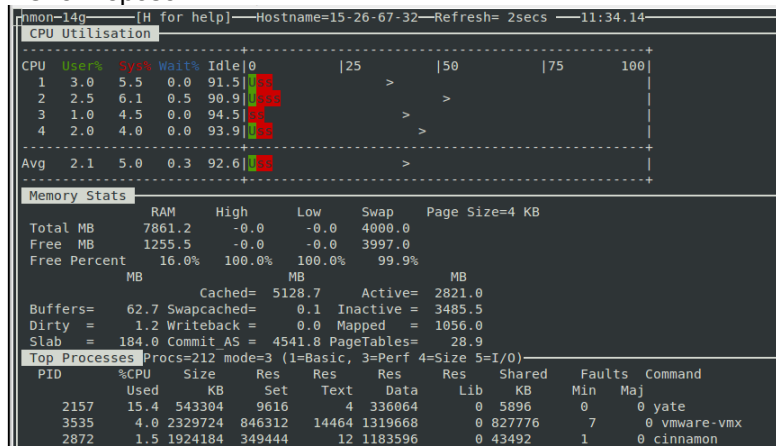
- Valores del heap de memoria que utiliza la máquina virtual Java.
  - Máximo: 512 MB
  - Mínimo: 512 MB
- Máximo número de conexiones a procesar simultáneamente por el servidor web: 5
- Tamaño máximo de la cola de conexiones pendientes de servicio: 4096
- Máximo número de sesiones en el contenedor Web: -1
- Máximo número de conexiones en pools JDBC: 32

## Ejercicio número 6:

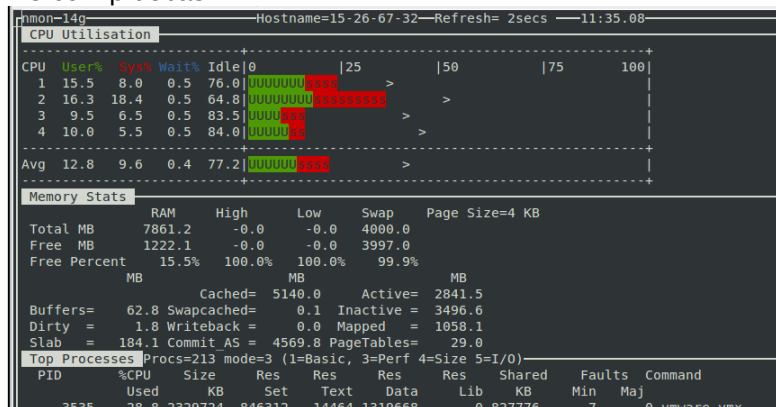
Tras habilitar la monitorización en el servidor, repita la ejecución del plan de pruebas anterior. Durante la prueba, vigile cada uno de los elementos de monitorización descritos hasta ahora. Responda a las siguientes cuestiones:

- 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,...)

PC en reposo:



PC con pruebas:



MV en reposo:



hmon-12f-----					Hostname=si2srv02-----					Refresh= 2secs -----					03:33.08-----				
CPU Utilisation																			
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
CPU	User%	Sys%	Wait%	Idle%	0	25	50	75	100										
1	1.0	0.0	0.0	99.0						>									
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
Memory Stats																			
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
	RAM	High	Low	Swap															
Total MB	749.2	0.0	749.2	150.0															
Free MB	125.9	0.0	125.9	150.0															
Free Percent	16.8%	0.0%	16.8%	100.0%															
	MB		MB	MB															
Buffers=	16.9	Cached=	168.0	Active=	481.0														
Dirty =	0.1	Swapcached=	0.0	Inactive =	120.2														
Slab =	12.9	Writeback =	0.0	Mapped =	27.9														
		Commit_AS =	1332.8	PageTables=	1.7														
Network I/O																			
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
I/F Name	Recv=KB/s	Trans=KB/s	packin	packout	insize	outsize	Peak->	Recv	Trans										
lo	0.0	0.0	0.0	0.0	0.0	0.0	0.7	0.7											
eth0	0.0	0.0	0.0	0.0	0.0	0.0	0.6	0.7											
eth1	4.6	0.2	68.3	0.5	69.6	498.0	138.6	256.5											
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
Top Processes Procs=81 mode=3 (1=Basic, 3=Perf 4=Size 5=I/O)																			
PID	%CPU	Size	Res	Res	Res	Res	Shared	Faults	Command										
	Used	KB	Set	Text	Data	Lib	KB	Min	Maj										
1014	0.5	45060	1500	4396	0	1016	644	0	0	postgres									

MV con pruebas:

```
hmon-12f-----[H for help]-----Hostname=si2srv02-----Refresh= 2secs -----03:27.01-----
```

```
CPU Utilisation
```

