



UNIVERSIDAD DE LA INTEGRACIÓN DE LAS AMÉRICAS

Facultad de Ingeniería
Carreras de Ingeniería en Informática e
Ingeniería en Sistemas

SISTEMAS OPERATIVOS

Informe de Laboratorio 1 Gestión de Procesos

Nombre: Gonzalo Aquino Alvarenga

Materia: Sistemas Operativos

Fecha: 21/06/2025

Introducción

El presente informe detalla las actividades realizadas en el primer laboratorio de análisis de sistemas operativos, centrado en la gestión de procesos. Se realizaron pruebas prácticas sobre el comportamiento de los procesos en Linux, simulando sus estados, observando la planificación del CPU y reproduciendo una situación de bloqueo mutuo (deadlock), todo dentro de un entorno controlado.

Simulación de Estados de un Proceso

Para esta parte del laboratorio se creó un script en Python que simula el paso de un proceso por cinco estados típicos: Nuevo, Listo, Ejecutando, Bloqueado y Terminado. Se estableció un retraso de 2 segundos entre cada estado para visualizar claramente la transición.

```
GNU nano 7.2                                proceso_estados.py *
```

```
import time

estados = ["Nuevo", "Listo", "Ejecutando", "Bloqueado", "Terminado"]
tiempos = []

print("== Simulación de estados de proceso ==\n")

for estado in estados:
    inicio = time.time()
    print(f"Estado: {estado}")
    time.sleep(2)
    fin = time.time()
    duracion = round(fin - inicio, 2)
    tiempos.append(duracion)

print("\n== Tiempos de transición por estado ==")
for estado, tiempo in zip(estados, tiempos):
    print(f"{estado}: {tiempo} segundos")
```

GNU nano 7.2 interface with keyboard shortcuts at the bottom:

Ctrl+O	Ayuda	Ctrl+S	Guardar	Ctrl+F	Buscar	Ctrl+X	Cortar	Ctrl+E	Ejecutar	Ctrl+U	Ubicación	Ctrl+Z	Deshacer	Ctrl+M	Poner marca	Ctrl+J	A llave	Ctrl+Q	Anterior	Ctrl+B	Atrás
Ctrl+W	Salir	Ctrl+G	Leer fich.	Ctrl+R	Reemplazar	Ctrl+V	Pegar	Ctrl+I	Justificar	Ctrl+N	Ir a línea	Ctrl+L	Rehacer	Ctrl+C	Copiar	Ctrl+K	Buscar atrás	Ctrl+M	Siguiente	Ctrl+F	Adelante

```

sole-cardozo@sole-cardozo-VirtualBox:~$ python3 proceso_estados.py
=== Simulación de estados de proceso ===

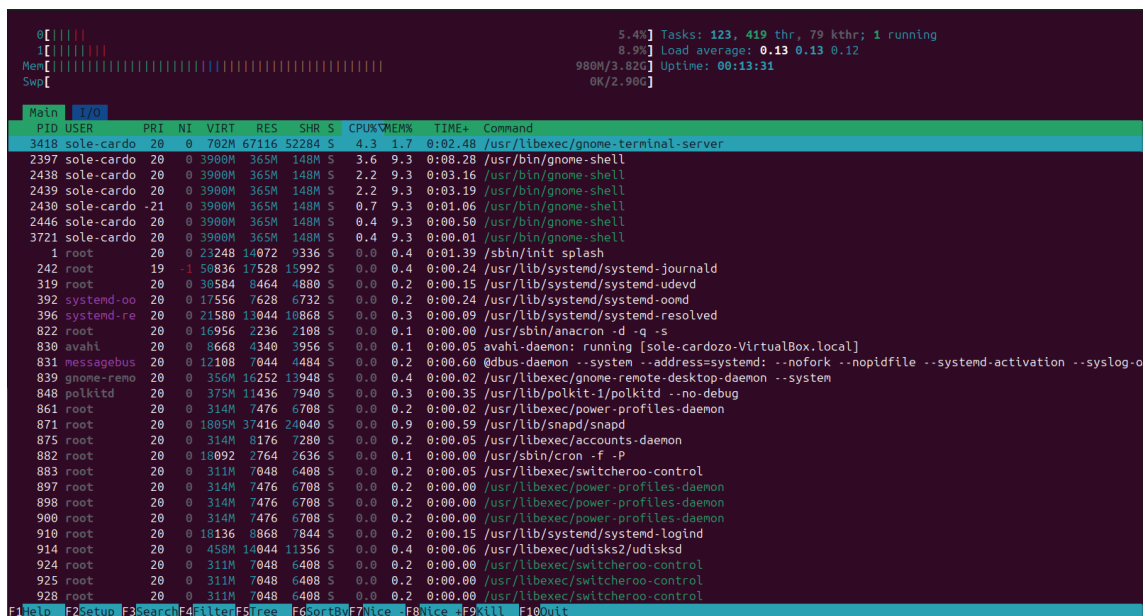
Estado: Nuevo
Estado: Listo
Estado: Ejecutando
Estado: Bloqueado
Estado: Terminado

=== Tiempos de transición por estado ===
Nuevo: 2.0 segundos
Listo: 2.0 segundos
Ejecutando: 2.0 segundos
Bloqueado: 2.0 segundos
Terminado: 2.0 segundos

