		<b>Escuela Politécnica Superior Ingeniería Informática Práctica de Sistemas Informáticos 1</b>			
<b>Grupo</b>	<b>2323</b>	<b>Práctica</b>	<b>2</b>	<b>Fecha</b>	<b>6/03/2023</b>
<b>Alumno/a</b>		Hidalgo, Gamborino, Sergio			
<b>Alumno/a</b>		Ibáñez, González, Miguel			

# Práctica 1: Arquitectura de JAVA EE

## 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, P1-ejb). Adjunte el fichero generado P2.jmx al entregable de la práctica

Para realizar este ejercicio descargamos apache Jmeter abrimos la aplicación y configuramos el archivo P2test.jmx según nos explica la práctica:

```
miguel@miguel-System-Product-Name:~$ apache-jmeter-5.5/bin/jmeter.sh
=====
Don't use GUI mode for load testing !, only for Test creation and Test debugging
.
For load testing, use CLI Mode (was NON GUI):
  jmeter -n -t [jmx file] -l [results file] -e -o [Path to web report folder]
& increase Java Heap to meet your test requirements:
  Modify current env variable HEAP="-Xms1g -Xmx1g -XX:MaxMetaspaceSize=256m" in
  the jmeter batch file
Check : https://jmeter.apache.org/usermanual/best-practices.html
=====
```

Dentro de P2test añadimos:

- HTTP Request Defaults
- User Defined Variables
- Crear P1-base, dentro añadimos:
  - Random Variable
  - Counter
  - CSV Data Set Config
  - HTTP Request
- Crear P1-ws, dentro añadimos:

- Lo mismo que en P1-base cambiando el idTransaccion, el idComercio, el nombre de la petición y la ruta.
- Crear P1-ejb, dentro añadimos:
  - Lo mismo que en P1-base cambiando el idTransaccion, el idComercio, el nombre de la petición y la ruta.

Después medimos los resultados con Add->listener->AggregateReport

Para comprobar el correcto funcionamiento se añade Add->listener->View Results Tree

## ***Ejercicio 2:***

**Preparar el PC con el esquema descrito en la Figura 22. Para ello:**

- **Anote en la memoria de prácticas las direcciones IP asignadas a las máquinas virtuales y al PC**
- **Detenga el servidor de GlassFish del PC host**
- **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.**
- **Revise y modifique si es necesario el fichero glassfish-web.xml, para indicar la IP del EJB remoto que usa P1-ejb-cliente.**
- **Despliegue las siguientes prácticas: P1-base, P1-ws, P1-ejb-servidor-remoto y P1-jeb-clienteremoto, con el siguiente esquema:**
  - o **El destino de despliegue de la aplicación P1-base será PC2VM con IP 10.X.Y.2 (as.host)**
  - o **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 de P1-ws y as.host de P1-ejb-cliente-remoto)**
  - o **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 de P1-ws y as.host.server y as.host.client de P1- ejb-servidor-remoto)**
  - o **La base de datos en todos ellos será la de PC1VM con IP 10.X.Y.1 (db.host)**

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 en el PC host. Anote sus comentarios en la memoria. Pruebe a ejecutar un pago “de calentamiento” por cada uno de los métodos anteriores y verifique que funciona a través de la página testbd.jsp

Se han tenido que cambiar los nombres de las aplicaciones, en la P1-base cambiar la variable nombre de P1 a P1-base, en las otras ya tienen dicho nombre.

Las IPs se han modificado según el enunciado:

P1-base “build.properties”:

```
build.properties X
P1-base > build.properties
1  # Propiedades de despliegue
2  nombre=P1-base
3  build=${basedir}/build
4
5
6  dist=${basedir}/dist
7
8
9  src=${basedir}/src
10
11
12 web=${basedir}/web
13
14
15 paquete=ssii2
16 war=${nombre}.war
17
18
19
20 asadmin=${as.home}/bin
21 as.home=${env.J2EE_HOME}
22 as.lib=${as.home}/lib
23 as.user=admin
24 as.host=10.6.4.2
```

P1-base “postgresql.properties”:

```
postgresql.properties X build.pr
P1-base > postgresql.properties
1  # Propiedades de la BD
2
3  # Parametros propios de la BD
4  db.name=visa
5  db.user=alumnodb
6  db.password=****
7  db.port=5432
8  db.host=10.6.4.1
9  # Recursos y pools asociados
10 db.pool.name=VisaPool
11 db.jdbc.resource.name=jdbc:postgresql:
12 db.url=jdbc:postgresql:
13 db.client.host=10.6.4.2
14 db.client.port=4848
15
```

P1-ws "build.properties":

```
build.properties X
1-ws > build.properties
1 # Propiedades de despliegue
2 nombre=P1-ws
3 build=${basedir}/build
4 build.client=${build}/client
5 build.server=${build}/server
6 dist=${basedir}/dist
7 dist.client=${dist}/client
8 dist.server=${dist}/server
9 src=${basedir}/src
10 src.client=${src}/client
11 src.server=${src}/server
12 web=${basedir}/web
13 conf=${basedir}/conf
14 conf.serverws=${conf}/serverws
15 paquete=ssii2
16 war=${nombre}.war
17 wswar=${nombre}-ws.war
18 jar=${nombre}.jar
19 tmpvisaclientjar=${nombre}-ws-client.jar
20 asadmin=${as.home}/bin/asadmin
21 as.home=${env.J2EE_HOME}
22 as.lib=${as.home}/lib
23 as.user=admin
24 as.host.client=10.6.4.2
25 as.host.server=10.6.4.1
```

P1-ws "postgresql.properties":

```
postgresql.properties X build.properties
1-ws > postgresql.properties
1 # Propiedades de la BD postgresql
2
3 # Parametros propios de la BD
4 db.name=visa
5 db.user=alumnodb
6 db.password=****
7 db.port=5432
8 db.host=10.6.4.1
9
10 # Recursos y pools asociados
11 db.pool.name=VisaPool
12 db.jdbc.resource.name=jdbc:postgresql://10.6.4.1:5432/visa
13 db.url=jdbc:postgresql://10.6.4.1:5432/visa
14 db.client.host=10.6.4.2
15 db.client.port=4848
```

P1-ws "web/WEB-INF/web.xml" en la ip de "<param-value>":

```
web.xml X
P1-ws > web > WEB-INF > web.xml
14 getServletContext().getInitParameter("nombre");
15
16 donde "nombre se corresponde con el elemento <param-name>
17 de uno de estos parámetros de inicialización.
18
19 Se puede definir cualquier número de parámetros de inicialización,
20 incluyendo cero parámetros.
21 -->
22
23 <context-param>
24 <param-name>webmaster</param-name>
25 <param-value>http://10.6.4.1:8080/P1-ws-ws/VisaDAOWSService</param-value>
26 </context-param>
27
28
```

P1-ejb-cliente-remoto "build.properties":

```
build.properties P1-ejb-cliente-remoto
P1-ejb-cliente-remoto > ⚙ build.properties
1  # Propiedades de despliegue
2  nombre=P1-ejb
3  build=${basedir}/build
4  |
5  dist=${basedir}/dist
6  |
7  src=${basedir}/src
8  |
9  web=${basedir}/web
10 |
11 paquete=ssii2
12 war=${nombre}.war
13 |
14 asadmin=${as.home}/bin
15 as.home=${env.J2EE_HOME}
16 as.lib=${as.home}/lib
17 as.user=admin
18 as.host=10.6.4.2
```

P1-ejb-cliente-remoto "postgresql.properties":

```
P1-ejb-cliente-remoto > ⚙ postgresql.properties
# Propiedades de la BD postgresql

# Parametros propios de postgresql
db.name=visa
db.user=alumnodb
db.password=****
db.port=5432
db.host=10.6.4.1
# Recursos y pools asociados
db.pool.name=VisaPool
db.jdbc.resource.name=jdbc/Visa
db.url=jdbc:postgresql://${db.host}:${db.port}/${db.name}
db.client.host=10.6.4.1
db.client.port=4848
```

