Estructura dels Sistemes Operatius

Unitat 1 - Nel Banqué Torné

---

1. Crides a sistema

Les crides a sistema són funcions proporcionades pel sistema operatiu per gestionar de manera eficient els recursos de l'ordinador. Permeten que els programes d'usuari puguin interactuar amb el maquinari de manera segura i controlada.

Funcions més comunes:

- Crear, obrir, tancar i eliminar fitxers.
- Crear i gestionar nous processos.
- Gestionar la xarxa.

Nota important sobre accés al maquinari:

Els programes d'usuari no tenen accés directe a la memòria ni als altres recursos del maquinari. Tot l'accés ha de passar pel sistema operatiu, que s'encarrega de gestionar de manera segura i controlada aquests recursos.

---

2. Accés al Kernel

El **Kernel** és la part central del sistema operatiu que gestiona directament el maquinari. Els programes d'usuari no poden accedir directament al Kernel, sinó que han d'utilitzar mecanismes especials anomenats **TRAPS** per sol·licitar serveis del Kernel.

Pasos d'un programa C per accedir al Kernel:

1. El programa col·loca els valors necessaris (file descriptor, buffer, nombre de bytes) als registres de la CPU.
2. La llibreria de sistema col·loca el número de syscall en el registre corresponent (per exemple, RAX en sistemes Linux).
3. S'executa la syscall, que indica al Kernel que s'ha sol·licitat un servei.
4. El Kernel pot cridar el planificador de tasques per gestionar l'execució de processos.
5. Si es tracta d'una operació d'E/S (com llegir d'un dispositiu), el Kernel llegeix les dades i les col·loca en el buffer d'usuari.
6. El resultat de la syscall és retornat al programa C.
7. El programa continua la seva execució utilitzant les dades que ha rebut.

---

3. Què és una interrupció?

Les **interrupcions** són esdeveniments, ja sigui de maquinari o de programari, que ocorren de manera asíncrona respecte al procés que s'està executant. Quan es produeix una interrupció, la CPU atura el que està fent per atendre la interrupció.

Gestió d'una interrupció:

1. El sistema operatiu guarda l'estat actual del procés (el seu **context**) per poder reprendre'l més tard.
2. Canvia el mode d'execució de **mode usuari** a **mode kernel**.
3. Determina la causa de la interrupció.
4. Executa la funció apropiada per atendre la interrupció.
5. Finalment, torna a **mode usuari** per continuar amb el procés interromput.

Tractament prioritari: Les interrupcions tenen una prioritat superior als processos normals, de manera que han de ser tractades immediatament.

---

4. Excepcions

Les **excepcions** són errors o esdeveniments inesperats que ocorren durant l'execució d'un programa. A diferència de les interrupcions, que són esperades, les excepcions normalment sorgeixen de problemes durant l'execució d'instruccions.

Tipus d'excepcions:

- **Errors detectats abans de l'execució**: Són errors que el sistema detecta abans d'executar una instrucció específica. En alguns casos, el sistema pot corregir-los.
- **Traps**: Aquests són tipus especials d'excepcions on el sistema operatiu pot corregir l'error o enviar un senyal al procés perquè prengui accions.

---

5. Algunes Estructures dels Sistemes Operatius

Els sistemes operatius tenen diverses estructures segons com organitzen els serveis d'usuari i kernel. Aquests són unes d'elles.

5.1 Estructura Monolítica

En una **estructura monolítica**, tant els serveis d'usuari com els serveis del Kernel s'implementen en un sol espai d'adreces.

Pros:

- Les crides a sistema són més ràpides, ja que tot està en el mateix espai de memòria.
- Els controladors de dispositius es carreguen directament al nucli, proporcionant un accés eficient al maquinari.

Cons:

- El codi és difícil d'entendre, modificar i mantenir, ja que tot està interconnectat.
- Qualsevol error en una part del nucli pot afectar tot el sistema.

5.2 Estructura per capes

En una **estructura per capes**, el sistema operatiu es divideix en capes, on cada capa només interactua amb les capes adjacents.

Pros:

- És més fàcil de dissenyar i mantenir, ja que les capes tenen una independència funcional.
- Permet una millor organització i control dels serveis.

Cons:

- El rendiment pot ser inferior, ja que hi ha més sobrecàrrega en la comunicació entre capes.
- Definir clarament les capes pot ser complicat.

5.3 Estructura Microkernel

En una **estructura de microkernel**, només els serveis essencials (com la gestió de processos i memòria) s'implementen al nucli. Altres serveis, com la gestió de dispositius, s'executen en espais d'usuari separats.

Pros:

- El nucli és més petit, el que redueix la probabilitat d'errors i millora la seguretat.
- És més fàcil afegir o treure components sense haver de modificar el nucli principal.

Cons:

- El rendiment pot ser inferior perquè hi ha més comunicació entre el nucli i els serveis externs.

6. Exemples i Arquitectures Especials

6.1 Exemple: Linux

El nucli Linux és un dels projectes de codi obert més grans del món. La seva arquitectura és una combinació d'una estructura **monolítica híbrida basada en mòduls**, on el nucli bàsic és monolític però permet carregar i descarregar mòduls de manera dinàmica.

6.2 Màquines Virtuals

Les **màquines virtuals** (VM) com la **JVM** (Java Virtual Machine) permeten executar codi en qualsevol plataforma independentment del sistema operatiu subyacente.

6.3 Exokernels

Els **exokernels** són una arquitectura minimalista que ofereix als programes accés directe als recursos del maquinari. A diferència d'altres sistemes operatius, no intenten abstraure el maquinari, sinó proporcionar primitives bàsiques perquè els programes tinguin control total sobre els recursos.

6.4 Unikernels

Els **unikernels** empaqueten tot el codi necessari per executar una aplicació en un sol paquet. Aquest paquet pot executar-se directament sobre una màquina virtual o un hipervisor sense necessitat d'un sistema operatiu complet. Són molt eficients i petits perquè només inclouen el codi estrictament necessari.