# Gestió de Processos

### Unitat 2.0 - Nel Banqué Torné

---

## 1. Processos en Unix/Linux

En els sistemes operatius **Unix/Linux**, la gestió de processos és essencial per controlar l'execució dels programes i assignar els recursos del sistema de manera eficient. A continuació, es presenta una introducció als conceptes bàsics de processos, la seva comunicació, sincronització i monitorització.

---

### 1.1 Creació de processos

Per començar a gestionar processos, podem crear tres processos en una terminal executant diverses vegades la mateixa ordre:

ps -e

### Comentaris sobre la comanda: ps -e

L'opció **-e** a l'ordre ps mostra tots els processos en execució en el sistema, no només els de l'usuari de la sessió actual. Aquestes són algunes observacions sobre la sortida de la comanda:

- Els processos poden tenir diferents **PID** (identificadors de procés), per exemple: **1053, 1054, 1055**.
- Fins i tot l'ordre ps en si mateixa és un procés, per tant, apareixerà a la llista.
- La tercera columna de la sortida indica el **temps de CPU** que ha utilitzat cada procés.

---

## 2. Monitorització de processos

### 2.1 Comanda top

Una altra ordre útil per supervisar els processos en temps real és top. Aquesta ordre proporciona una visió contínua de l'activitat del processador i altres recursos del sistema.

- La primera columna de la sortida identifica els **indicadors de procés**, com ara l'estat del procés (actiu, inactiu, etc.).

### Observacions

La majoria dels processos del sistema estan **inactius**, a l'espera d'algun esdeveniment, com ara un clic del ratolí o la premsa d'una tecla. A la sortida també s'indica l'**usuari propietari del procés**, el **PID** (identificador de procés) i el **PPID** (identificador del procés pare), que mostra d'on es va originar cada procés.

---

# 3. Estats dels processos

## 3.1 Estats comuns dels processos

- **Dormit o bloquejat en memòria principal**: El procés està esperant que es produeixi un determinat esdeveniment (com la finalització d'una operació d'E/S).
- **Zombi**: Aquest és l'estat final d'un procés després d'executar la crida a sistema exit. Un procés arriba a aquest estat després de completar la seva execució, però encara està registrat fins que el procés pare l'elimina definitivament.

## 3.2 Transicions d'estat

Un procés pot estar en diversos estats durant la seva execució, com ara:

- **Execució en mode usuari**: El procés està executant codi d'usuari.
- **Execució en mode nucli**: El procés ha realitzat una crida a sistema i està executant codi del sistema operatiu.
- **Intercanvi a memòria secundària**: Si el sistema no té prou memòria, el planificador pot moure processos a memòria secundària (swap) fins que es puguin tornar a executar.

## Exemple de transició d'estat, comanda fork:

Quan un procés invoca una crida a sistema com fork que, crea processos a partir d'un procés inicial (pare i fill), pot canviar entre diversos estats:

1. **Mode usuari** → El procés comença l'execució en mode usuari.
2. **Mode nucli** → Passa a mode nucli quan invoca una crida a sistema.
3. **Bloqueig o espera** → El procés pot quedar bloquejat esperant que es completi una operació.
4. **Zombi** → Quan finalitza, entra en estat zombi fins que el procés pare l'allibera.

---

# 4. Eina per espiar processos: strace

Una manera útil per analitzar el comportament d'un procés és utilitzar l'ordre **strace**, que permet monitoritzar les crides a sistema fetes per un procés.

### Com espiar un procés

En una terminal separada, es pot executar la comanda següent per espiar un procés amb el PID corresponent:

strace -f -p {pid}

- Si el procés està en una crida a sistema com read, això indica que el procés està esperant entrada de dades.

---

# 5. El PCB (Process Control Block)

El **PCB** (Bloc de Control de Procés) és una estructura de dades que utilitza el sistema operatiu per controlar i supervisar un procés. Cada procés en execució té el seu PCB corresponent.

### Components del PCB

- **Punters**: Apunten a altres estructures necessàries per a l'execució del procés.
- **Estat del procés**: El seu estat actual (execució, bloquejat, zombi, etc.).
- **Identificadors**: Com el PID i PPID.
- **Taula de fitxers oberts**: Els fitxers que el procés té oberts.
- **Recursos assignats**: Recursos com la memòria i el temps de CPU.
- **Context dels registres de CPU**: Els valors dels registres de la CPU quan el procés es va interrompre.
- **Informació sobre la memòria**: Regions de memòria utilitzades pel procés.
- **Informació sobre la planificació**: Prioritat i temps d'execució del procés.
- **ptrace**: Un camp especial per monitoritzar i depurar el procés.

### 5.1 Taula PCB

El **kernel** manté una estructura de taula coneguda com la **Taula PCB** per gestionar eficientment els processos. Aquesta taula permet accedir de manera ràpida a les dades necessàries per a la planificació i gestió dels processos.

# 6. Conclusió

La gestió de processos en **Unix/Linux** és crucial per a la utilització eficient dels recursos del sistema i per assegurar una correcta assignació de temps de CPU i memòria. Comprendre com funcionen els estats dels processos, les crides a sistema i com el sistema operatiu gestiona els PCB permet una gestió adequada dels processos i la detecció de possibles problemes o ineficiències en l'execució del sistema.