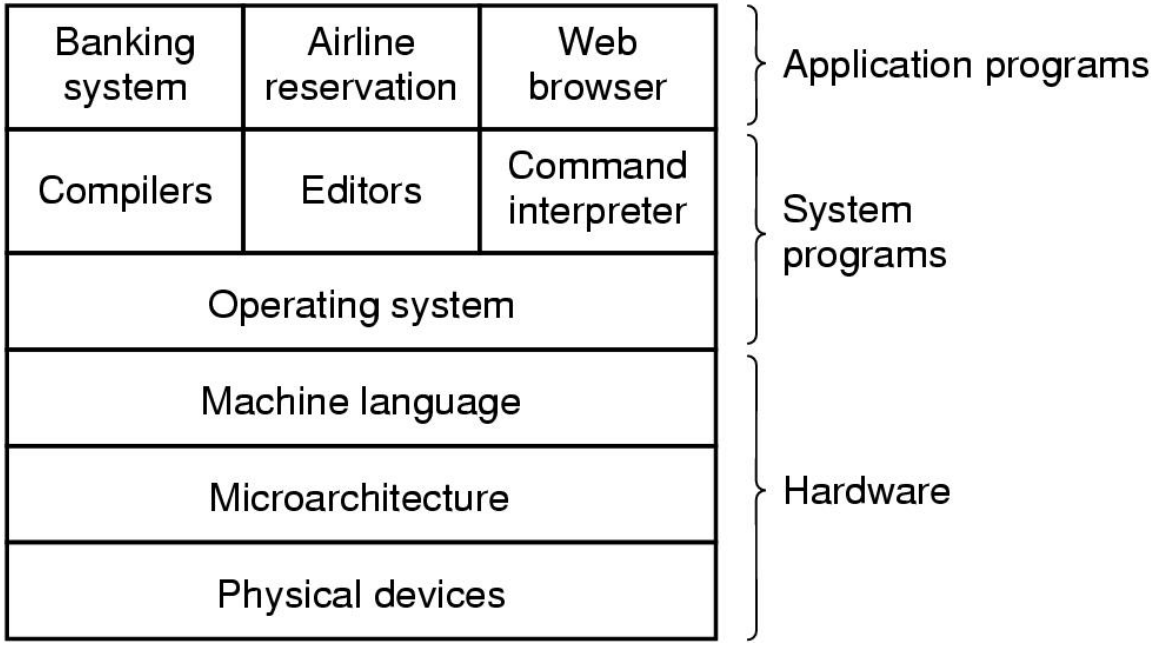
Introduzione

- Cosa è un Sistema Operativo?
- Una breve storia
- Alcuni concetti di base
- Organizzazione di un SO

Introduzione



Un sistema di elaborazione può essere visto come un insieme di livelli di astrazione sovrapposti

– hardware, prog. di sistema, prog. applicativi

Cos'è un sistema operativo

- Una macchina estesa
  - nasconde i dettagli complessi dell'hardware
  - presenta agli utenti una *macchina astratta*, più facile da usare
- Un gestore di risorse
  - permette a più programmi di condividere le risorse della macchina (stampanti, dischi …)
  - cerca di sfruttare al meglio le risorse presenti

Storia dei Sistemi Operativi (1)

- Prima generazione 1945 - 1955
  - valvole, schede a spinotti
- Seconda generazione 1955 - 1965
  - transistor, sistemi batch
- Terza generazione 1965 – 1980
  - circuiti integrati, multiprogrammazione
- Quarta generazione 1980 – oggi
  - personal computer

