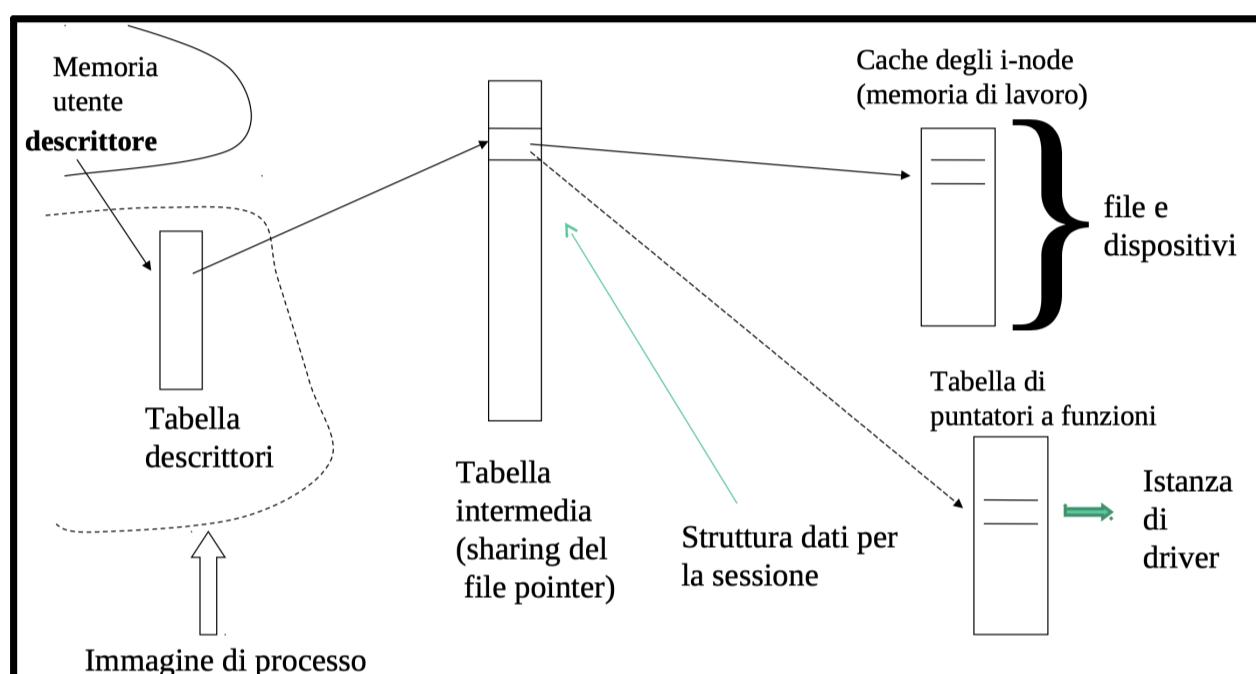
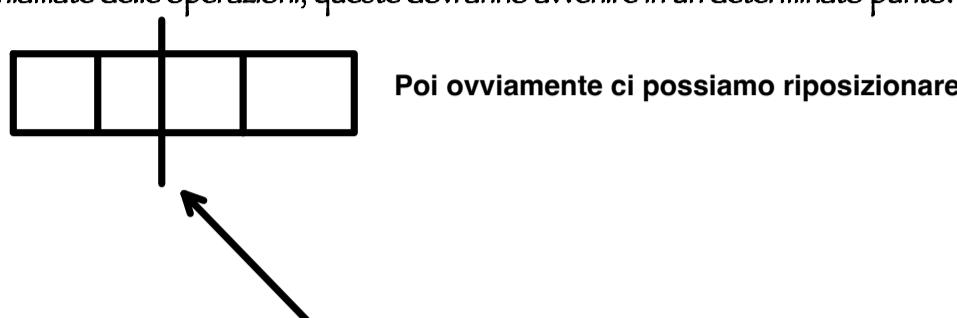


UNIX virtual file system (I)

Quando noi lavoriamo ad un'applicazione attiva e sappiamo che per arrivare a lavorare su un oggetto di I/O dobbiamo passare un descrittore, ossia un codice di canale, e sappiamo che per questa applicazione esiste una tabella dei descrittori di canali che ci può portare a lavorare su un oggetto di I/O, sappiamo che la tabella ci porta in un'informazione intermedia che è l'informazione di sessione. All'interno della tabella intermedia abbiamo ad esempio il file pointer, che quindi va ad indicare al virtual file system in che punto di quell'oggetto di I/O (file) deve avvenire la prossima operazione.

È legato alla possibilità di avere accesso diretto su UNIX.

Questo significa che se noi stiamo lavorando su un file unix, all'interno di quell'elemento nella tabella viene indicato al vfs che quando verranno chiamate delle operazioni, queste dovranno avvenire in un determinato punto.



Poi per poter gestire correttamente il file, all'interno di questa informazione di sessione noi dobbiamo sicuramente mantenere un'informazione che ci porta ad identificare in un'altra tabella qual è il driver che va veramente chiamato quando l'applicazione sta cercando di fare qualcosa.

L'applicazione chiama per esempio una system call di lettura di dati presenti all'interno di un file, la lettura avverrà caricando un blocco del dispositivo da memoria di massa all'interno del buffer cache, prelevando alcuni byte che caratterizzano il contenuto di quel file, e copiarli da qualche parte all'interno della memoria utente.

Ma se noi chiamiamo una read su un oggetto che non è fatto così, magari PIPE/socket, il driver deve fare operazioni differenti.

Dobbiamo sapere l'istanza da usare a seconda dell'oggetto di I/O che stiamo gestendo: questo lo dice la sessione.

Ma nella sessione teniamo traccia di un'altra CACHE software dove è presente l'i-node che noi associamo all'oggetto di I/O su cui stiamo lavorando. Attenzione: se l'oggetto di I/O su cui stiamo lavorando è un file, chiaramente li dentro viene caricato a partire dal disco rigido, l'i-node del file su cui vogliamo lavorare. Ma ovviamente se questo oggetto di I/O non è un file regolare, verrà popolato all'interno di una entry della cache, un i-node associato ad un oggetto che è l'oggetto gestito dal driver che in qualche modo sarà differente per quanto riguarda le operazioni che realmente vengono eseguite dal VFS quando vengono chiamate queste System Call per andare a lavorare in I/O.

La cosa importante è che le applicazioni per arrivare sugli oggetti di I/O, devono arrivarci con un codice numerico, che è il canale di I/O che ci porta ad identificare tramite la tabella dei descrittori, la sessione di I/O che ci porta poi su TUTTO. Codice che va eseguito, metadati basici per gestire l'oggetto di I/O.

56:05

- i dispositivi vengono gestiti come file
- ogni i-node del file system virtuale viene associato o ad un file o ad un dispositivo
- le funzioni realmente eseguite su richiesta delle applicazioni dipendono dall'entità associata all'i-node relativo al descrittore per il quale viene invocata la funzione
- gli i-node associati ai file sono riportati su hard-drive allo spegnimento (shutdown) se non prima
- gli i-node associati ai dispositivi possono essere rimossi allo spegnimento (i-node dinamici)
- la cache degli i-node serve per gestire i-node dinamici e per rendere l'accesso ai file più efficiente
- esiste un buffer-cache per tenere temporaneamente i blocchi di dati relativi ai file in memoria di lavoro

