


# Tema 16

## Sistemas Operativos: Gestión de Procesos

 Esquema general de conceptos clave

# 1. Introducción

-  Un **proceso** es un programa en ejecución
- Utiliza recursos: **CPU, memoria, E/S**
- Buena gestión permite:
  - Multitarea eficiente
  - Aislamiento seguro
  - Aprovechamiento óptimo del hardware
- Fundamental en entornos: **cloud, contenedores, IA**

## 2. Ciclo de vida del proceso (5 estados)

Modelo clásico:

Nuevo → Listo → Ejecutando → Bloqueado → Terminado


 Transiciones:

- Dispatch → CPU
- I/O → Bloqueado
- Fin I/O → Listo
- Interrupciones → Listo

### 3. Modelo extendido: 7 estados

 Añade:

- Listo suspendido
- Bloqueado suspendido

 Permite usar disco para procesos suspendidos

✓ Mejora rendimiento en sistemas con alta carga

 Usado en SO con paginación bajo demanda

## 4. PCB – Process Control Block

 Estructura que representa cada proceso

Contiene:

- PID, estado, prioridad
- Registros de CPU
- Memoria, archivos abiertos
- UID/GID, estadísticas

 Esencial para cambios de contexto

## 5. Hilos (threads)

🧩 Subprocesos dentro de un mismo proceso

📦 Comparten espacio de direcciones

### Modelos:

- 1:1 (Linux): buen rendimiento, más coste
- N:1: solo en espacio usuario, limitado
- M:N: híbrido (Go, Erlang)

⚠️ Riesgos:

- Carreras, interbloqueos
  - ✓ Soluciones: mutex, semáforos, monitores

## 6. Planificación de CPU

 Criterios:

- Tiempo de retorno, espera, respuesta
- Rendimiento global (throughput)

### Algoritmos:

- FCFS (First Come, First Served)
- SJF (Shortest Job First)
- RR (Round Robin)
- Prioridades
- Multicolos con realimentación




 Linux usa: **CFS** (Completely Fair Scheduler)

 Árbol rojo-negro → eficiencia  $O(\log n)$

## 7. Concurrency and synchronization

 Procesos concurrentes requieren control:

### Mechanisms:





-  Semáforos (wait/signal)
-  Monitores (síncronos, OO)
-  Spinlocks (bloqueo activo)

 Ejemplo clásico: **Filósofos comensales**



## 8. Comunicación entre procesos (IPC)

 Métodos comunes:

-  Memoria compartida
-  Pipes (anónimos / con nombre)
-  Colas de mensajes
-  Sockets (locales o red)

 Usos: microservicios, pipelines, servicios REST

## 9. Interbloqueos (Deadlocks)

### Condiciones de Coffman:

1. Exclusión mutua
2. Retención y espera
3. No expropiación
4. Espera circular

### Estrategias:

- Prevención: evitar condiciones
- Evitación: algoritmo del **banquero**
- Detección y recuperación

## 10. Casuística por sistema operativo

Sistema	Gestión de procesos	Herramientas
Linux	<code>fork()</code> , <code>exec()</code> , <code>/proc</code>	<code>ps</code> , <code>htop</code> , <code>strace</code>
Windows	<code>CreateProcess()</code>	Task Manager, PowerShell
Contenedores	<code>namespaces</code> , <code>cgroups</code>	Docker, Podman

# 11. Seguridad en la gestión de procesos

## Riesgos comunes:

- Procesos zombie
- Condiciones de carrera
- IPC inseguro

## Mitigaciones:

- ASLR (direcciones aleatorias)
- UID/GID + sandboxing
- Control de acceso + auditoría

## Conclusión

- ✓ Un proceso es mucho más que un programa:
  - Tiene estado, recursos y control
- ✓ La buena gestión de procesos permite:
  - Seguridad, rendimiento, concurrencia
- ✓ Dominio esencial para administrar sistemas modernos