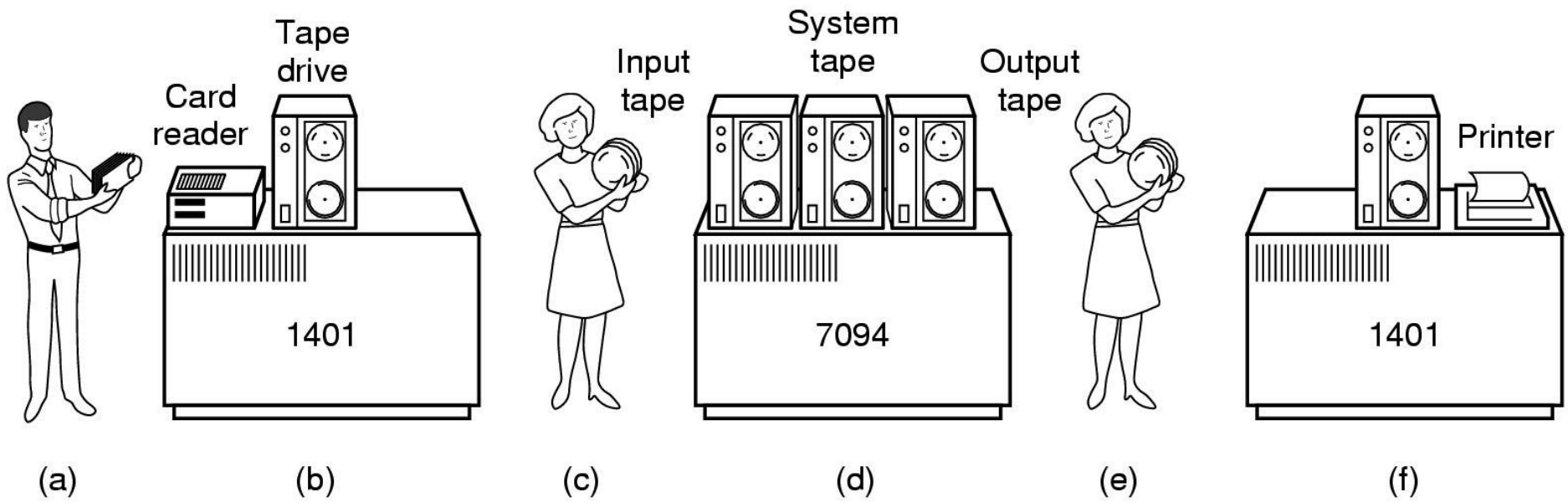
Storia dei Sistemi Operativi (2)

- Prima generazione 1945 - 1955
  - valvole, macchine enormi, poco affidabili
  - un solo team costruiva, programmava, manuteneva la macchina
  - programmi in linguaggio macchina assoluto
  - no SO, no assembler
  - ciclo di utilizzo
    - prenotazione
    - inserimento scheda a spinotti
    - calcolo sotto la supervisione del programmatore
  - appl tipiche: tabelle matematiche

Storia dei Sistemi Operativi (3)

- Seconda generazione 1955 - 1965
  - transistor, sistemi commerciali
  - molto grossi e costosi (banche, università..)
  - assembler, FORTRAN
  - primi SO
    - un *job* in esecuzione alla volta fuori linea
    - l'operatore caricava le schede perforate relative al programma e ritirava le stampe
  - inefficiente

Storia dei Sistemi Operativi (4)



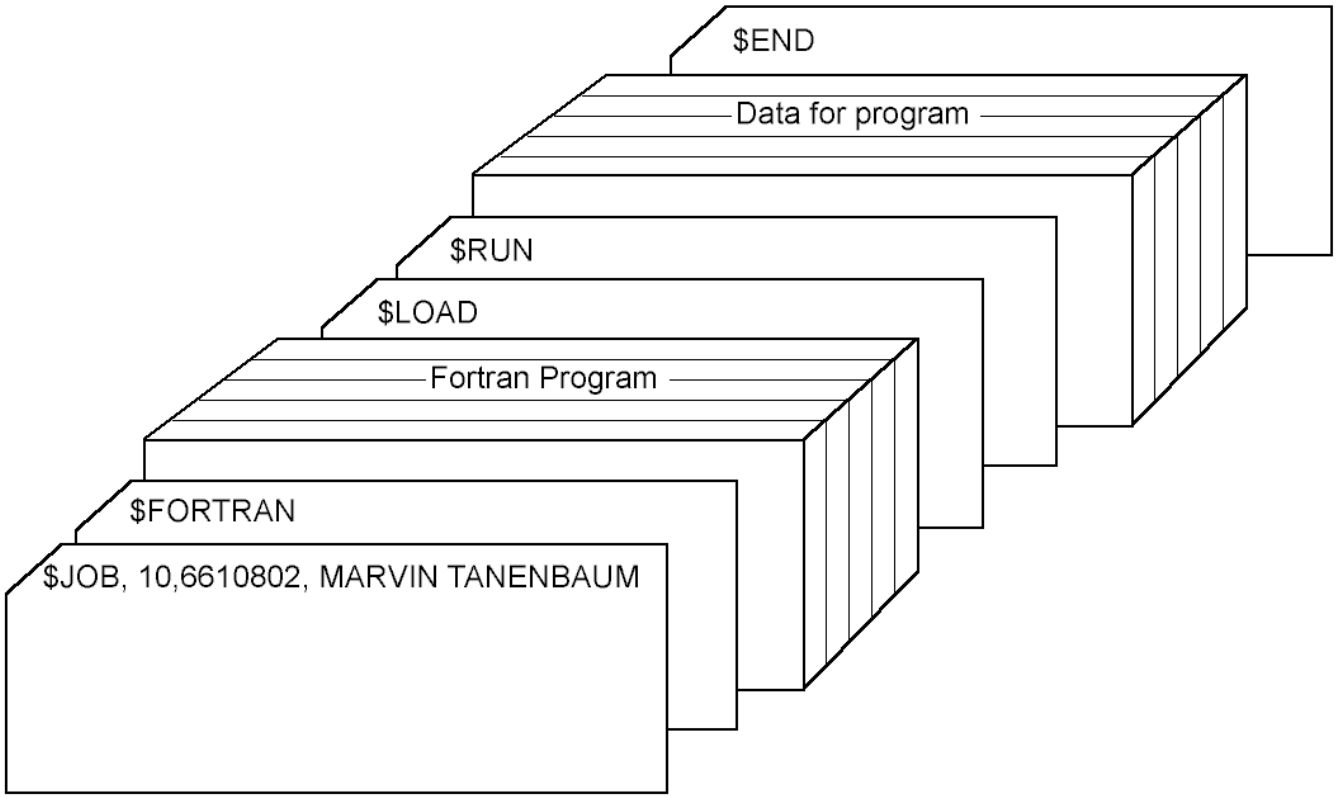
Un esempio di Sistema Operativo *Batch (a lotti)*

