

---

## TEMA 2

---

# Monitorización

**PROBLEMA 2.1** En un sistema Linux se ha ejecutado la orden `uptime` tres veces en momentos diferentes. El resultado, de forma resumida, es el siguiente:

```
... load average:  6.85,  7.37,  7.83
... load average:  8.50, 10.93,  8.61
... load average: 37.34,  9.47,  3.30
```

Indique si la carga crece, decrece, se mantiene estacionaria o bien no puede decidir sobre ello.

**SOLUCIÓN:** No se puede decidir sobre la evolución porque no hay una tendencia clara en los valores de las medidas.

---

**PROBLEMA 2.2** En un sistema Linux se ha ejecutado la siguiente orden:

```
$ time quicksort
real 0m40.2s
user 0m17.1s
sys  0m3.2s
```

Indique si el sistema está soportando mucha o poca carga. Razone la respuesta.

**SOLUCIÓN:** Se trata de un sistema con una carga considerable, ya que hay una notable diferencia entre el tiempo de respuesta que experimenta el usuario (parámetro real) y el tiempo que realmente tarda en ejecutarse su programa (suma de los parámetros user y sys).

---

**PROBLEMA 2.3** Se sabe que la sobrecarga (*overhead*) de un monitor software sobre un computador es del 4 %. Si el monitor se activa cada 2 segundos, ¿cuánto tiempo tarda el monitor en ejecutarse por cada activación?

**SOLUCIÓN:** El monitor tarda en ejecutarse 80 milisegundos por activación.

---

**PROBLEMA 2.4** A continuación se muestra el resultado obtenido tras ejecutar la orden `top` en un sistema informático que emplea Linux como sistema operativo:

```
2:52pm up 17 days, 3:41, 1 user, load average: 0.15, 0.27, 0.32
54 processes: 51 sleeping, 3 running, 0 zombie, 0 stopped
CPU states: 23.8% user, 14.0% system, 17.0% nice, 45.2% idle
Mem: 257124K av, 253052K used, 4072K free, 8960K shrd, 182972K buff
Swap: 261496K av, 21396K used, 240100K free, 26344K cached
```

PID	USER	PRI	NI	SIZE	RSS	SHARE	STAT	LC	%CPU	%MEM	TIME	COMMAND
807	joan	18	2	5708	5708	532	R N	0	23.0	2.2	6:16	p_exec
809	joan	14	2	5708	5708	532	R N	0	14.0	2.2	3:42	p_exec
185	tomi	1	0	824	824	632	R	0	0.5	0.3	0:00	top
201	xp	1	0	1272	1208	644	S	0	0.1	0.4	5:49	xp_stat
1	root	0	0	60	56	36	S	0	0.0	0.0	0:03	init
2	root	0	0	0	0	0	SW	0	0.0	0.0	0:13	kflushd
7	root	0	0	0	0	0	SW	0	0.0	0.0	0:00	nfsiod
194	root	0	0	72	4	4	S	0	0.0	0.0	0:00	migetty
195	root	0	0	68	0	0	SW	0	0.0	0.0	0:00	migetty
179	root	0	0	532	312	236	S	0	0.0	0.1	0:00	sendmail

- ¿Cuánta memoria física tiene la máquina?
- ¿Qué porcentaje de la memoria física está marcada como usada según el monitor?
- ¿Cuál es la utilización media del procesador?
- ¿Cómo es la evolución de la carga media del sistema, ascendente o descendente?
- ¿Hay algún proceso ejecutándose en baja prioridad?
- ¿Cuánta memoria física ocupa el monitor?

SOLUCIÓN:

- a) 257124KiB. b) 98.4%. c) 54.8%. d) Descendente. e) Dos procesos. f) 824KiB.



**PROBLEMA 2.5** Considere las órdenes siguientes ejecutadas en un sistema Linux:

\$ time simulador_original	\$ time simulador_mejorado
real 0m24.2s	real 0m32.8s
user 0m15.1s	user 0m10.7s
sys 0m1.6s	sys 0m2.1s

- ¿Cuál es el tiempo de ejecución de ambos simuladores?
- Calcule, si es el caso, la mejora en el tiempo de ejecución del simulador mejorado respecto del original.

SOLUCIÓN: El simulador original se ejecuta en 16,7 segundos y el mejorado en 12,8 segundos (en realidad, los datos muestran que al sistema le cuesta más tiempo porque ha de atender otras tareas). El simulador mejorado es 1,3 veces más rápido que el original.