```

Paralelamente, se monitoreó el proceso en ejecución utilizando la herramienta htop. Esto permitió ver cómo el sistema reconoce y maneja dicho proceso en tiempo real.

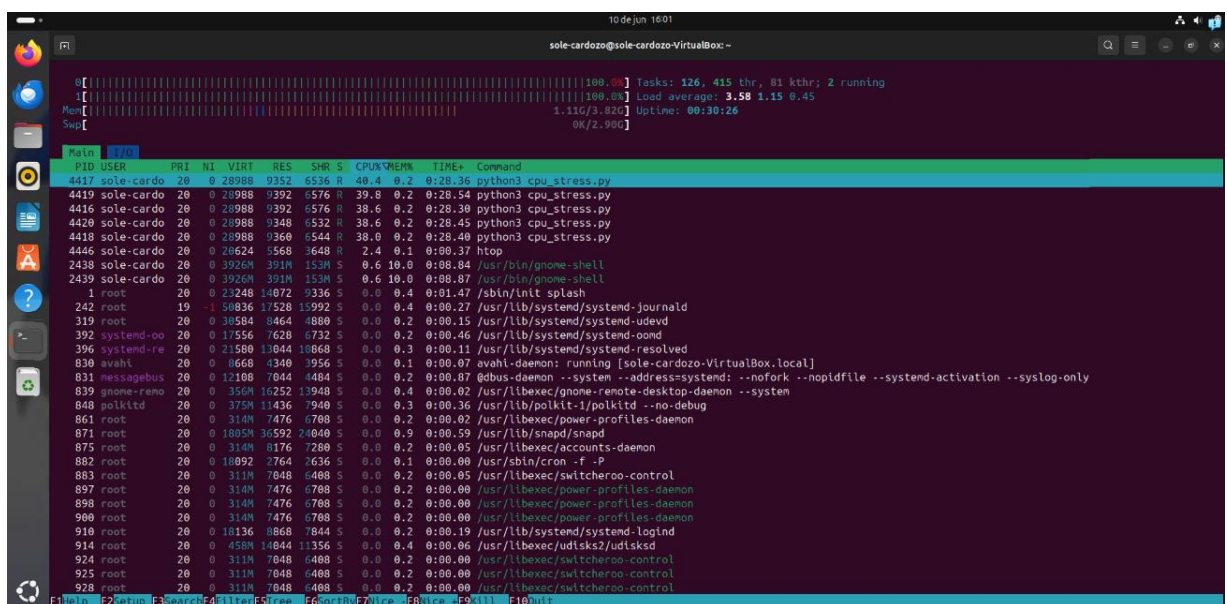
```
~$ htop
```



Además, se midieron los tiempos exactos entre los estados y se registraron en una hoja de cálculo (datos_mediciones.xlsx) para análisis posterior.