(a,b) le schede relative a un gruppo di programmi vengono lette da un computer specializzato (1401) e trasferite su nastro (*tape*)

(c,d) il nastro di input viene trasportato su un 7094, che effettua il calcolo e produce un nastro di risultati

(e,f) il nastro dei risultati complessivi viene stampato da un 1401

Storia dei Sistemi Operativi (5)

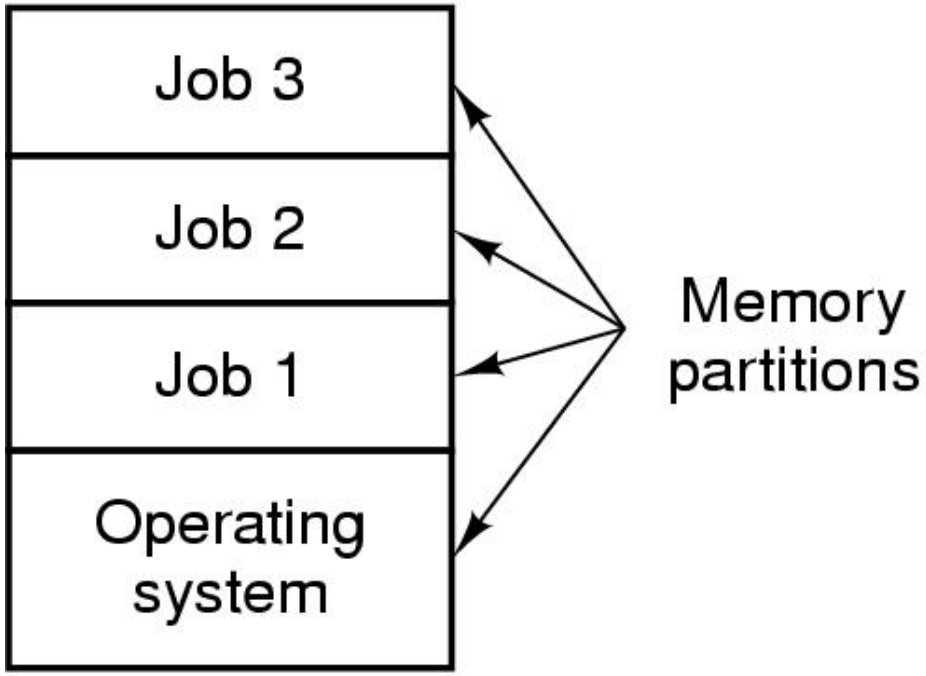


- Struttura di un tipico *job* in un sistema operativo batch (FMS –Fortran Monitor System)

Storia dei Sistemi Operativi (6)

- Terza generazione 1965 - 1980
  - IC su bassa scala
  - computer largamente diffusi
  - diversi linguaggi di programmazione
  - SO molto complessi, diretti predecessori dei sistemi attuali
  - multiprogrammazione
    - OS/360
  - timesharing
    - CTSS, MULTICS, UNIX ..