**PROBLEMA 2.6** El monitor *sar* (*system activity reporter*) de un computador se activa cada 15 minutos y tarda 750 ms en ejecutarse por cada activación. Se pide:

- Calcular la sobrecarga que genera este monitor sobre el sistema informático.
- Si la información generada en cada activación ocupa 8192 bytes, ¿cuántos ficheros históricos del tipo *saDD* se pueden almacenar en el directorio */var/log/sa* si se dispone únicamente de 200 MB de capacidad libre?

**SOLUCIÓN:** La sobrecarga es del 0,083 % y se pueden almacenar en total 266 ficheros históricos.



**PROBLEMA 2.7** El día 8 de octubre se ha ejecutado la siguiente orden en un sistema Linux:

```
% ls /var/log/sar
-rw-r--r--  1 root    root      3049952 Oct  6  23:50 sa06
-rw-r--r--  1 root    root      3049952 Oct  7  23:50 sa07
-rw-r--r--  1 root    root     2372184 Oct  8  18:40 sa08
```

¿Cada cuánto tiempo se activa el monitor *sar* instalado en el sistema? ¿Cuánto ocupa el registro de información almacenada cada vez que se activa el monitor?

**SOLUCIÓN:** El monitor se activa cada 10 minutos (la última activación del día se hace a las 23:50 horas). La información generada en cada activación ocupa aproximadamente 21 KB de capacidad.



**PROBLEMA 2.8** Indique el resultado que produce la ejecución de las siguientes órdenes sobre un sistema Linux con el monitor *sar* instalado:

1. *sar*
2. *sar -A*
3. *sar -u 1 30*
4. *sar -uB -f /var/log/sa/08*
5. *sar -d -s 12:30:00 -e 18:15:00 -f /var/log/sa/08*
6. *sadc*
7. *sadc 2 4*
8. *sadc 2 4 fichero*

**SOLUCIÓN:**

1. Utilización del procesador durante el día actual.
2. Toda la información recogida durante el día actual.
3. Utilización actual del procesador: 30 medidas tomadas con un período de un segundo.
4. Utilización del procesador y paginación de la memoria virtual durante el día 8 del mes.
5. Transferencias de disco desde las 12:30 hasta las 18:15 horas del día 8 del mes.
6. Información actual en formato binario recogida en una activación del monitor.
7. Información actual en formato binario recogida en dos activaciones separadas 4 segundos.
8. Información actual en formato binario recogida en dos activaciones separadas 4 segundos y depositadas en un fichero del disco de nombre *fichero*.



**PROBLEMA 2.9** Indíquese una orden (u órdenes) que se podría emplear para monitorizar los aspectos siguientes de la actividad en un computador que trabaja con el sistema operativo Linux:

1. Capacidad de memoria física ocupada por un proceso.
2. Número de cambios de contexto por segundo.
3. Carga media del sistema.
4. Número de interrupciones por segundo.
5. Capacidad libre de la unidad de disco magnético.
6. Usuarios conectados a la máquina.
7. Utilización del procesador en modo usuario.
8. Tiempo que lleva ejecutándose un proceso.
9. Tiempo que tarda un proceso en ejecutarse.

SOLUCIÓN:

1. Órdenes top, ps.
2. Órdenes vmstat, sar.
3. Órdenes uptime, sar.
4. Órdenes vmstat, sar.
5. Orden df.
6. Orden who.
7. Órdenes top, vmstat, sar.
8. Órdenes top, ps.
9. Orden time.



**PROBLEMA 2.10** Después de instrumentar un programa con la herramienta gprof el resultado obtenido ha sido el siguiente:

Flat profile:  
Each sample counts as 0.01 seconds.

% time	cumulative seconds	self seconds	calls	self s/call	total s/call	name
59.36	27.72	27.72	3	9.24	14.39	reduce
33.08	43.17	15.45	6	2.57	2.57	invierte
7.56	46.70	3.53	2	1.76	1.76	calcula

El grafo de dependencias muestra que `invierte()` es llamado desde el procedimiento `reduce()`.

1. ¿Cuánto tarda en ejecutarse el código propio del procedimiento `reduce()`?
2. ¿Cuál es el procedimiento más lento del programa? ¿Y el más rápido?
3. Si el código propio del procedimiento más lento de todos se sustituye por otro tres veces más rápido, ¿cuánto tiempo tardará en ejecutarse el programa?
4. Si el procedimiento `invierte()` se sustituye por una nueva versión cuatro veces más rápida, ¿qué mejora se obtendrá en el tiempo de ejecución?
5. Calcule cuál es la ganancia en velocidad máxima que se podría conseguir en el tiempo de ejecución mediante la optimización del código del procedimiento `invierte()`.

## SOLUCIÓN:

1. El código propio de reduce() tarda 9,24 s.
2. El procedimiento más lento es reduce() y el más rápido es calcula().
3. El programa se ejecutaría en 28,22 s.
4. El programa se ejecutaría 1,33 veces más rápidamente.
5. La máxima ganancia en velocidad que se podría conseguir es 1,49.