P1-ejb-cliente-remoto "web/WEB-INF/glassfish-web.xml" en la ip de "<jndi-name>":

```
glassfish-web.xml X
P1-ejb-cliente-remoto > web > WEB-INF > ⚙ glassfish-web.xml
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE glassfish-web-app PUBLIC "-//GlassFish.org/DTD/
3  <glassfish-web-app>
4  |   <ejb-ref>
5  |       <ejb-ref-name>VisaDAOBean</ejb-ref-name>
6  |       <jndi-name>corbaname:iiop:10.6.4.1:3700#java:comp/env/ejb/VisaDAOBean</jndi-name>
7  |   </ejb-ref>
8  </glassfish-web-app>
```

P1-ejb-servidor-remoto “build.properties”:

```
build.properties X
b-servidor-remoto > build.properties

# Propiedades de despliegue
nombre=P1-ejb
build=${basedir}/build
build.client=${build}/client
build.server=${build}/server
dist=${basedir}/dist
dist.client=${dist}/client
dist.server=${dist}/server
src=${basedir}/src
src.client=${src}/client
src.server=${src}/server
web=${basedir}/web
conf=${basedir}/conf
conf.server=${conf}/server
conf.application=${conf}/app
paquete=ssii2
war=${nombre}-cliente.war
jar=${nombre}.jar
ear=${nombre}.ear
asadmin=${as.home}/bin/asadm
as.home=${env.J2EE_HOME}
as.lib=${as.home}/lib
as.user=admin
as.host.client=10.6.4.1
as.host.server=10.6.4.1
```

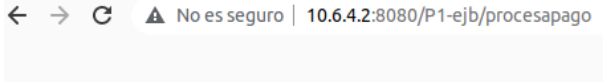
P1-ejb-servidor-remoto “postgresql.properties”:

```
postgresql.properties X
-ejb-servidor-remoto > postgresql.prope
1 # Propiedades de la BD post
2
3 # Parametros propios de po
4 db.name=visa
5 db.user=alumnodb
6 db.password=****
7 db.port=5432
8 db.host=10.6.4.1
9 # Recursos y pools asociad
10 db.pool.name=VisaPool
11 db.jdbc.resource.name=jdbc
12 db.url=jdbc:postgresql://$
13 db.client.host=10.6.4.1
```

P1-ejb-servidor-remoto “web/WEB-INF/web.xml” en la ip de “<param-value>”:

```
web.xml X
P1-ejb-servidor-remoto > web > WEB-INF > web.xml
12
13 String value =
14     getServletContext().getInitParameter("nombre");
15
16     donde "nombre se corresponde con el elemento <param-name>
17     de uno de estos parámetros de inicialización.
18
19     Se puede definir cualquier número de parámetros de inicialización,
20     incluyendo cero parámetros.
21     -->
22
23     <context-param>
24         <param-name>webmaster</param-name>
25         <param-value>http://10.6.4.1:8080/P1-ws-ws/VisaDAOWSService</param-value>
26     </context-param>
27
```

Todo funciona correctamente al ser accesibles las 3 páginas con los nombres propuestos por la práctica y en la IP 10.6.4.2:



### Pago con tarjeta

Numero de visa:

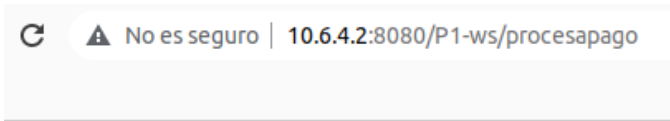
Campo vacío

Titular:

Fecha Emisión:

Fecha Caducidad:

CVV2:

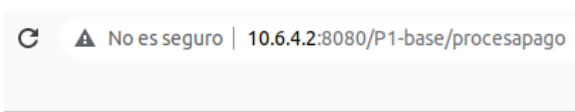


### Pago con tarjeta

de visa:

Emisión:

Caducidad:



### Pago con tarjeta

de visa:

Emisión:

Caducidad:

Estando los servicios de ejb y ws desplegados en la ip 10.6.4.1:

<https://10.6.4.1:4848/common/index.jsf#>

Server: 10.6.4.1

Source Edition

### Applications

Applications can be enterprise or web applications, or various kinds of modules. Restart an application or module by clicking on the reload icon.

Deployed Applications (2)

☒

Filter:

Select	Name	Deployment Order	Enabled
<input type="checkbox"/>	P1-ejb	100	✓
<input type="checkbox"/>	P1-ws-ws	100	✓

Memoria mostrada por el comando free en la máquina virtual del ordenador 10.6.4.1:

```

P1-ejb-
seregio
plicatio
seregio
plicatio
P1-base
seregio
plicatio
i/si2/SI
atagen/
1-base/
1-ws/
2-curve

```

si2srv - VMware Workstation 17 Player (Non-commercial use only)
File Virtual Machine Help

```

si2@si2srv01:~$ free
              total        used        free      shared    buffers     cached
Mem:           767168      689808        77360           0        41472      217712
-/+ buffers/cache:      430624      336544
Swap:          153592           0        153592
si2@si2srv01:~$

```

Memoria mostrada por el comando free en la máquina virtual del ordenador 10.6.4.2:

```

sergio@
seregio
lassfis
tions/
loy/
seregio
lassfis
tions/
seregio
lassfis
seregio

```

si2srv - VMware Workstation 17 Player (Non-commercial use only)
File Virtual Machine Help

```

si2@si2srv02:~$ free
              total        used        free      shared    buffers     cached
Mem:           767168      597200      169968           0        34520      171124
-/+ buffers/cache:      391556      375612
Swap:          153592           0        153592
si2@si2srv02:~$

```

Ram mostrada por nmon en la máquina virtual del ordenador 10.6.4.1:

```

P1-ejb-
seregio
plicatio
seregio
plicatio
P1-base
seregio
plicatio
i/si2/SI
atagen/
1-base/
1-ws/
2-curve
2.jmx
seregio@s

```

si2srv - VMware Workstation 17 Player (Non-commercial use only)
File Virtual Machine Help

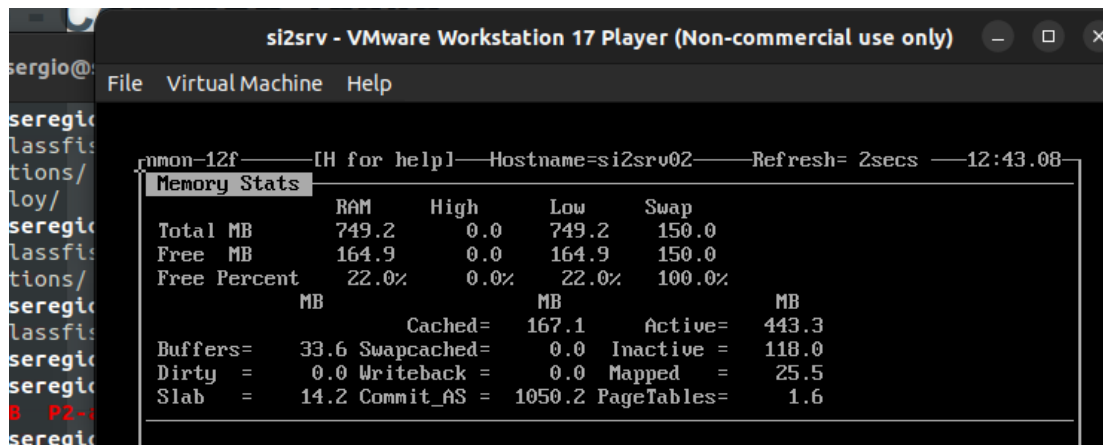
```

nmon-12f [H for help] Hostname=si2srv01 Refresh= 2secs
Memory Stats
RAM      High      Low      Swap
Total MB  749.2    0.0      749.2    150.0
Free MB   73.8    0.0      73.8     150.0
Free Percent  9.9%    0.0%    9.9%    100.0%
MB
Cached= 212.6 Active= 501.7
Buffers= 40.4 Swapcached= 0.0 Inactive = 149.3
Dirty = 0.0 Writeback = 0.0 Mapped = 26.8
Slab = 15.0 Commit_AS = 1125.4 PageTables= 1.6

```

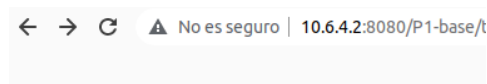
Ram mostrada por nmon en la máquina virtual del ordenador 10.6.4.2:





Como se puede ver en el output de ambos programas la ip 10.6.4.1, que aloja la base de datos y el servicio, usa más memoria que la ip 10.6.4.2, que aloja el cliente. Pese a esto, la memoria liberada es menor en la primera debido a que no se puede liberar los recursos del servicio, pues este debe estar siempre disponible.