Storia dei Sistemi Operativi (7)

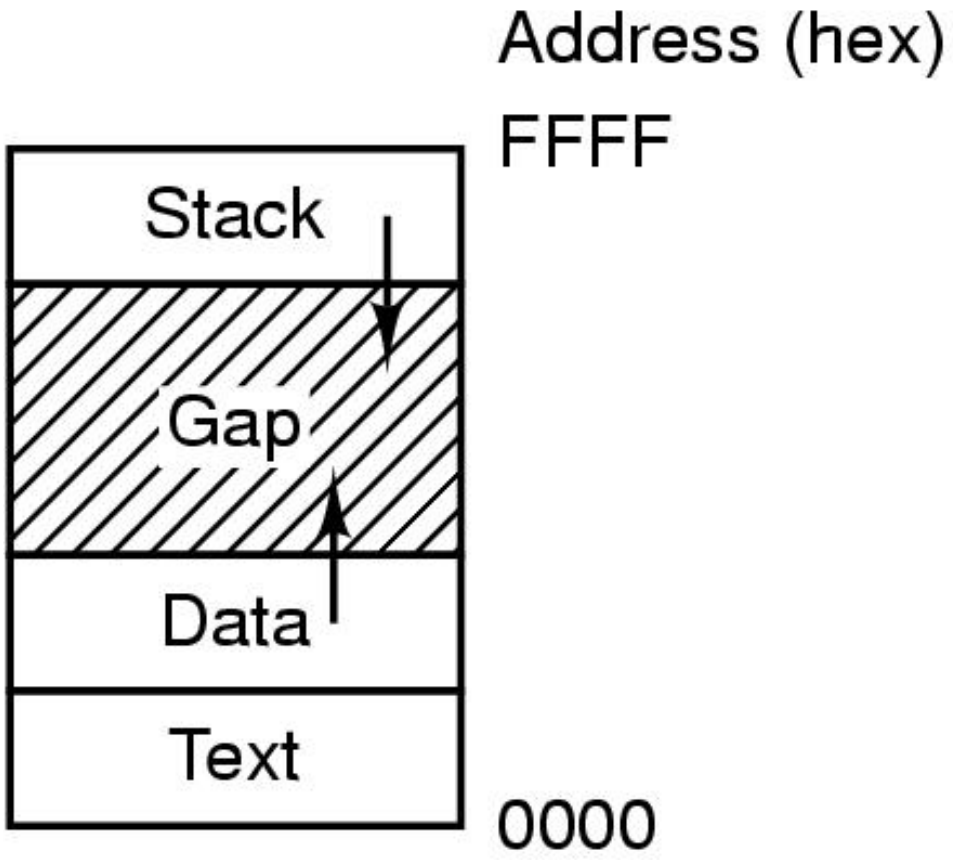


- Un sistema multiprogrammato (3a generazione)
  - tre job distinti coesistono in memoria centrale
  - sovrapposizione di calcolo ed I/O

Gestione di più job/processi in memoria centrale (1)

