Linux & Windows

Da Unix a Linux:

Il primo Unix i primi elaboratori multiutente con processi in parallelo DTSS, CTSS , MULTICS e UNICS. Nel 1979 dopo l'introduzione del C nasce la versione più famosa, Unix v7 con novità motro importanti come:

- La portabilità dei programmi: programmi ad alto livello con compilatore per elaboratori diversi e minore dipendenza dall'architettura hardware
- Diversificazione degli idiomi: System V con xeinx di Microsoft e BSD con *virtual memory e TCP/IP*.

UNIX/LINUX:

I processi: sono la principale entità attiva nel sistema inizialmente sequenziali poi diventati concorrenti. Per creare un processo si utilizza fork() che genera un processo figlio e possono scambiarsi messaggi tramite i comandi pipe o signal. Appena creati i processi figli hanno la stessa memoria del genitore se poi fanno delle modifiche vengono separate. Un processo ha più flussi di controllo interni detti thread. I thread a differenza dei processi figli non sottostanno alle politiche di concorrenza ma stanno nel programma chiamante, inoltre condividono tutto le risorse e logiche e fisiche quindi anche variabili con i genitori. I thread si terminano da soli quando terminano il lavoro. UNIX dispone della tabella dei processi che resta in RAM e contiene informazioni riguardanti tutti i processi: Parametri di ordinamento(priorità, tempo di esecuzione ecc),Descrittore della memoria virtuale del processo, Lista segnali identificativi del loro stato e gruppi di appartenenza. Questo in figura 27 `e un esempio di shell che genera un processo figlio. Una volta creato il figlio `e come se i codici si doppiassero ma PID assume valori diversi per passaggio e nel if il figlio esegue un codice diverso. Con fork() si duplica il processo chiamante creando un processo figlio uguale ma distinto e se il padre aveva più thread ci sono 2 possibilit`a: O i thread vengono clonati tutti(molto complesso) o un solo thread viene clonato (non rispetta del tutto le richieste). I thread sono gestiti dal nucleo del S/O, ordinamento avviene per task (ovvero thread o processi) e seguono 3 classi di priorità FCFS a priorità senza prerilascio, a tempo reale con RR a priorità (prerilacio per quanti) o divisione di tempo RR a priorità. All'avvio parte il BIOS che carica l'MBR (Master Boot Record) poi MBR una volta eseguito carica il programma di boot dal corrispondente blocco. In questa fase di inizializzazione parte il processo 0 che configura orologi installa il FS e crea i processi 1 e 2. Il processo 1 configura il lato utente quindi avvia shell e il processo 2 chiamato *getty* che configura il prompt di login per l'utente.

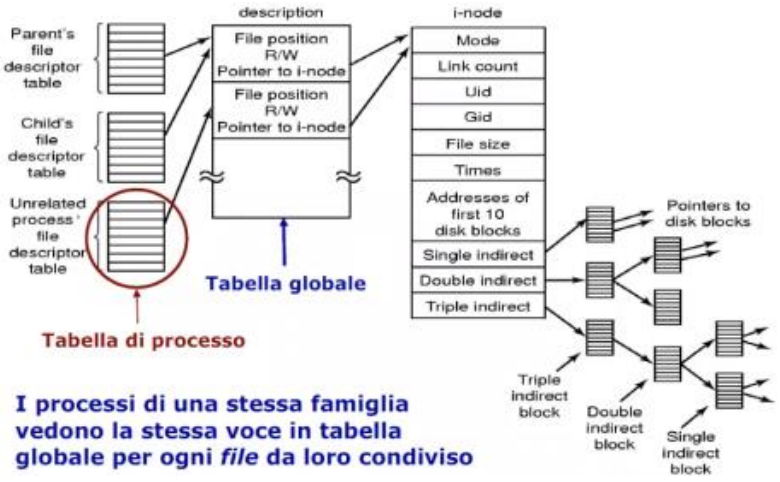
Gestione della memoria: in Unix è pensata per renderlo il più portabile possibile, ogni processo a un proprio spazio in memoria virtuale e il segmento dei dati varia a seconda dell'attività del programma. Lo stack contiene l'ambiente di esecuzione mentre lo swapper(gestore) si occupa di salvare su disco i processi sospesi con priorità minore. Solo in seguito venne aggiunta la paginazione a richiesta con *pagefault;* il *page deamon* verifica che ci sia sempre un n-lots free di pagine libere, la rimozione delle pagine avviene con la *second chance* ovvero se la pagina è stata riferita ovvero ha r-bit uguale a 1 viene rimessa in coda mentre se è a zero viene rimossa. La RAM è allocata in maniera dinamica e variabile. L'algoritmo di allocazione primario per cui ogni richiesta di ampiezza N è arrotondata a $\boldsymbol{2^m => N}$

Gestione degli I/O: Gli vengono trattati come dei file speciali con una posizione speciale nel File System e ad ogni dispositivo è associato un driver diverso. Linux invece permette di caricare dinamicamente moduli di gestione dei dispositivi(in quanto su Unix si dovrebbe ricompilare l'intero nucleo per cambiare la configurazione). Per la connessione di rete e protocolli si usa il socket, è un file che comunica con un alto socket associato ad un indirizzo diverso(porta di rete); le connessioni