Prueba de pago P1-base:



## Pago con tarjeta

### Proceso de un pago

Id Transacción:

Id Comercio:

Importe:

Numero de visa:

Titular:

Fecha Emisión:

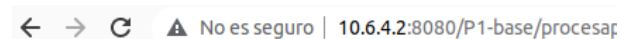
Fecha Caducidad:

CVV2:

Modo debug: ☒ True ☐ False

Direct Connection: ☒ True ☐ False

Use Prepared: ☒ True ☐ False

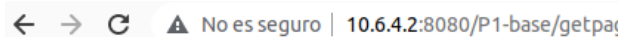


## Pago con tarjeta

Pago realizado con éxito. A continuación se muestra el comprobante

idTransaccion: 1  
idComercio: 1  
importe: 123.0  
codRespuesta: 000  
idAutorizacion: 5

[Volver al comercio](#)

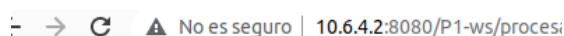


## Pago con tarjeta

Lista de pagos del comercio 1

idTransaccion	Importe	codRespuesta	idAutorizacion
1	123.0	000	5

Prueba de pago P1-ws:



## Pago con tarjeta

Pago realizado con éxito. A continuación se muestra el comprobante

# 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

Prueba de pago P1-ejb:

# Pago con tarjeta

Pago realizado con éxito. A continuación se muestra el compro

idTransaccion: 3  
idComercio: 3  
importe: 124.0  
codRespuesta:  
idAutorizacion:

# Pago con tarjeta

Lista de pagos del comercio 2

idTransaccion	Importe	codRespuesta	idAutorizacion
2	1234.0	000	8

[Volver al comercio](#)

# 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

# Pago con tarjeta

Lista de pagos del comercio 3

idTransaccion	Importe	codRespuesta	idAutorizacion
3	124.0	000	11

[Volver al comercio](#)

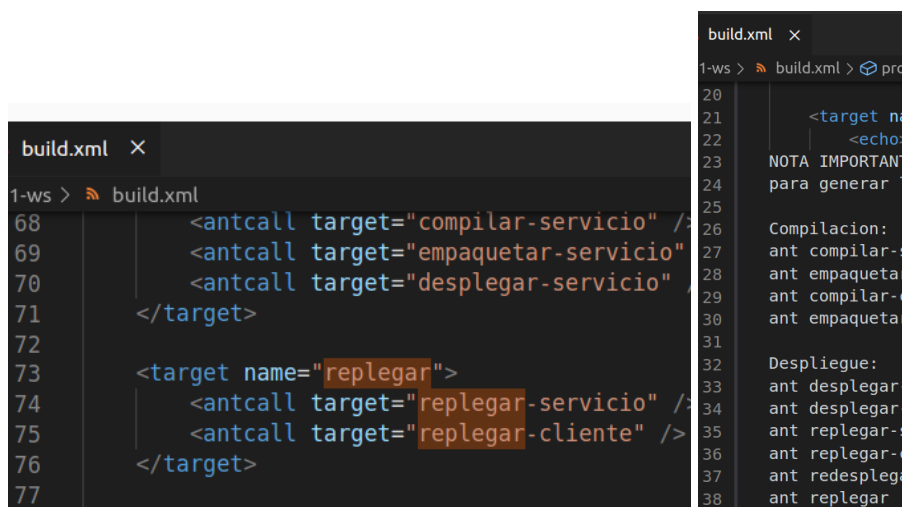
### Ejercicio 3:

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;`
- 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.
- 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? Incluir el directorio P2 en la entrega. 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. El fichero P2.jmx entregado no debe contener estos cambios, es decir, debe estar configurado para probar el EJB remoto.

Se crea un nuevo target de nombre “replegar” en el fichero “build.xml” de P1-ws, además se deberá poner en las ordenes del inicio del fichero como “ant replegar” ,



```
build.xml ×
1-ws > build.xml
68     <antcall target="compilar-servicio" />
69     <antcall target="empaquetar-servicio" />
70     <antcall target="desplegar-servicio" />
71 </target>
72
73 <target name="replegar">
74     <antcall target="replegar-servicio" />
75     <antcall target="replegar-cliente" />
76 </target>
77

build.xml ×
1-ws > build.xml > proj
20
21     <target name="replegar">
22         <echo>
23             NOTA IMPORTANT
24             para generar l
25
26         Compilacion:
27         ant compilar-s
28         ant empaquetar
29         ant compilar-c
30         ant empaquetar
31
32         Despliegue:
33         ant desplegar-
34         ant desplegar-
35         ant replegar-s
36         ant replegar-c
37         ant redesplega
38         ant replegar
```

Tras ejecutar “script-auto.bash” se ejecutan las pruebas con jmeter.

Comentarios													
Escribir todos los datos a Archivo													
Nombre de archivo								Navegar...	Log/Mostrar sólo: <input type="checkbox"/> Escribir en Log Sólo Errores <input type="checkbox"/> Éxitos <input type="button" value="Configurar"/>				
Etiqueta	# Muestras	Media	Mediana	90% Line	95% Line	99% Line	Min	Máx	% Error	Rendimiento	Kb/sec	Sent Kb/sec	
P1-base	1000	17	13	23	28	53	8	1415	0,00%	56,6/sec	72,63	0,00	
P1-ws	1000	86	78	104	117	229	49	894	0,00%	11,6/sec	14,92	0,00	
P1-ajb	1000	37	35	47	56	98	23	466	0,00%	26,2/sec	34,18	0,00	
Total	3000	47	36	87	98	135	8	1415	0,00%	21,1/sec	27,25	0,00	

El P1-base parece ser el mejor pues tiene el rendimiento más alto (puede atender 56.6 peticiones por segundo). Se ha elegido el rendimiento como columna porque se refleja mejor que servicio, a nivel práctico, sería más eficaz respondiendo solicitudes.

Para comprobar que se insertan 3000 pagos accedemos a la base de datos de la VM 10.6.4.1.

```
eth1      Link encap:Ethernet  HWaddr aa:cc:cc:cc:cc:0a
          inet addr:10.6.4.1  Bcast:10.255.255.255  Mask:255.0.0.0
          inet6 addr: fe80::a8cc:ccff:fecc:cc0a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:231509 errors:0 dropped:0 overruns:0 frame:0
          TX packets:273431 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:63742384 (63.7 MB)  TX bytes:95230577 (95.2 MB)
          Interrupt:16 Base address:0x2080

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:6681 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6681 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2656768 (2.6 MB)  TX bytes:2656768 (2.6 MB)

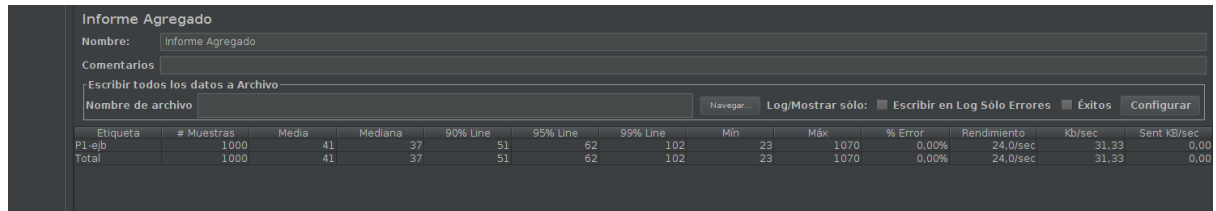
sergio@ser...$ cd ...
sergio@ser...$ cd ..
sergio@ser...$ si2@si2srv01:~$ psql visa -U alummodb
psql (8.4.10)
Type "help" for help.

visa=# select count(*) from pago;
count
-----
  3000
(1 row)

visa=#
```

Los archivos server.log se encuentran en las carpetas serverlog1 y 2 (para cada server de manera respectiva).