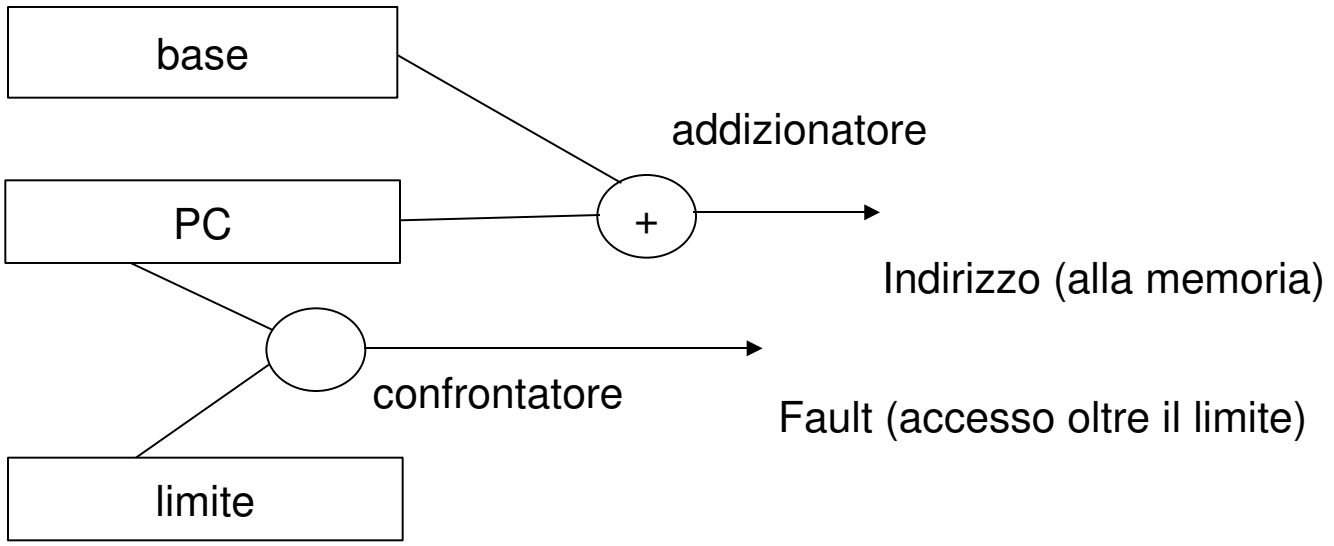
- Protezione da comportamenti erronei o utenti malintenzionati
  - Necessità di proteggere sia i job che il sistema operativo
- Problema della rilocazione
  - il compilatore e il linker assumono che il programma venga caricato all'indirizzo 0
  - dobbiamo assicurare il funzionamento corretto a partire dall'indirizzo a cui viene caricato (es. 16K)
    - indirizzi di istruzioni e dati devono essere incrementati (*rilocazione*)
  - rilocazione statica/dinamica

Spazio di indirizzamento di un processo



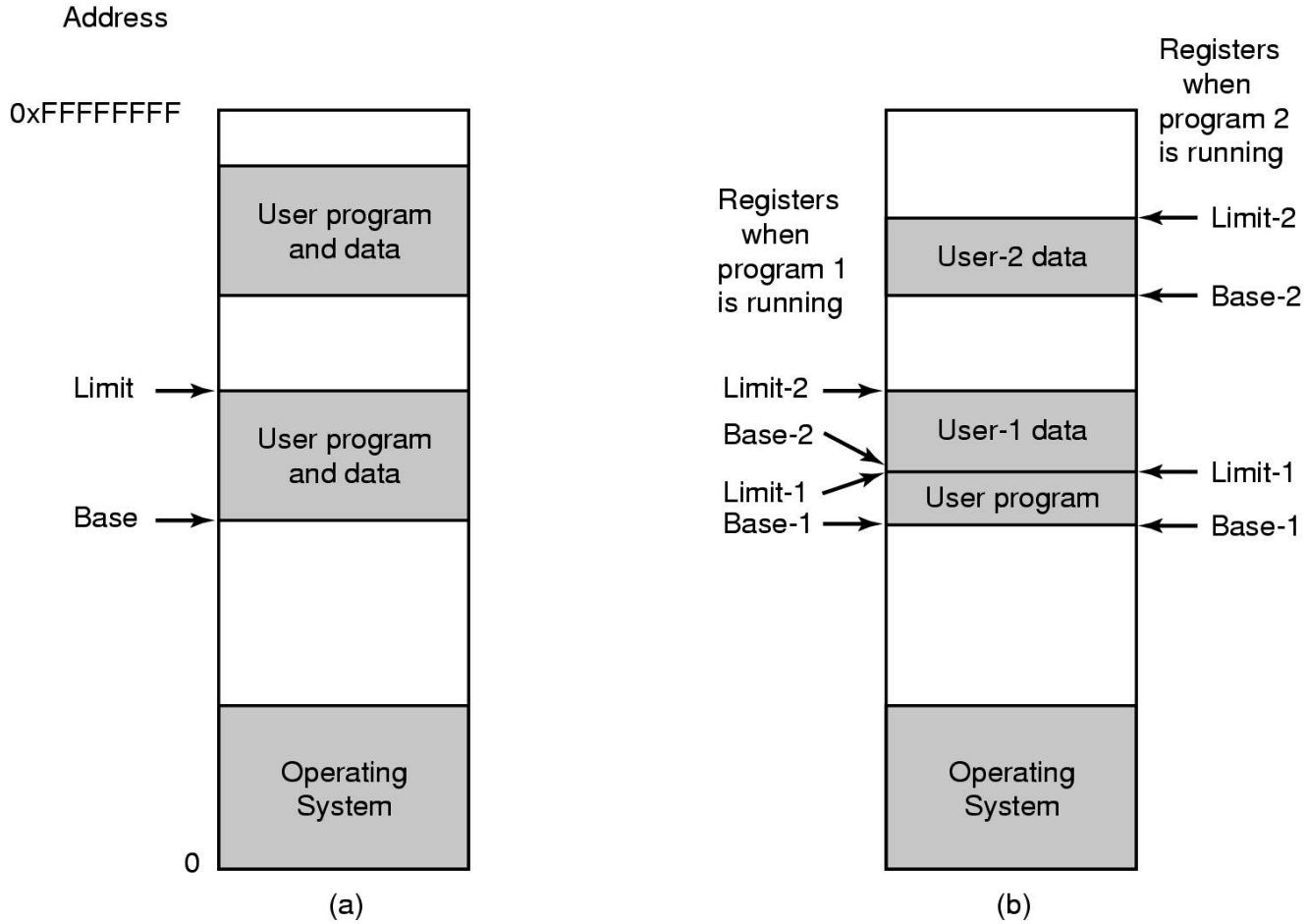
- Tre segmenti: testo, dati (statici e *heap*) e *stack*