CPU	User%	Sys%	Wait%	Idle%	0	25	50	75	100
1	91.3	6.2	0.0	2.6					

```
Memory Stats
```

	RAM	High	Low	Swap
Total MB	749.2	0.0	749.2	150.0
Free MB	146.7	0.0	146.7	150.0
Free Percent	19.6%	0.0%	19.6%	100.0%

	MB		MB		MB
Buffers=	16.5	Cached=	163.5	Active=	465.0
Dirty =	0.2	Swapcached=	0.0	Inactive =	116.3
Slab =	12.8	Writeback =	0.0	Mapped =	27.9
		Commit_AS =	1321.2	PageTables=	1.7

```
Network I/O
```

I/F Name	Recv=KB/s	Trans=KB/s	packin	packout	insize	outsize	Peak->	Recv	Trans
lo	0.0	0.0	0.0	0.0	0.0	0.0	0.7	0.7	
eth0	0.0	0.0	0.0	0.0	0.0	0.0	0.6	0.7	
eth1	121.6	216.5	624.8	566.4	199.2	391.4	121.6	216.5	

```
Top Processes Procs=81 mode=3 (1=Basic, 3=Perf 4=Size 5=I/O)
```

PID	%CPU	Size	Res	Res	Res	Shared	Faults	Command	
	Used	KB	Set	Text	Data	Lib	KB	Min	Maj
1372	96.9	970812	416304	4	0	947060	18520	241	0 java

Como podemos observar, el elemento más utilizado es la CPU especialmente al principio de la ejecución de pruebas, cuando la anterior imagen fue capturada. También la red se utiliza bastante mientras que el tamaño de memoria utilizada durante las pruebas no es significativo.

### ● ¿Le parece una situación realista la simulada en este ejercicio? ¿Por qué?

La situación simulada en el ejercicio no es realista ya que estamos en un entorno controlado, los laboratorios, con unas características fijadas de red. Además, el número de pruebas realizadas, 1000, no es suficiente para obtener un resultado fiable.

### ● Teniendo en cuenta cuál ha sido el elemento más saturado, proponga otro esquema de despliegue que resuelva esa situación.

Teniendo en cuenta que la CPU es el elemento más saturado, habría que aumentar la capacidad de procesamiento de la máquina virtual, habilitando varios procesadores en vez de uno solo.

## Ejercicio número 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.



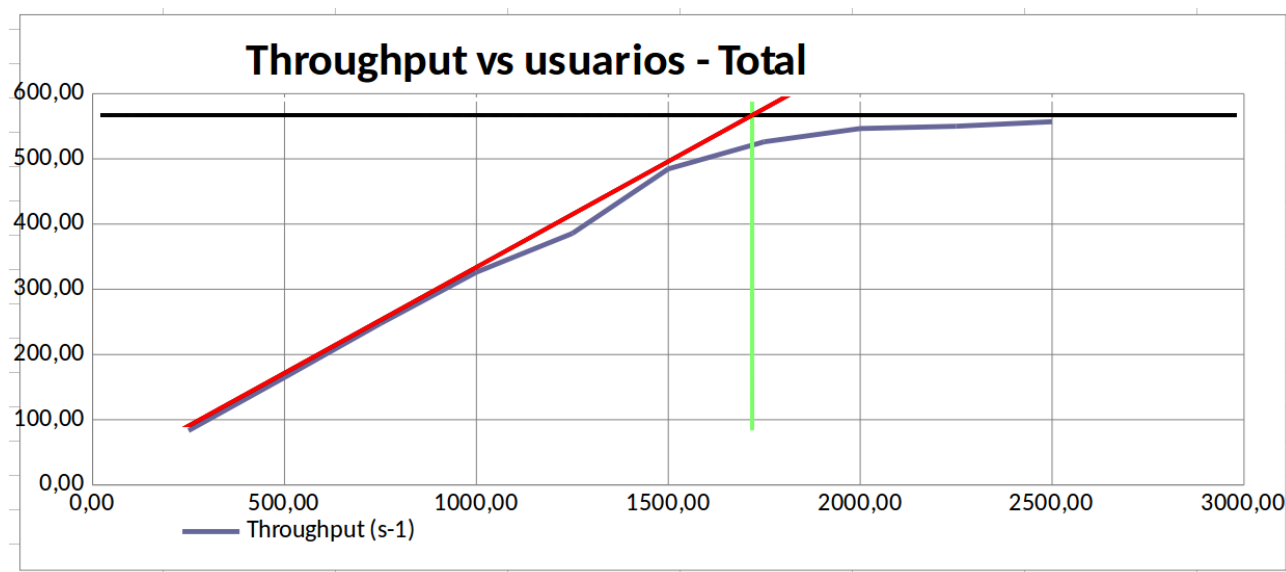
## Ejercicio número 8:

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 acción 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 saturan, 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.