P1-ejb-cliente-remoto:



Informe Agregado

Nombre: Informe Agregado

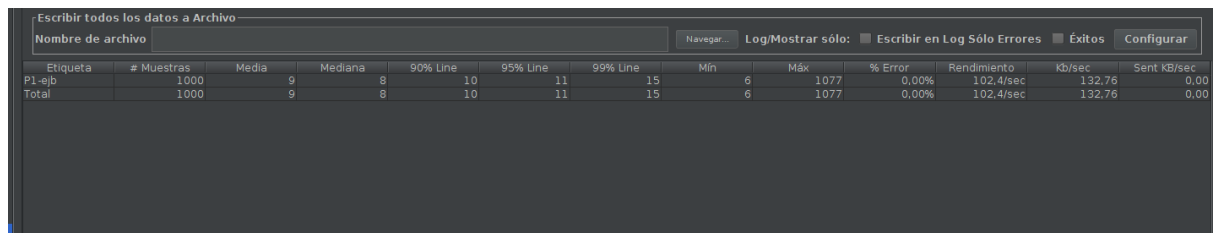
Comentarios

Escribir todos los datos a Archivo

Nombre de archivo   Log/Mostrar sólo: ☐ Escribir en Log Sólo Errores ☐ Éxitos

Etiqueta	# Muestras	Media	Mediana	90% Line	95% Line	99% Line	Min	Máx	% Error	Rendimiento	kb/sec	Sent KB/sec
P1-ejb	1000	41	37	51	62	102	23	1070	0.00%	24.0/sec	31.33	0.00
Total	1000	41	37	51	62	102	23	1070	0.00%	24.0/sec	31.33	0.00

P1-ejb-cliente:



Escribir todos los datos a Archivo

Nombre de archivo   Log/Mostrar sólo: ☐ Escribir en Log Sólo Errores ☐ Éxitos

Etiqueta	# Muestras	Media	Mediana	90% Line	95% Line	99% Line	Min	Máx	% Error	Rendimiento	kb/sec	Sent KB/sec
P1-ejb	1000	9	8	10	11	15	6	1077	0.00%	102.4/sec	132.76	0.00
Total	1000	9	8	10	11	15	6	1077	0.00%	102.4/sec	132.76	0.00

Como era de esperar, el ejb local es más rápido pues no tiene que conectarse a otro servidor (el rendimiento es mucho más alto). Respecto al resto, la diferencia también es notable por lo mismo anteriormente mencionado.

## 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**

**\$opt/glassfish4/glassfish/domains/domain1/config/domain.xml<sup>3</sup> . 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 con scp.**

**Revisar el script si2-monitor.sh e indicar los mandatos asadmin<sup>4</sup> que debemos ejecutar en el PC host para averiguar los valores siguientes, mencionados en el Apéndice 1, del servidor PC1VM1:**

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

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

**Así como el mandato para monitorizar el número de errores en las peticiones al servidor web.**

Dentro de JVM Options se elimina -client y se añade -server

Options (31)

☒
☐

Add JVM Option
Delete

Select	Value
<input type="checkbox"/>	-XX:MaxPermSize=192m
<input checked="" type="checkbox"/>	-client
<input type="checkbox"/>	-Djava.awt.headless=true
<input type="checkbox"/>	-Djdk.corba.allowOutputStreamSubclass=true

Manage JVM options for the server. View

Configuration Name: server-config

Options (31)

☒
☐

Add JVM Option
Delete

Select	Value
<input type="checkbox"/>	-Dorg.glassfish.additionalOS
<input type="checkbox"/>	-Djavax.management.builder
<input type="checkbox"/>	-Djavax.net.ssl.keyStore=
<input type="checkbox"/>	-server

La opción del valor máximo de memoria está correctamente configurada, y se añade el valor mínimo

<input type="checkbox"/>	-Xmx512m	<input type="checkbox"/>	-Xms512m
--------------------------	----------	--------------------------	----------

En la configuración de aplicaciones del dominio, se desactiva el despliegue automático (Auto Deploy) y la recarga automática (Reload), además de activar la precompilación de JSPs.

Domain Attributes
Applications Configuration
Admin

### Applications Configuration

Enable reloading so that changes to deployed applications are detected.

Load Defaults

Reload:

☐ Enabled

Enables dynamic reloading of applic

Reload Poll Interval:

Seconds

Frequency for checking reload requ

Admin Session Timeout:

Minutes

A value of 0 means the session never

---

#### Auto Deploy Settings

Auto Deploy:

☐ Enabled

Automatically deploys application

Auto Deploy Poll Interval:

Second

Frequency at which the autodepl

Auto Deploy Retry Timeout:

Second

Time to report failure after a file r

Auto Deploy Directory:

Directory to monitor for autodepl

XML Validation:

Type of deployment descriptor va

Verifier:

☐ Enabled

Performs detailed verification bef

Precompile:

☒ Enabled

Precompiles JSPs. denlovs only

Se asigna el valor HIGH en los siguientes módulos:

### Monitoring Service

Enable monitoring for a component or service by selecting either LOW or HIGH.

Configuration Name: server-config

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"/> <input type="button" value="Change Level"/>
Select	Module
<input checked="" type="checkbox"/>	Jvm
<input type="checkbox"/>	Transaction Service
<input type="checkbox"/>	Connector Service
<input type="checkbox"/>	Jms Service
<input type="checkbox"/>	Security
<input checked="" type="checkbox"/>	Web Container
<input type="checkbox"/>	Jersey(RESTful Web Services)
<input type="checkbox"/>	Web Services Container
<input type="checkbox"/>	Java Persistence
<input checked="" type="checkbox"/>	Jdbc Connection Pool
<input checked="" type="checkbox"/>	Thread Pool
<input type="checkbox"/>	Ejb Container
<input type="checkbox"/>	ORB (Object Request Broker)
<input type="checkbox"/>	Connector Connection Pool
<input type="checkbox"/>	Deployment
<input checked="" type="checkbox"/>	Http Service

El archivo "domain.xml" está guardado en la carpeta P2.

Max queue size:

Se ha utilizado el comando "ssh -o HostKeyAlgorithms=ssh-rsa si2@10.6.4.2" para obtener una shell remota y no tener que utilizar la opción -host.

"sadmin get --user admin server.thread-pools.thread-pool.http-thread-pool.\*"

```
si2@si2srv02:~$ asadmin get --user admin server.thread-pools.thread-pool.http-thread-pool.*
Enter admin password for user "admin">
server.thread-pools.thread-pool.http-thread-pool.classname=org.glassfish.grizzly.threadpool.GrizzlyExecutorService
server.thread-pools.thread-pool.http-thread-pool.idle-thread-timeout-seconds=900
server.thread-pools.thread-pool.http-thread-pool.max-queue-size=4096
server.thread-pools.thread-pool.http-thread-pool.max-thread-pool-size=5
server.thread-pools.thread-pool.http-thread-pool.min-thread-pool-size=5
server.thread-pools.thread-pool.http-thread-pool.name=http-thread-pool
Command get executed successfully.
si2@si2srv02:~$
```

El máximo es 4096 (max-thread-pool-size).