Observación de la Planificación del CPU

La siguiente etapa consistió en ejecutar cinco procesos al mismo tiempo, cada uno corriendo un script que mantiene al procesador ocupado constantemente. El objetivo era observar cómo el sistema operativo distribuye el uso del CPU entre ellos.

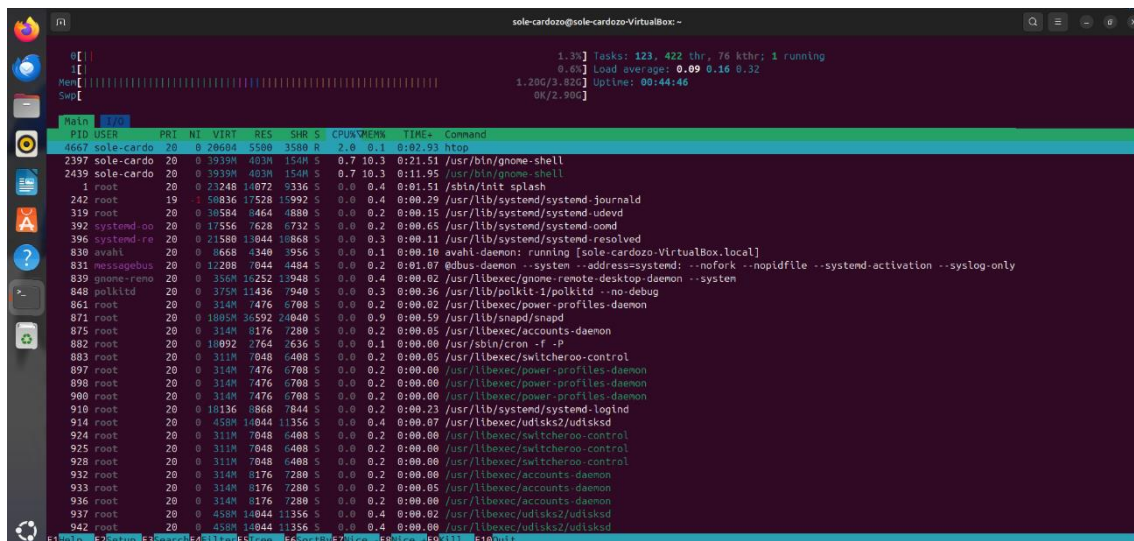


Se comprobó que Linux reparte la carga de manera equitativa, alternando el uso del procesador entre todos los procesos. Este comportamiento se asemeja al algoritmo de planificación Round Robin, en el que cada proceso recibe un turno. A diferencia del algoritmo FIFO, que atiende por orden de llegada sin alternancia, Round Robin permite mayor fluidez y evita bloqueos prolongados.

Simulación de Deadlock

Para finalizar el laboratorio, se simuló un bloqueo mutuo entre dos hilos. Cada hilo intentaba acceder a dos recursos compartidos en orden contrario, generando un escenario donde ambos quedan esperando al otro indefinidamente.

```
sole-cardozo@sole-cardozo-VirtualBox:~$ python3 deadlock_simulado.py
Hilo 1: esperando recurso A
Hilo 1: obtuvo recurso A
Hilo 2: esperando recurso B
Hilo 2: obtuvo recurso B
Hilo 1: esperando recurso B
Hilo 2: esperando recurso A
```



The screenshot shows a terminal window with a dark background. At the top, it displays system statistics: 1.3% tasks, 123 tasks, 422 threads, 76 kbytes, 1 running process, 0.6% load average, 0.09 0.16 0.32, 1.20G/3.02G uptime, and 00:44:46. Below this is a table of processes with columns: Main, PID, USER, PRI, NI, VIRT, RES, SHR, S, CPU%, MEM%, TIME+, and Command. The table lists various system processes and user processes, including the user's shell and several system daemons.