Gestione di più job/processi in memoria centrale (2)



- Due registri: *base* e *limite*
- *base*: indirizzo iniziale del programma
- *limite*: ampiezza spazio indirizzi

Gestione di più job/processi in memoria centrale (3)

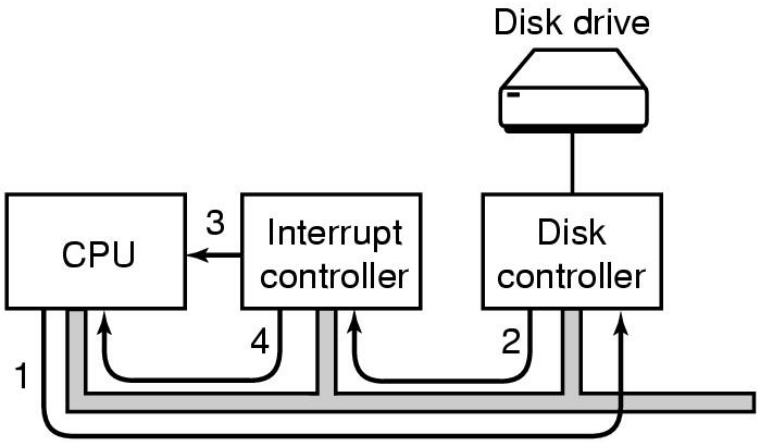


- *base* e *limite* sono gestiti dal SO
- Con due coppie di registri è possibile condividere il segmento testo

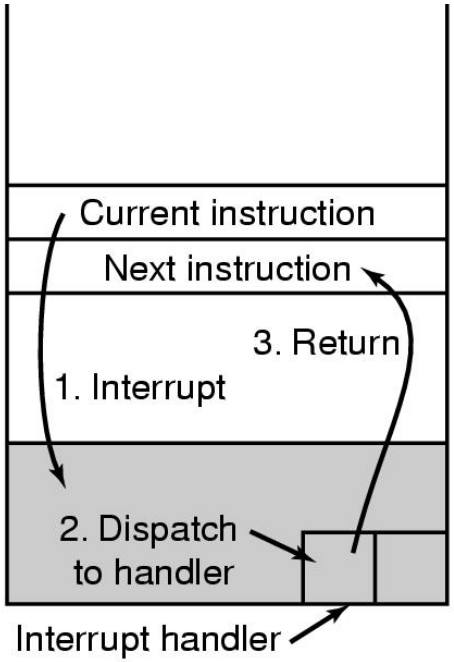
Gestione di più job/processi in memoria centrale (4)

- Protezione dei registri da accessi erronei/intenzionali
- Due stati di funzionamento: modo utente e modo supervisore (*kernel*)
- L'accesso completo all'hardware solo in modo kernel
- Passaggio controllato da modo utente a modo kernel
  - interruzioni
  - chiamate di sistema (system call)