Se ha utilizado ssh: "sadmin get --user admin  
resources.jdbc-connection-pool.VisaPool.max-pool-size "

```
si2@si2srv02:~$ asadmin get --user admin resources.jdbc-connection-pool.VisaPool.
x-pool-size
Enter admin password for user "admin">
resources.jdbc-connection-pool.VisaPool.max-pool-size=32
Command get executed successfully.
si2@si2srv02:~$
```

### **Ejercicio 5:**

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

Thread Pools (3)					
		<a href="#">New...</a>	<a href="#">Delete</a>		
Select	Thread Pool ID	Max Thread Pool Size	Min Thread Pool Size	Max Queue Size	Idle Thread Timeout
<input type="checkbox"/>	admin-thread-pool	50	5	256	900
<input type="checkbox"/>	http-thread-pool	5	5	4096	900
<input type="checkbox"/>	thread-pool-1	200	5	4096	900

General	Session Properties	Manager Properties	Storage
---------	--------------------	--------------------	---------

## Manager Properties

#### Load Defaults

**Configuration Name:** server-config

**Reap Interval:**

Number of seconds until inactive

Max Sessions:

## Pool Settings

**Initial and Minimum Pool Size:**  Connections  
Minimum and initial number of connections

**Maximum Pool Size:**  Connections  
Maximum number of connections that can

**Pool Resize Quantity:**  Connections  
Number of connections to be removed when

**Idle Timeout:**  Seconds  
Maximum time that connection can remain

**Max Wait Time:**  Milliseconds  
Amount of time caller waits before connect

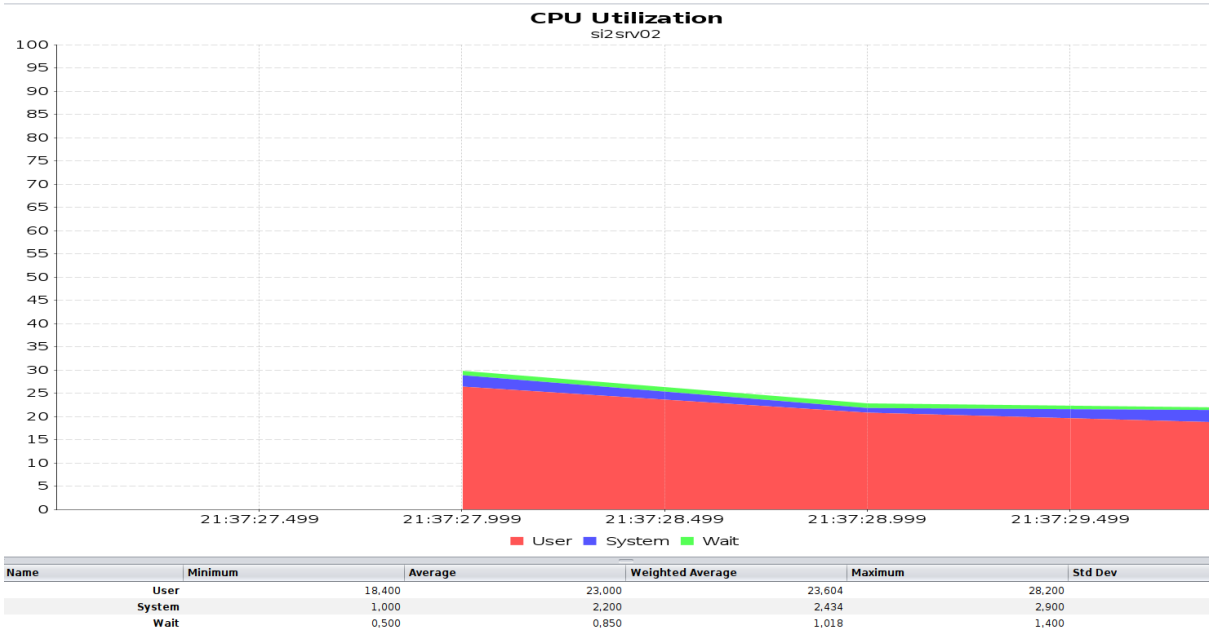


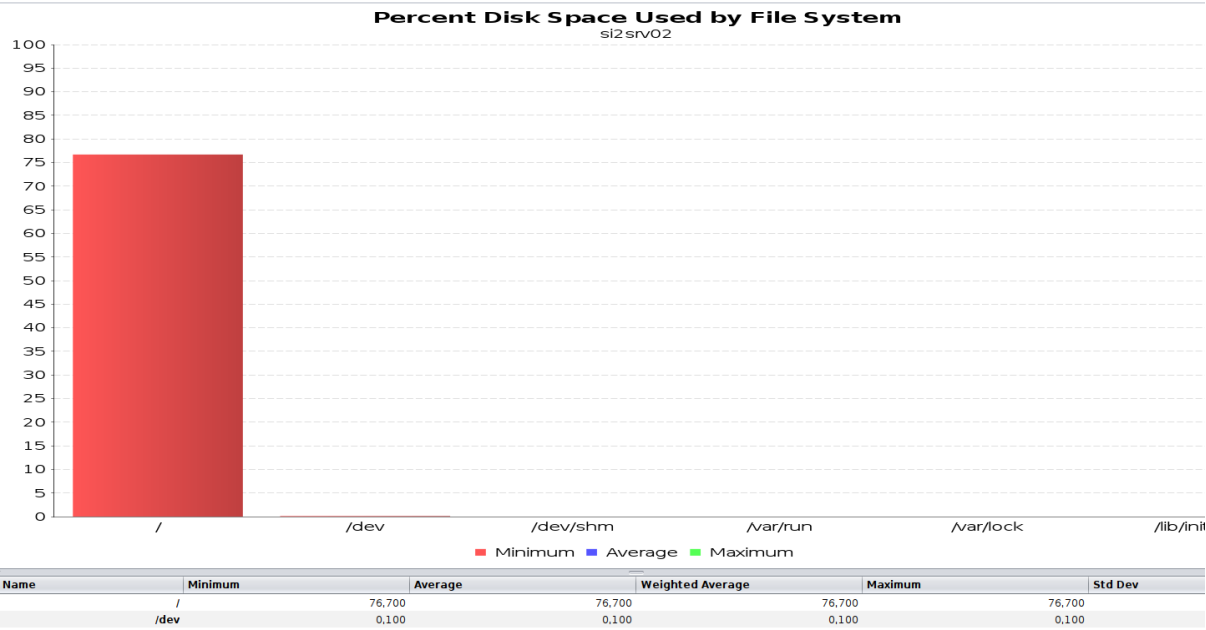
Valores máximo y mínimo del heap de memoria que utiliza la máquina virtual Java	512m ambos
Máximo número de conexiones a procesar simultáneamente por el servidor web	5
Tamaño máximo de la cola de conexiones pendientes de servicio	4096
Máximo número de sesiones en el contenedor Web. (Valor por defecto -1, ilimitadas)	-1
Máximo número de conexiones en pools JDBC	32

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:

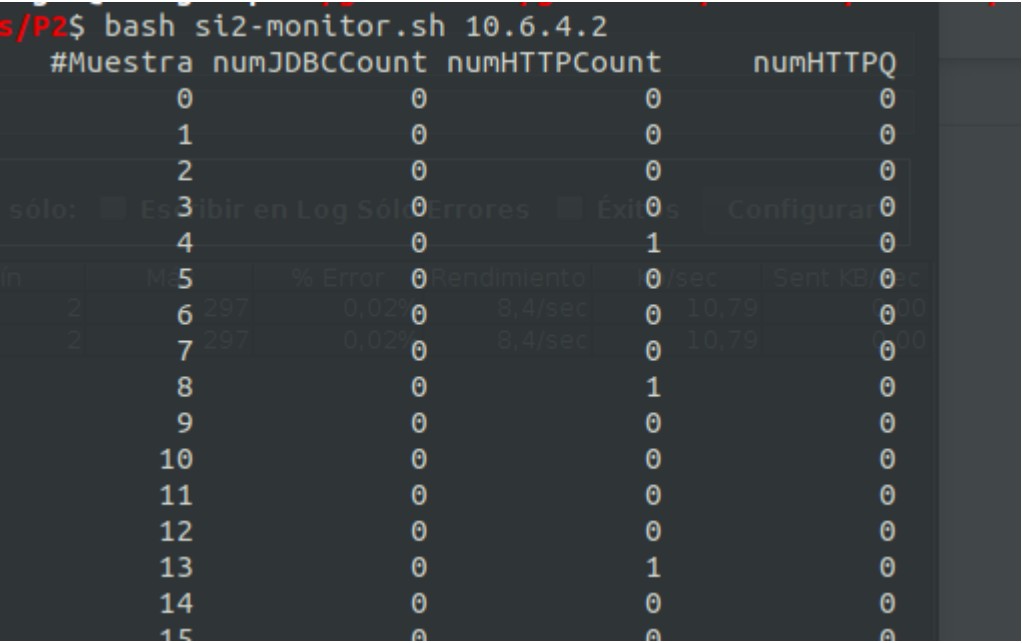
Utilizando el comando “nmon -f -s 0.1 -c 100” se generan 2 archivos (uno para el PC1, si2srv01 y otro para el 2 si2srv02) y se visualizan con nmon visualizer





Se ve un mayor uso de la CPU y de memoria (pues solo se disponen de 512M)

si2-monitor.sh:



- **A la vista de los resultados, ¿qué elemento de proceso le parece más costoso? ¿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 ...)**

Por los resultados arrojados por nmon podemos afirmar que el proceso con más costo es el de la memoria, pues cabe destacar que la máquina virtual solo tiene 512MB, aunque también hay un uso importante de la CPU en el PC2.