Main	PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
	1667	sole-cardo	20	0	20604	5580	3580	P	2.0	0.1	0:00.53	http
	2397	sole-cardo	20	0	3939M	403M	154M	S	0.7	10.3	0:21.51	/usr/bin/gnome-shell
	2439	sole-cardo	20	0	3939M	403M	154M	S	0.7	10.3	0:11.95	/usr/bin/gnome-shell
	1	root	20	0	2324B	14072	9336	S	0.0	0.4	0:01.51	/sbin/init splash
	242	root	19	1	50836	17528	15992	S	0.0	0.4	0:00.29	/usr/lib/systemd/systemd-journald
	319	root	20	0	30584	8464	4880	S	0.0	0.2	0:00.15	/usr/lib/systemd/systemd-udev
	392	systemd-oo	20	0	17556	7628	6732	S	0.0	0.2	0:00.65	/usr/lib/systemd/systemd-oomd
	396	systemd-re	20	0	21580	13044	18868	S	0.0	0.3	0:00.11	/usr/lib/systemd/systemd-resolved
	830	avahi	20	0	8668	4340	3956	S	0.0	0.1	0:00.10	avahi-daemon: running [sole-cardozo-VirtualBox.local]
	831	messagebus	20	0	12208	7044	4484	S	0.0	0.2	0:01.07	dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-activation --syslog-only
	839	gnome-remo	20	0	356M	16252	13948	S	0.0	0.4	0:00.02	/usr/libexec/gnome-remote-desktop-daemon --system
	848	polkitd	20	0	375M	11436	7940	S	0.0	0.3	0:00.36	/usr/lib/polkit-1/polkitd --no-debug
	861	root	20	0	314M	7476	6708	S	0.0	0.2	0:00.02	/usr/libexec/power-profiles-daemon
	871	root	20	0	1805M	36592	24040	S	0.0	0.9	0:00.59	/usr/lib/snapd/snapd
	875	root	20	0	314M	8176	7280	S	0.0	0.2	0:00.05	/usr/libexec/accounts-daemon
	882	root	20	0	18092	2764	2636	S	0.0	0.1	0:00.00	/usr/sbin/cron -f -P
	883	root	20	0	311M	7048	6408	S	0.0	0.2	0:00.05	/usr/libexec/switcheroo-control
	897	root	20	0	314M	7476	6708	S	0.0	0.2	0:00.00	/usr/libexec/power-profiles-daemon
	898	root	20	0	314M	7476	6708	S	0.0	0.2	0:00.00	/usr/libexec/power-profiles-daemon
	900	root	20	0	314M	7476	6708	S	0.0	0.2	0:00.00	/usr/libexec/power-profiles-daemon
	910	root	20	0	18136	8868	7844	S	0.0	0.2	0:00.23	/usr/lib/systemd/systemd-logind
	914	root	20	0	458M	14044	11356	S	0.0	0.4	0:00.07	/usr/libexec/udisks2/udisksd
	924	root	20	0	311M	7048	6408	S	0.0	0.2	0:00.00	/usr/libexec/switcheroo-control
	925	root	20	0	311M	7048	6408	S	0.0	0.2	0:00.00	/usr/libexec/switcheroo-control
	928	root	20	0	311M	7048	6408	S	0.0	0.2	0:00.00	/usr/libexec/switcheroo-control
	932	root	20	0	314M	8176	7280	S	0.0	0.2	0:00.00	/usr/libexec/accounts-daemon
	933	root	20	0	314M	8176	7280	S	0.0	0.2	0:00.05	/usr/libexec/accounts-daemon
	936	root	20	0	314M	8176	7280	S	0.0	0.2	0:00.00	/usr/libexec/accounts-daemon
	937	root	20	0	458M	14044	11356	S	0.0	0.4	0:00.02	/usr/libexec/udisks2/udisksd
	942	root	20	0	458M	14044	11356	S	0.0	0.4	0:00.00	/usr/libexec/udisks2/udisksd

Conclusión

El laboratorio permitió comprender de manera práctica cómo un sistema operativo gestiona procesos, reparte el uso del CPU y responde ante errores de sincronización como los interbloqueos. La experiencia también sirvió para familiarizarse con herramientas de monitoreo como htop y con la creación de scripts simples que simulan comportamientos reales del sistema.