Interruzioni (1)



(a)



(b)

(a) passi necessari per inizializzare un dispositivo e ricevere una interruzione

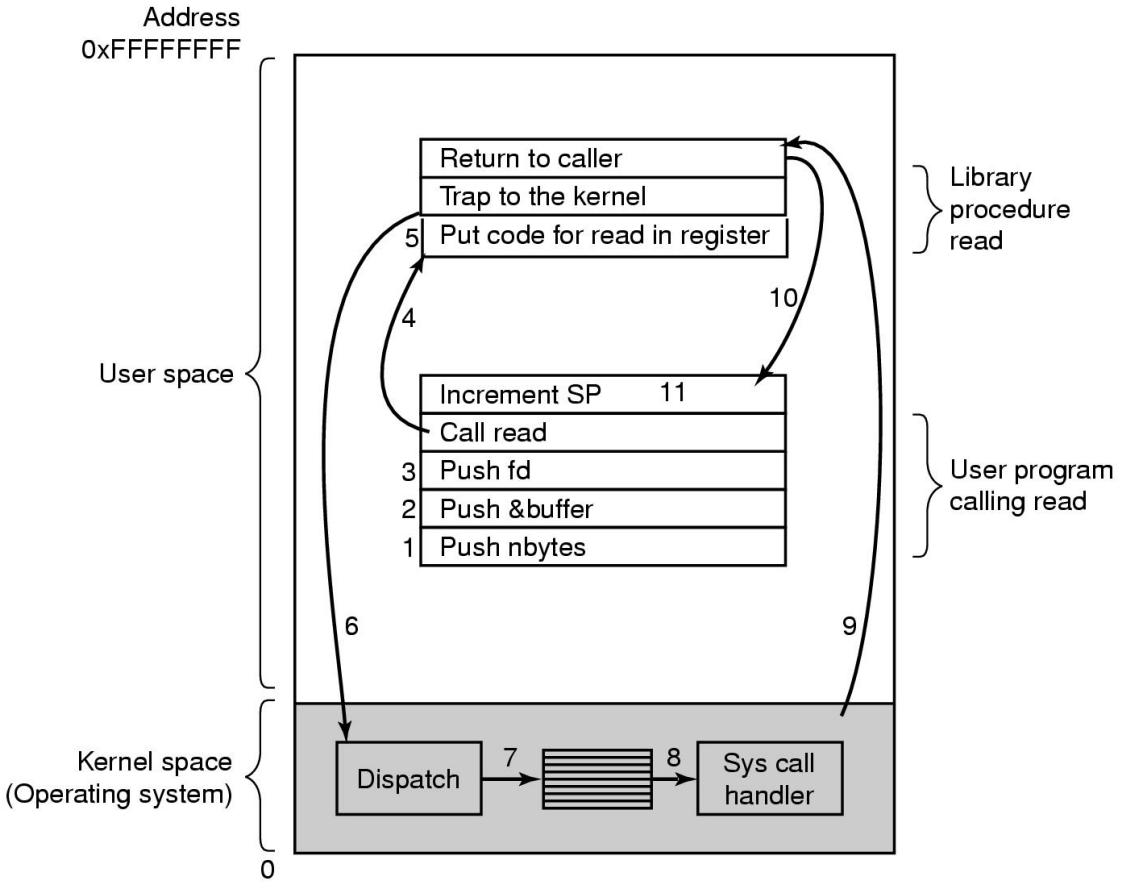
(b) come viene interrotta la CPU

Interruzioni (2)

Passi necessari alla gestione di una interruzione:

1. CPU rileva l'interruzione e legge l'indirizzo del dispositivo (*device*) su bus
   - salvataggio PC/PSW sullo stack, passaggio in modo kernel
2. L'indirizzo del dispositivo viene usato come indice nella tabella dei gestori delle interruzioni (*interrupt vector*)
3. Il gestore selezionato prende il controllo e svolge le operazioni necessarie
4. Quando il gestore termina
   - ripristino PC/PSW , ritorno in modo utente
   - si esegue l'istruzione successiva a quella interotta

Chiamate di Sistema (*System Call*) (1)



Gli 11 passi necessari per effettuare la chiamata di sistema `read (fd, buffer, nbytes)`

Chiamate di Sistema (*System Call*) (2)

COUNT= READ (FD, BUFFER, NBYTES)

Dettaglio dell'esecuzione di read (fd, buffer, nbytes)