Incluir el fichero P2-curvaProductividad.jmx en la entrega.



¿Cómo debemos invocar a nmon para recolectar muestras cada 5 segundos durante 10 minutos que incluyan los 'top processes' en el fichero log-file.nmon?

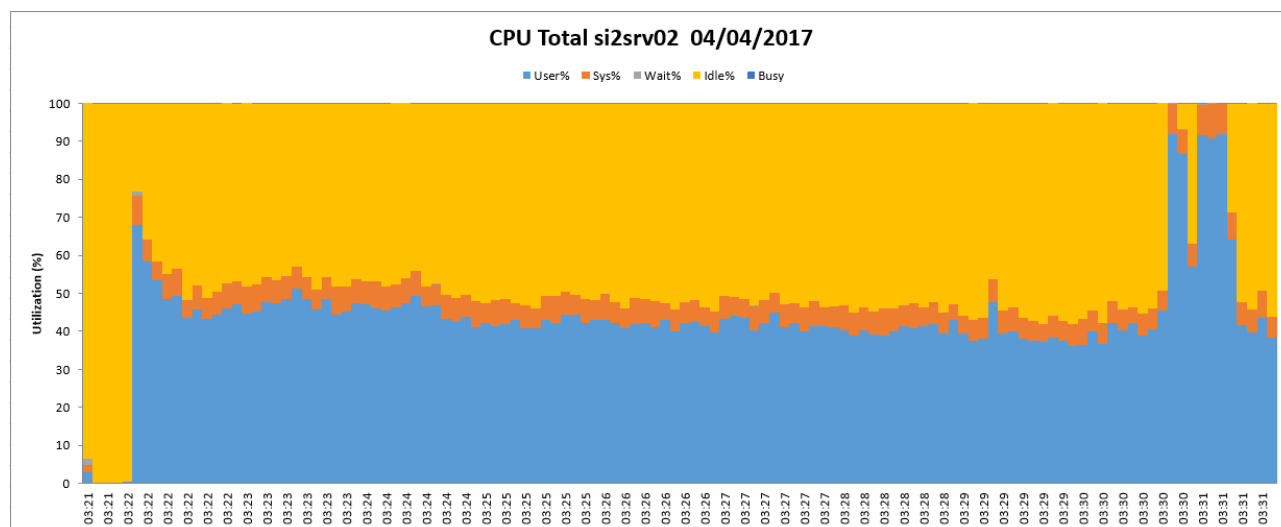
```
mon -F log-file.nmon -t -s 5 -c 120
```

Para invocar el comando se usan los siguientes parámetros:

- -F <filename> para indicar el fichero donde se volcarán los datos
- -t para que se incluyan los Top porcesses en el ficheros
- -s <seconds> para indicar el tiempo entre muestras
- -c <times> para indicar el número de muestras que se deben tomar (120 muestras \* 5 segundos = 10 minutos)

Ejecutar el mandato anterior en PC1VM1 durante una de las pruebas de P1-ws y usar nmon Analyser ([https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Power+Systems/page/nmon\\_analyser](https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Power+Systems/page/nmon_analyser)) para generar un fichero Excel con los resultados. Revisadlo y comentad algún aspecto relevante.

El fichero obtenido se adjunta en la práctica como log-file.nmon.xlsx. En él podemos observar que las operaciones en disco no son destacables en número mientras que el uso del procesador se ve incrementado cuando se ejecutan más de una prueba a la vez.



## Ejercicio número 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.

El punto de saturación se produce para unos 1.700 usuarios, marcado en la gráfica anterior con una línea verde. En este punto el throughput es aproximadamente 525 peticiones/s.

El throughput máximo que se obtiene es de unas 575 peticiones/s.

- 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.

Como visto en anteriores ejercicios, el parámetro limitante del servidor es la capacidad de procesamiento. Por tanto, para obtener el punto de saturación en un número mayor de usuarios habría que aumentar tal capacidad.

- 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.

Hemos dotado a la máquina virtual donde se despliega el servidor con 4 procesadores en vez de uno. Como consecuencia, el throughput no mejora excesivamente pero el tiempo de respuesta baja muy considerablemente, del orden de 20 veces mejor, debido a que los mensajes no esperan tanto en la cola.

Etiqueta	# Muestras	Media	Mediana	Línea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
P1-base/comie...	170000	111	53	164	1	5201	3,23%	587,8/sec	1138,2
P1-base/proce...	170000	117	58	169	2	5199	3,09%	587,5/sec	723,6
Total	340000	114	56	166	1	5201	3,16%	1170,6/sec	1854,2

Para aumentar la intensidad de la prueba hemos bajado el thinking time a 0,5-1 segundo y aumentado el número de bucles a 100 doblando las peticiones que se procesan por segundo (throughput) a 1170.