■

**PROBLEMA 2.11** Considere la siguiente secuencia de órdenes en un sistema informático donde está instalado el monitor sar:

```
% sadc resultado
% ls -l resultado
-rw-r--r-- 1 usuario grupo 712 2006-01-16 10:55 resultado
```

1. Indique qué contiene el fichero resultado y cómo se codifica esta información.
2. ¿Qué orden habría que emplear para visualizar en formato ASCII toda la información capturada por el monitor de actividad en la activación anterior?
3. Si el monitor sar está instalado en la máquina para ejecutarse cada tres minutos y disponemos de 50 MB de espacio en el disco duro para almacenar la información de actividad, ¿cuántos ficheros históricos diarios podremos mantener en esta instalación?

## SOLUCIÓN:

1. El fichero contiene toda la información, en formato binario, recogida sobre la actividad del sistema por la activación del monitor mediante la orden sadc.
2. La orden es sar -A -f resultado.
3. Se pueden mantener 153 ficheros.

■

**PROBLEMA 2.12** Un informático desea evaluar el rendimiento de un computador por medio del benchmark SPEC CPU2006. Una vez compilados todos los programas del paquete y lanzado su ejecución monitoriza el sistema con la orden vmstat 1 5. El resultado de las medidas de este monitor es el siguiente:

procs		memory-----				---swap--		-----io ---		---system		-- ----cpu----			
r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa
0	0	8	14916	92292	833828	0	0	0	3	0	7	3	1	96	0
1	0	8	14916	92292	833828	0	0	0	0	1022	40	100	0	0	0
3	0	8	14916	92292	833828	2	1	16	3	1016	34	99	1	0	0
1	0	8	14916	92292	833828	0	4	0	8	1035	36	98	2	0	0
2	0	8	14916	92292	833828	1	5	4	28	1035	36	99	1	0	0

Indique si, a la vista de los datos anteriores, los resultados obtenidos en la prueba evaluación serán correctos o no. Justifique la respuesta.

**SOLUCIÓN:** Los resultados serían incorrectos porque el sistema operativo presenta actividad de intercambio con el disco magnético (swapping).

■

**PROBLEMA 2.13** La monitorización de un programa de dibujo en tres dimensiones mediante la herramienta gprof ha proporcionado la siguiente información (por errores en la transmisión hay valores que no están disponibles):

Flat profile:

% time	cumulative seconds	self seconds	calls	self s/call	total s/call	name
xxxxx	xxxxx	15.47	3	5.16	5.16	colorea
xxxxx	xxxxx	1.89	5	0.38	0.38	interpola
xxxxx	xxxxx	1.76	1	1.76	3.65	traza
xxxxx	xxxxx	0.46				main

Call graph:

index	% time	self	children	called	name	
[1]	100.0	0.46	19.12		main	[1]
		15.47	0.00	3/3	colorea	[2]
		1.76	1.89	1/1	traza	[3]
-----						
		15.47	0.00	3/3	main	[1]
[2]	79.0	15.47	0.00	3	colorea	[2]
-----						
		1.76	1.89	1/1	main	[1]
[3]	18.6	1.76	1.89	1	traza	[3]
		1.89	0.00	5/5	interpola	[4]
-----						
		1.89	0.00	5/5	traza	[3]
[4]	9.7	1.89	0.00	5	interpola	[4]
-----						

1. ¿En cuánto tiempo se ejecuta el programa de dibujo?
2. Indique cuánto tiempo tarda en ejecutarse el código propio de main().
3. Establezca la relación de llamadas entre los procedimientos del programa así como el número de veces que se ejecuta cada uno de ellos.
4. Calcule el nuevo tiempo de ejecución del programa si se elimina el código propio de main() y se reduce a la mitad el tiempo de ejecución del código propio del procedimiento traza().
5. Proponga y justifique numéricamente una acción sobre el programa original que no afecte el procedimiento colorear() (ni su código ni el número de veces que es ejecutado) con el fin de conseguir que el programa se ejecute en 10 segundos.

SOLUCIÓN:

1. El programa se ejecuta en 19,58 s.
2. El código propio de main() se ejecuta en 0,46 s.
3. El procedimiento main() llama 3 veces a colorear() y una vez a traza(); a su vez, traza() llama 5 veces a interpola().
4. El tiempo de ejecución sería de 18,24 s.
5. El tiempo de ejecución no se puede reducir a 10 segundos sin afectar el procedimiento colorear() porque su contribución ya sobrepasa este valor.