- **¿Le parece una situación realista la simulada en este ejercicio? ¿Por qué?**

No, el control que se tiene sobre aquello que está sucediendo y la duración y estrés al que se somete el sistema no es realista, además de que un servidor web suele tener muchas más prestaciones en la actualidad (y se ve sometido a mucho más estrés), que la simulación en la VM.

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

Redirigiendo el tráfico del servidor a otro, o creando una cola que tenga un tiempo de espera para llegar al servidor.

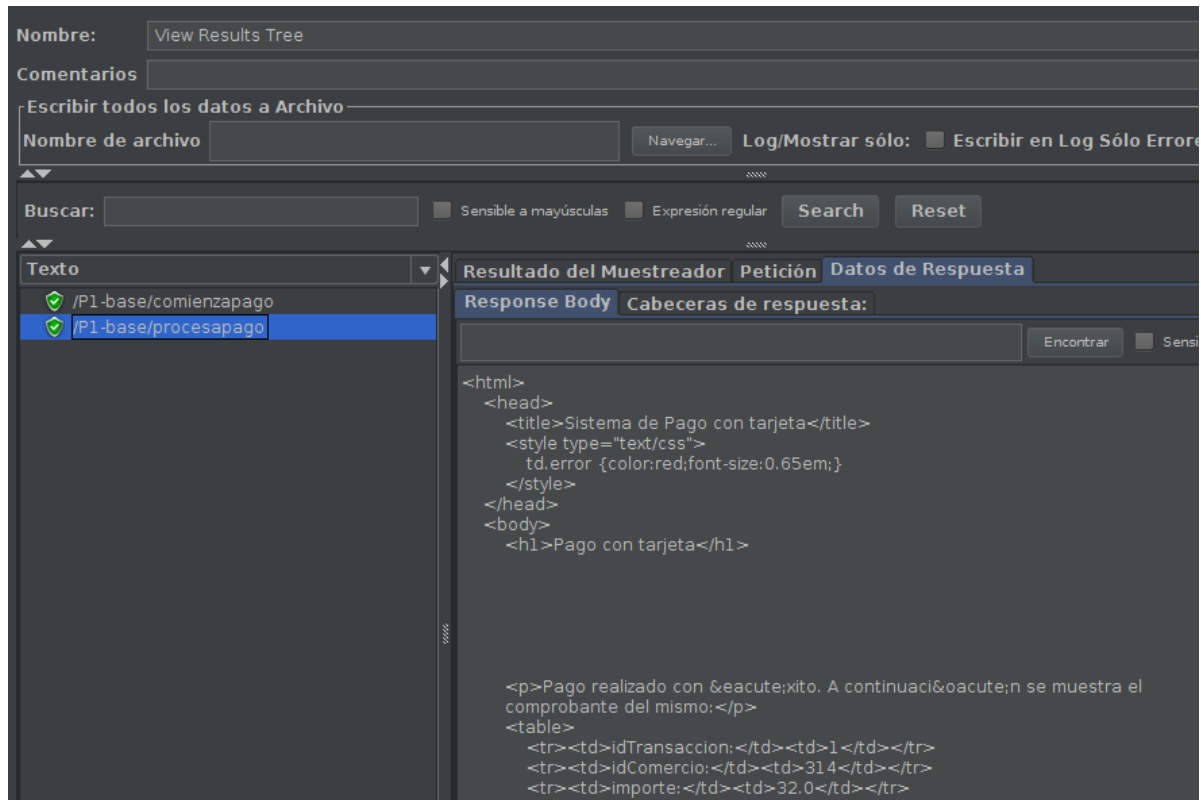
## ***Ejercicio 7:***

**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 los ejercicios 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.**

Funciona de manera correcta



## 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.
- Ejecutar la herramienta de monitorización nmon en ambas máquinas, preferiblemente en modo "Data-collect" (Ver 8.2.2).
- 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

o Arrancarlo cuando haya pasado el tiempo definido como rampa de subida de usuarios en JMeter (el tiempo de ejecución en JMeter se puede ver en la esquina superior derecha de la pantalla).

o Detenerlo cuando esté a punto de terminar la ejecución de la prueba. Este

momento se puede detectar observando cuando el número de hilos concurrentes en JMeter (visible en la esquina superior derecha) comienza a disminuir (su máximo valor es C).

o 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. En caso de usar nmon de forma interactiva, se deben tomar varios pantallazos del estado de la CPU durante la prueba, para volcar en la hoja de cálculo del dato de uso medio de la CPU (CPU average %). En caso de usar nmon en modo “Data-collect”, esta información se puede ver posteriormente en NMonVisualizer. Una tercera opción (recomendada) es ejecutar el comando vmstat en una terminal remota a la máquina si2srv02, para extraer directamente el valor de uso medio de su CPU 5 .

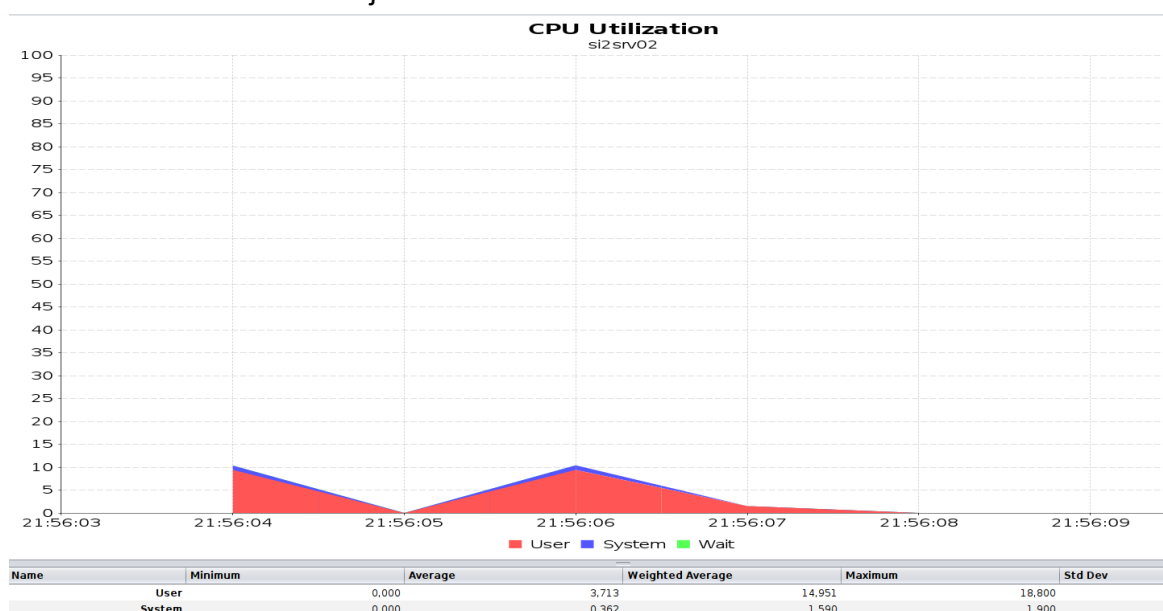
- Finalizada la prueba, salvar el resultado de la ejecución del Aggregate Report en un archivo, y registrar en la hoja de cálculo de resultados los valores Average, 90% line y Throughput para las siguientes peticiones:

o ProcesaPago.

