CAPITOLO 1

#Sistemi_Operativi

Cosa è un sistema operativo?

Un sistema operativo è un software che si interfaccia con i componenti hardware della macchina e fornisce servizi fondamentali per l'esecuzione di programmi applicativi.

Un calcolatore moderno presenta componenti hardware complesse come uno o più processori, memoria principale, dischi e unità flash, periferiche I/O.

L'Hardware supporta due modalità ben distinte: modalità utente (quella a cui hanno diretto accesso anche le applicazioni), modalità kernel (ha accesso il SO che ha visione totale delle componenti hardware).

Il SO mette a disposizione l'hardware attraverso le chiamate di sistema
.

Tre prospettive per descrivere il SO:

- Astrazione : il SO si pone tra l'hardware e le applicazioni (l'obiettivo del SO è astrarre l'hardware).
- Visione top-down : il SO come astrazione dell'hardware per i programmi applicativi.
- Visione bottom-up : SO come gestore delle complesse componenti hardware, permettendo di allocare in modo controllato e ordinato le risorse.

Il SO permette di gestire le risorse per:

- eseguire più programmi in esecuzione.
- supportare più utenti.

Il multiplexing permette di mettere a disposizione delle risorse in modo condiviso, sia nel tempo (CPU) che nello spazio (memoria centrale, disco).

Processore

Esegue istruzioni dalla memoria. Il ciclo della CPU è : preleva (fetch), decodifica (decode), esegue (execute).

Registri importanti:

- Program Counter : indica l'istruzione successiva.
- IR : indica l'istruzione che viene eseguita.
- Stack Pointer : punta alla cime dello stack della memoria.
- Program Status Word (PSW) : contiene informazioni sullo stato del programma, fondamentale per le chiamate di sistema e I/O.

Multiplexing : il sistema operativo esegue programmi in modo efficiente.

Pipeline : esegue in parallelo istruzioni che possono essere eseguite a livello circuitale. Anche in caso di un'istruzione condizionale, la pipeline esegue anche l'operazione successiva pur di non fermarsi dall'eseguire istruzioni.

Più di un processore : più processori fisici o logici, multithreading.

Dispositivi di I/O

- Il controller : più semplice da usare per il SO, ogni controller ha bisogno di un driver.
- Il dispositivo : interfaccia elementare ma complicata da pilotare.

Driver : pezzetto di sistema operativo che non viene fornito con il sistema operativo, deve essere inserito all'interno del sistema operativo e avere tutti i diritti del sistema operativo. L'unico momento in cui il sistema operativo concede al driver tutti i diritti è al momento dell'avvio, infatti dopo aver installato un driver, il sistema operativo chiede all'utente di riavviare il sistema.

Il driver interagisce con il controller per :

- Eseguire l'I/O:
  - il processo esegue la chiamata di sistema
  - il kernel effettua una chiamata al driver
  - il driver avvia l'I/O.
- Interroga il dispositivo per vedere se ha finito oppure chiede al dispositivo di generare un interrupt quando ha finito.

Il DMA

DMA (Direct Memory Access) consente ai componenti di accedere direttamente alla memoria del computer senza coinvolgere la CPU, migliora l'efficienza ed aumenta le prestazioni nelle operazioni di I/O.

Buses

Dispositivi di legacy collegati a un processore hub separato. La CPU comunica con la memoria attraverso un bus veloce DDR4 , con una periferica grafica esterna sul bus PCIe , con tutti gli altri dispositivi attraverso un hub su un bus DMI .

Avvio del sistema

Quando si accende la macchina, si legge la ROM con le istruzioni per tutte le periferiche. Il BIOS esegue i comandi e dà retta al BOATLOADER che legge il sistema operativo in base alla periferica.

Chiamate di sistema

Le chiamate di sistema rappresentano l'interfaccia che il sistema operativo offre alle applicazioni per richiedere servizi. Le chiamate di sistema sono tendenzialmente racchiuse in librerie C, ogni chiamata di libreria rappresenta una specifica chiamata di sistema.

Quando un processo necessita di un certo servizio di sistema, per richiederlo deve eseguire una trap o un'istruzione che effettui una chiamata di sistema.
Se avviene una chiamata di sistema, si passa da modalità utente alla modalità kernel, all'interno della quale il SO può gestire correttamente e ordinatamente le risorse fornite dall'hardware senza comprometterne la funzionalità per risolvere la richiesta.

Struttura di un sistema operativo

Un sistema operativo può essere strutturato in due modi:

- modello monolitico : il kernel include tutte le funzionalità principali del sistema operativo (come gestione dei processi, memoria, file system,...). Ciò comporta buone prestazioni (le varie componenti possono comunicare vicendevolmente, chiamate di sistema senza intermezzi) ma un errore in una singola componente può compromettere l'intero sistema operativo.
- modello modulare : il SO è suddiviso in tre strati principali : modalità utente, modalità kernel e hardware. Il meccanismo di trap
  funge da interfaccia tra questi livelli.

Virtualizzazione

La virtualizzazione rappresenta una tecnologia che è in grado di creare una versione virtuale di una risorsa fisica (es. sistemi operativi, server etc...). Quando si parla di virtualizzazione, si fa tipicamente riferimento a macchine virtuali (VM) o a Container .

- VM rappresenta un imitazione completa di un sistema operativo che gira su hardware fisico. Poiché possono essere eseguite più VM all'interno della stessa macchina fisica e

poiché possono presentare sistemi operativi diversi dall' host, è necessaria la presenza di un Hypervisor affinché le risorse vengano condivise e assegnate correttamente.
- Container : rappresentano un metodo di virtualizzazione a livello di sistema operativo. Mentre le VM virtualizzano l'hardware, il container virtualizza il SO consentendo a più container di condividere il kernel del SO mantenendo le applicazioni isolate.

Quindi con i container : kernel in comune ma modalità utente indipendenti, rappresentano un approccio più leggero rispetto alle VM (dove viene simulato anche l'hardware) ma comportano un limite per cui ogni container non può presentare sistemi operativi troppo diversi da quello dell'host.

Kernel

Tre principali approcci nella progettazione del kernel di un sistema operativo:

- Exokernel : si tratta di un approccio la cui idea di base, al contrario dell'approccio tradizionale (monolitica) è fornire maggiore controllo da parte delle applicazioni verso le componenti hardware. Viene quindi ridotta notevolmente l'attrazione top-down da parte delle applicazioni verso l'hardware, e quindi maggiore complessità per gli sviluppatori (tuttavia, se ben scritte, applicazioni efficienti, libere e controllate appieno).
- Unikernel : sistemi progettati al fine di "specializzarsi" sull'eseguire e supportare una singola applicazione (o comunque un numero ridotto di specifiche applicazioni).
- Microkernel : approccio per cui si vuole minimizzare la quantità di codice eseguito in modalità kernel. Il sistema è diviso in piccoli moduli, ognuno dei quali funziona come processo separato. Mentre una piccola parte esegue funzioni essenziali in modalità kernel, il resto è lasciato alla modalità utente. Questo approccio comporta talvolta prestazioni peggiori rispetto a quello tradizionale, ma maggiore sicurezza e stabilità.