**PROBLEMA 2.14** El resultado de la monitorización de una aplicación informática dedicada al análisis de modelos atmosféricos se muestra a continuación (nótese que hay información no disponible):

Flat profile:

% time	cumulative seconds	self seconds	calls	self s/call	total s/call	name
xxxxx	xxxxx	30.16	52	0.58	0.58	nimbo
xxxxx	xxxxx	5.13	2	2.56	2.56	borrasca
xxxxx	xxxxx	3.51	2	1.75	1.75	lluvia
xxxxx	xxxxx	1.76	1	1.76	34.17	nube

1. Indique cuánto tiempo tarda en ejecutarse el programa.
2. Determine el porcentaje del tiempo de ejecución que consume el procedimiento lluvia().
3. ¿Cuál es el procedimiento más lento de todo el programa (incluyendo código propio + código de las subrutinas a las que llama)?
4. ¿Cuánto tiempo tarda en ejecutarse el código propio de borrasca()?
5. Calcule el nuevo tiempo de ejecución del programa si el procedimiento nimbo() se rediseña y mejora 3 veces.
6. Proponga y justifique numéricamente alguna manera de reducir el tiempo de ejecución del programa original hasta los 20 segundos.

SOLUCIÓN:

1. El programa se ejecuta en 40,56 s.
2. El procedimiento lluvia() es responsable del 8,7 % del tiempo de ejecución.
3. El procedimiento más lento del programa es nube().
4. El código propio de borrasca() se ejecuta en 2,56 s.
5. El nuevo tiempo de ejecución sería de 20,45 s.
6. Solución libre. Por ejemplo, mejorando 4 veces el tiempo de ejecución del procedimiento nimbo() el tiempo total del programa se reduce hasta 17,94 s.

■

**PROBLEMA 2.15** Después de conectarse a un sistema informático, un usuario ejecuta las dos órdenes siguientes con el resultado que se muestra:

```
% uptime
9:50am up 173 days, 23:02, 1 user, load average: 0.00, 0.00, 0.00

% time simulador
real 8m0.70s
user 3m5.20s
sys 0m4.01s
```

1. ¿En qué condición de carga se encuentra el computador (baja, media o alta) en el momento de conexión del usuario?
2. ¿Cuál es el tiempo (en segundos) de ejecución del programa simulador?
3. ¿Encuentra alguna incoherencia en los resultados anteriores? Justifique la respuesta con argumentos sólidos.

## SOLUCIÓN:

1. El computador está en una situación de baja carga.
2. El computador ejecuta el simulador en 480,7 segundos, aunque el tiempo de ejecución efectivo del mismo es de 189,21 segundos.
3. En efecto, el hecho de que la carga sea baja (parámetros de load average a cero) está en contradicción con la espera de 291,49 segundos que experimenta la ejecución del simulador.

■

**PROBLEMA 2.16** En un servidor con S.O. Linux se tiene instalado un monitor de actividad sar (system activity reporter). Se sabe que cada activación del monitor implica la ejecución de un total de 450 instrucciones máquina y almacena un total de 1024 bytes de información en el fichero /var/log/sa/saDD del día DD correspondiente. Si el procesador del equipo tiene una velocidad de ejecución de 75 MIPS (millions of instructions per second):

- a) ¿Qué valor debe tener el periodo de muestreo (en milisegundos) si se quiere una sobrecarga (overhead) del 5%?
- b) Suponiendo ahora que el monitor se activa una vez cada 10 minutos, ¿cuál será el tamaño máximo de cada fichero del directorio /var/log/sa?

## SOLUCIÓN:

- a) 0.12 ms. b) 147456 bytes.

■

**PROBLEMA 2.17** Después de instrumentar un programa con la herramienta gprof el resultado obtenido ha sido el siguiente:

Flat profile:  
Each sample counts as 0.01 seconds.

% time	cumulative seconds	self seconds	calls	self ms/call	total ms/call	name
60.28	70.32	70.32	10	7032.00	7032.00	ordena
28.96	104.10	33.78	8	4222.50	4222.50	inicio
10.76	116.66	12.56	34	369.41	369.41	escala

- a) ¿Cuántas medidas se han tomado para obtener estos resultados?
- b) Si el procedimiento más rápido de los tres se sustituye por otro tres veces más rápido, ¿cuánto tiempo tardará en ejecutarse el programa?
- c) Si los tres procedimientos `ordena`, `inicio` y `escala` se sustituyen al mismo tiempo por nuevas versiones 2, 4 y 5 veces más rápidas, respectivamente, ¿cuál sería la ganancia en velocidad conseguida con respecto al programa original?

## SOLUCIÓN:

- a) 11666 muestras. b) 108.29 segundos. c) 2.53.

■