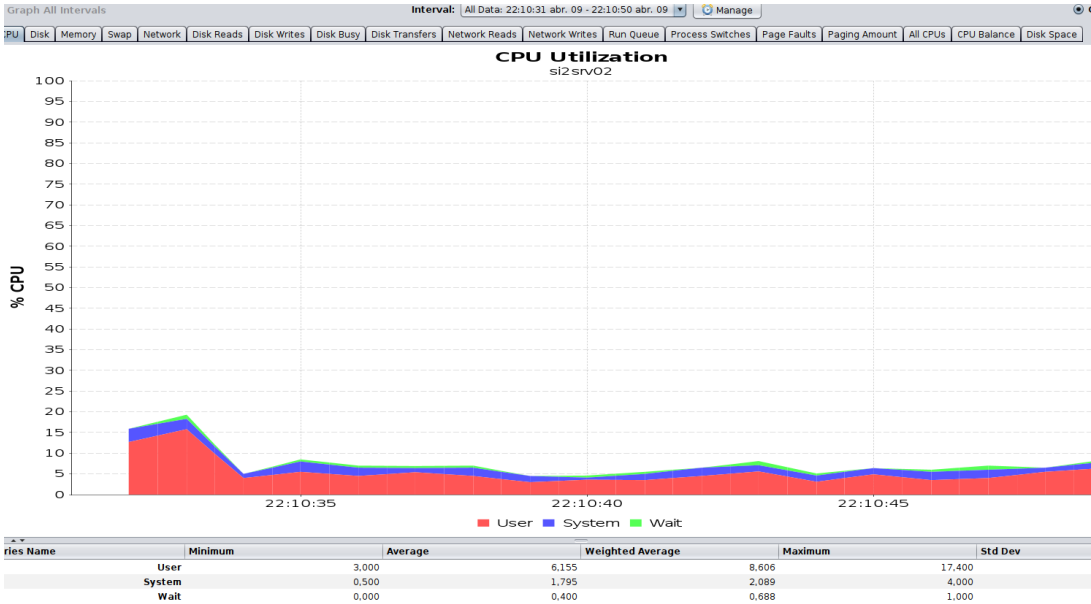
o Total.

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

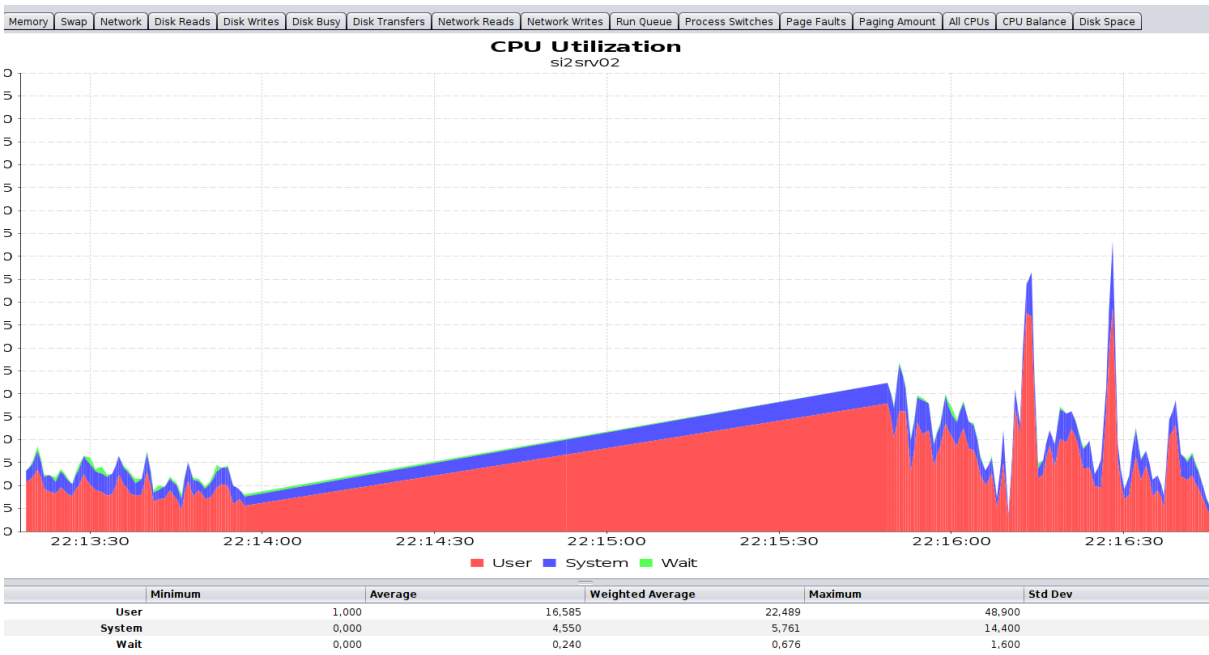
Con un usuario la CPU refleja este uso:



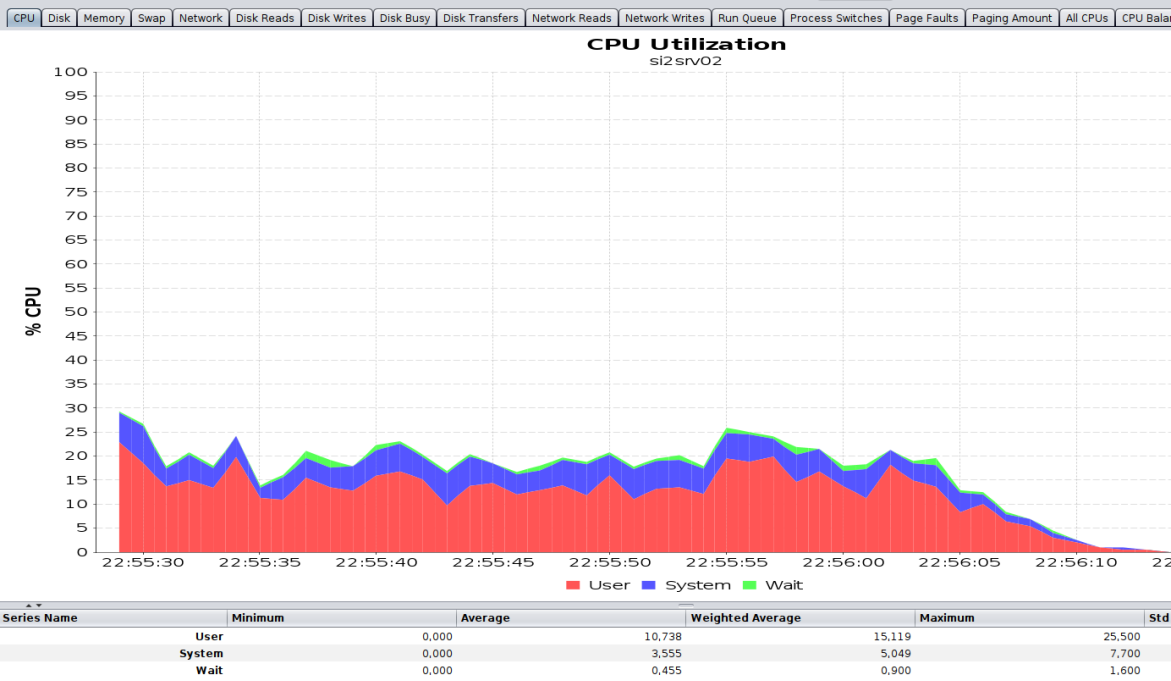
250:



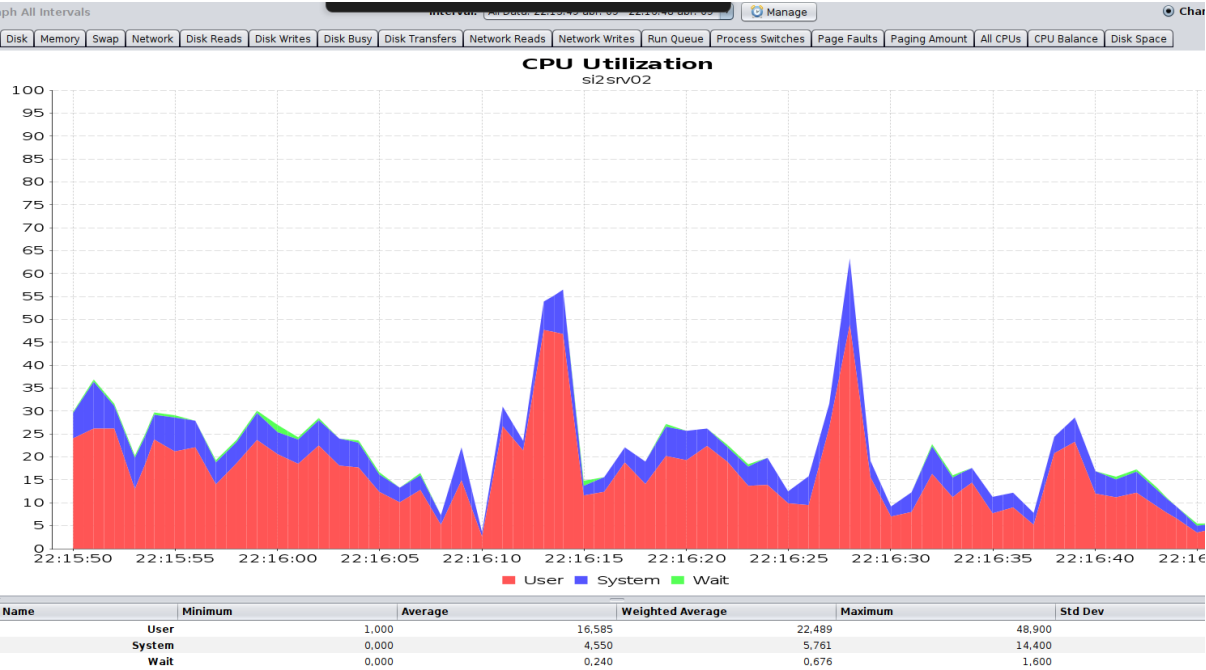
500:



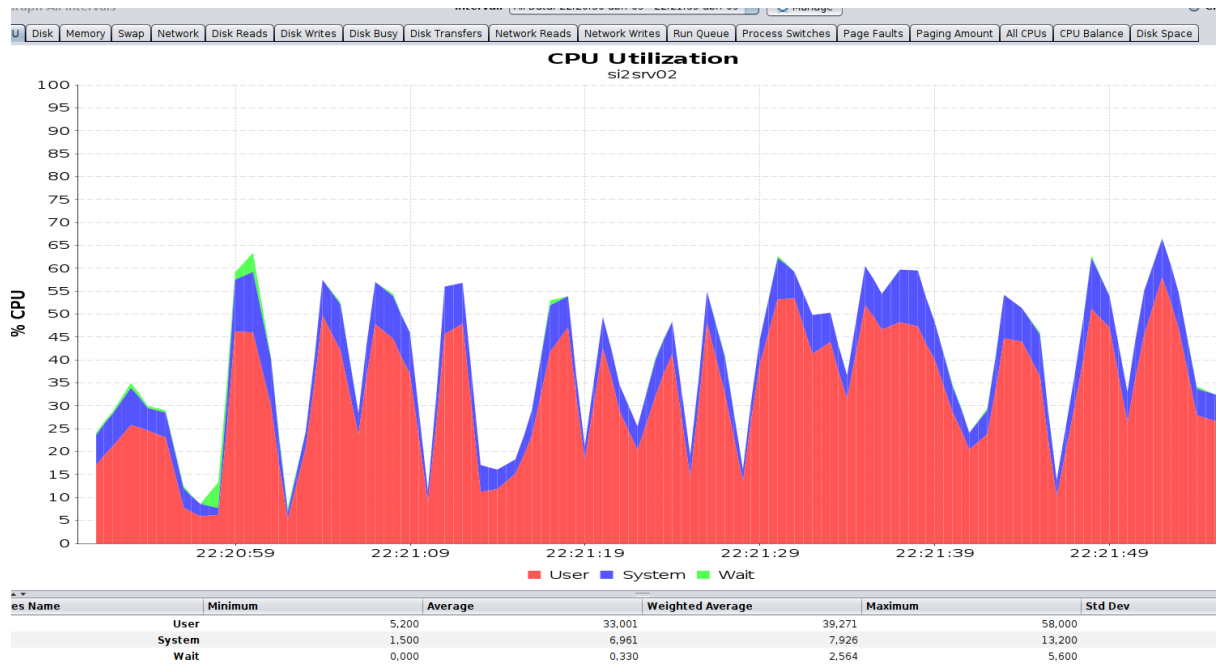
750:



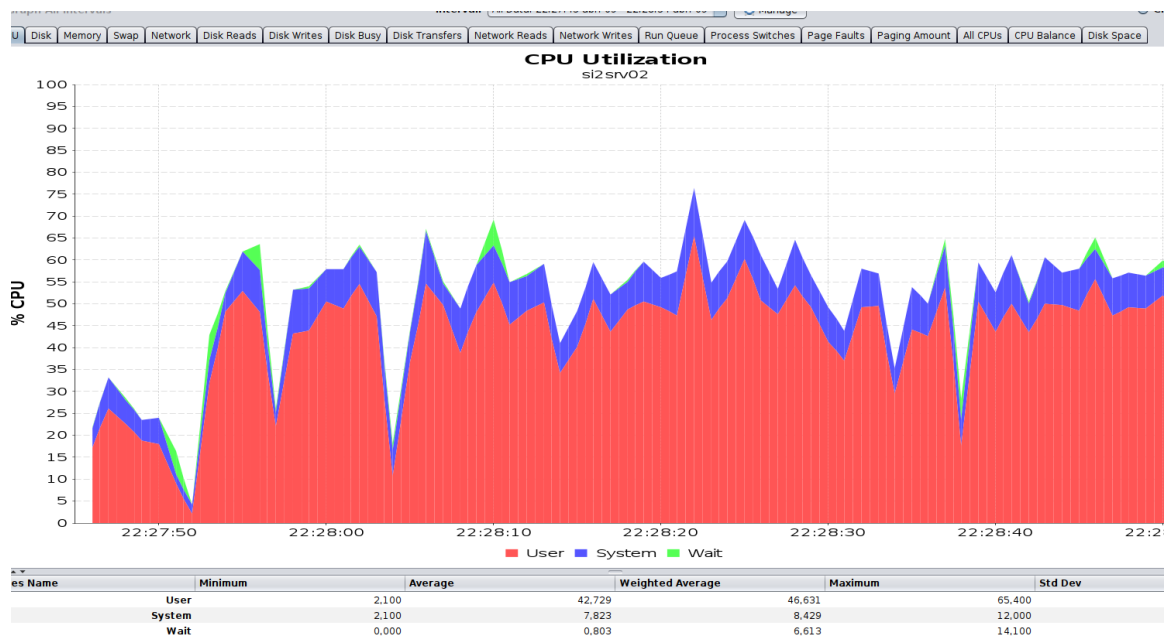
1000:



1250:



1250:



Todas las evidencias se encuentran en la carpeta de recursos en su correspondiente ejercicio.

El programa "si2-monitor.sh" no funcionaba aun siendo ejecutado con el passwordfile en la VM.



```
sergio@seregio-pc: ~/glassfish4/glassfish/domains/domain1/...
si2@si2srv02:~$ ls
passwordfile  si2-monitor.sh
si2@si2srv02:~$ bash si2-monitor.sh localhost
#Muestra numJDBCCount numHTTPCount numHTTPQ
0 0 0 0
1 0 0 0
2 0 1 0
3 0 0 0 00:00:39 0 0
4 0 0 0
5 0 0 0
6 0 1 0
7 0 0 0
8 0 0 0
9 0 1 0
10 0 0 0
11 ar test ahora 0 0
12 0 0 0
13 0 1 0
14 0 0 0
15 0 0 0
16 0 0 0
17 0 0 0
18 0 0 0
19 0 0 0
20 0 0 0
^C 21 0 0
22 0 0
^C
TOT.MUESTRAS MEDIA:
23 0 0.173913 0
```

## 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 throughput que se alcanza en ese punto, y cuál 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.