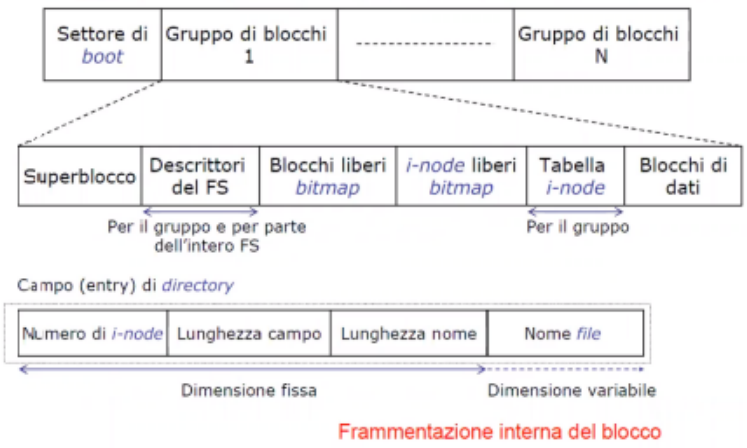
avvengono con due protocolli, TCP che garantisce la trasmissione completa dei dati (a blocchi o senza) o con UDP più veloce ma senza garanzia di trasmissione.

File System: un file è una sequenza di byte definita da un cammino(path) assoluto(rispetto alla radice) o relativo(rispetto a dove ci troviamo). Il controllo degli accessi concorrenti viene fatto mediante lo standard POSIX mediante meccanismi dedicati: 2 a grana grossa per file o directory e semafori o a grana fine per gruppi di byte in un file. L'accesso è di tipo *shared lock*(accesso simultaneo condiviso) che consente l'accesso alla stessa zona o *l'exclusive lock* che consente solo un accesso per zona. Per operare sui file si usano i comandi:

- *Iseek:* fissa l'indice di posizione di un file.
- *Stat:* fornisce info sul file consultando il suo i-node.



Su Unix le partizioni sono composte da un super blocco che indica i blocchi del File System ed il numero di i-node e contiene anche il puntatore alla lista dei blocchi liberi. Gli i-node sono ordinati in un elenco da 1 a n mentre le directory sono viste come un insieme di variabili non ordinato di informazioni.



Il nucleo usa due strutture di controllo: una tabelle che i descrittori utente dei file attualmente in uso di ciascun processo. E un tabella globale che mantiene la corrispondenza tra tutti i file aperti ed i rispettivi i-node.

Il file system in Linux era basato su quello di Minix ma poi fu sostituito perché troppo limitante, quindi si passo a ext2 che di innovativo aveva la possibilità di suddividere la partizione in gruppi di blocchi e di caricare dinamicamente i moduli di gestione dei dispositivi.

WINDOWS:

Genesis: Windows nasce da MS-DOS Microsoft Disk Operating System che era mono utente monoprogrammato e solo command line, con windows vien introdotta la GUI.

GUI: Grapihical User Interface fu introdotta per la prima volta da Macintosh di Apple che implementava icone ,finestre, menu ecc. Windows di 2a generazione era un vero e proprio S/O multiprogrammato ma sempre monoutente con FS FAT. Con Windows di 3a generazione parte il progetto NT ovvero l'abbandono di MS-DOS e si sviluppano sicurezza e affidabilità e viene introdotto un nuovo FS ntfs. Inoltre si provò a introdurre un sistema a microkernel e modello client-server che permetteva un elevata portabilità ma una bassa velocità per questo si ritornò a un nucleo monolitico. Nel 2001 esce Windows XP con delle migliorie grafiche e fu un successo e poi tornò nel 2007 con Windows Vista che fù un flop per la sua lentezza e pesantezza. Nel 2009 (poco dopo a causa del flop)esce Windows 7 con alcuni miglioramenti di efficienza e grafici. Nel 2012 esce Windows 8 che permette l'integrazione con altri dispositivi e miglioramenti di efficienza energetica.

Windows Interface: Per la programmazione mette a disposizione Win32 API Application Programming Interface che include alcune chiamate di sistema e altre svolgono servizi e utilità. Tutte le informazioni vitali di configurazione sono raccolte in una specie di FS detto REGISTRY Salvato su disco in un file speciale. Le directory corrispondono alle key e i file alle entry. Inoltre esistono 6 directory principali con prefisso HKEY (+ trattino basso) con entry descrittive dell'hardware e periferiche (HARDWARE) programmi (SOFTWARE)e informazioni di installazione (SYSTEM).

Windows Gestione processi:

- Processo possessore di risorse con più di un thread
- Job Processo gestito come singola unità e con limiti di risorse (Contiene istruzioni)
- Thread Flusso di controllo gestito dal nucleo eseguito e risiedente nell'ambiente del processo.
- Fiber Suddivisione di thread ignota al nucleo

I thread possono essere sincronizzati tra loro tramite oggetti di ordinamento, semafori binari e sezioni critiche.
I thread hanno anche vari modi di comunicare
senza sincronizzarsi:

- Pipe Canali bidirezionali a sequenza di byte senza struttura
- Mailslot Canali unidirezionali anche su rete
- Soket Come le Pipe ma per le comunicazioni remote
- RPC (Chiamata di procedura remota) Per invocare procedure nello spazio di altri processi e riceverne il risultato localmente
- Condivisione memoria Usando file in memoria

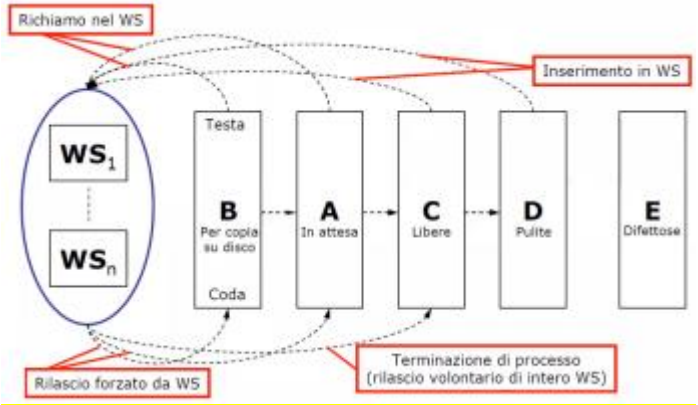
Politiche di ordinamento: Windows usa la politica di ordinamento con pre-rilascio a priorità. Ci sono quindi thread con priorità maggiore che vengono eseguiti prima e possono essere pre-rilasciati. Ogni processo ha 6 classi di priorità invece ogni thread ne ha 7. Ci sono 32 livelli di priorità in cui ciascuno viene associato a una coda di thread. Le classi dei processi vanno da realtime a Idle e i livelli di priorità da *Time critical a Idle* perché i processi con priorità maggiore non creino starvation si adotta una gestione dinamica della priorità. Ogni processo ad alta priorità dopo essere stato in esecuzione per molto tempo perde priorità al termine del quanto mentre un processo di I/O guadagna priorità quando completa un operazione di I/O. Ogni processo ha uno spazio di indirizzamento virtuale paginato ampio 4GB suddiviso in 2 zone. Su una pagina virtuale ci sono 3 operazioni lettura scrittura e esecuzione e inoltre una pagina può essere:

- Libera: Non riferita.
- Assegnata: In uso per codice o dati.
- Prenotata: Non ancora in uso, ma non libera.

Trattamento delle pagine in windows:

l'accesso alle pagine di un file mappato in memoria può essere condiviso da più processi(per esempio il DDL) quindi bisogna gestirlo, anche il sistema operativo è visto come un processo con un proprio working set e pagine rimpiazzabili. Per verificare che ci siano sempre delle pagine libere il *deamon* attiva un thread del memory manager che libera le pagine necessarie, le pagine possono essere in uso se appartiene ad un working space oppure rilasciabile se appartiene a una di queste tipologie: in attesa, da copiare su disco?, libera, azzerata o difettosa?.

Lo swapper thread (deamon) del Memory manager porta le pagine dello stack dei processi in cui i thread siano stati recentemente inattivi.

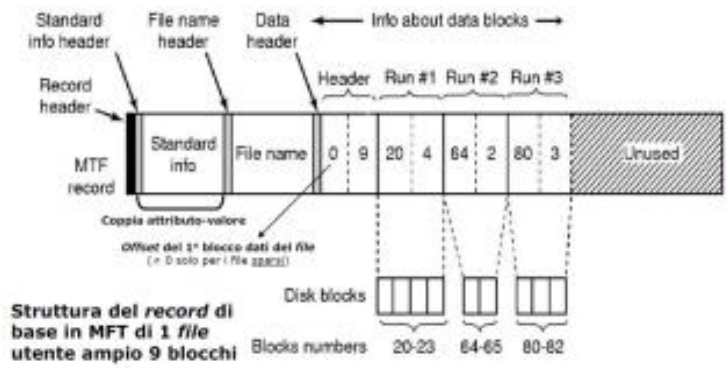


File System in windows:

Da windows NT e poi vista e XP si comincio ad utilizzare NTFS come file system, questo comporta diversi vantaggi rispetto al FAT come:

- Nome file fino a 255 caratteri.
- I file sono visti come aggregati di attributi.
- File system ad architettura gerarchica per indicare il cammino e supporto sia per hard-link che simbolic-link.

NTFS è una collezione di volumi, ogni volume è una sequenza di blocchi di ampiezza fissa e ha un proprio MFT(struttura dati) ogni record contiene una copia di descrittori che specificano la struttura dell'attributo e il suo valore. Un file è messi preferibilmente in blocchi contigui per semplificare la sua lettura è minimizzare l'uso di record, quando un file necessita di più record nel record base si mettono i puntatori ad altri(modello i-node).



Struttura del *record* di base in MFT di 1 *file* utente ampio 9 blocchi