1-3. Salvataggio dei parametri sullo stack

4. Chiamata della funzione di libreria read()

5. Caricamento del codice della system call in un registro fissato Rx

6. Esecuzione TRAP
   - passaggio in kernel mode, salto al codice del dispatcher

7-8. Selezione della SC secondo il codice in Rx

9. Ritorno alla funzione di libreria
   - ripristino user mode, caricamento PC

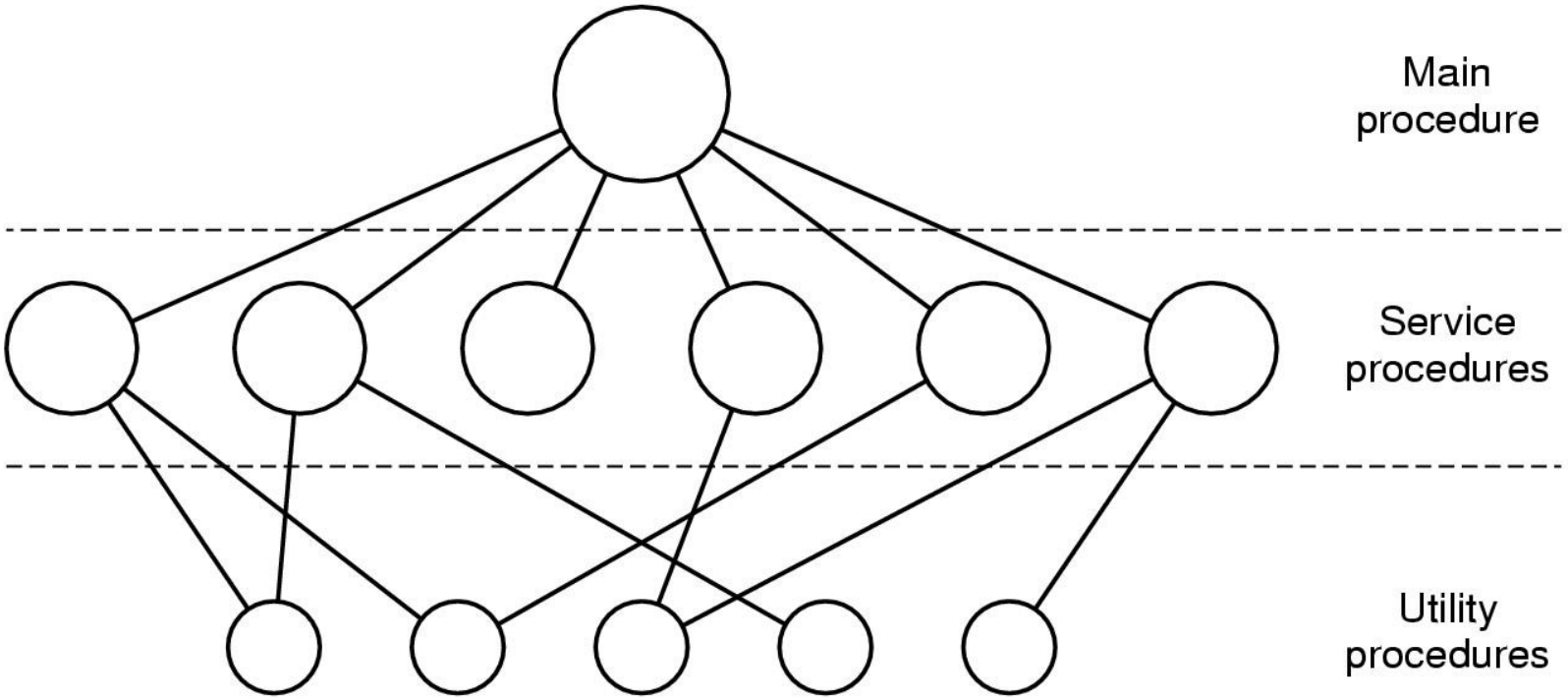
10-11. Ritorno al codice utente (nel modo usuale)

Struttura di un sistema operativo (1)

Sistemi monolitici:

- insieme di procedure compilate in un unico oggetto
- ogni procedura può chiamare tutte le altre/ha visibilità delle SD
- ogni processo esegue parzialmente in modo kernel
- system call bloccanti
- i sistemi Unix/Linux e Windows sono monolitici

Struttura di un sistema operativo (2)



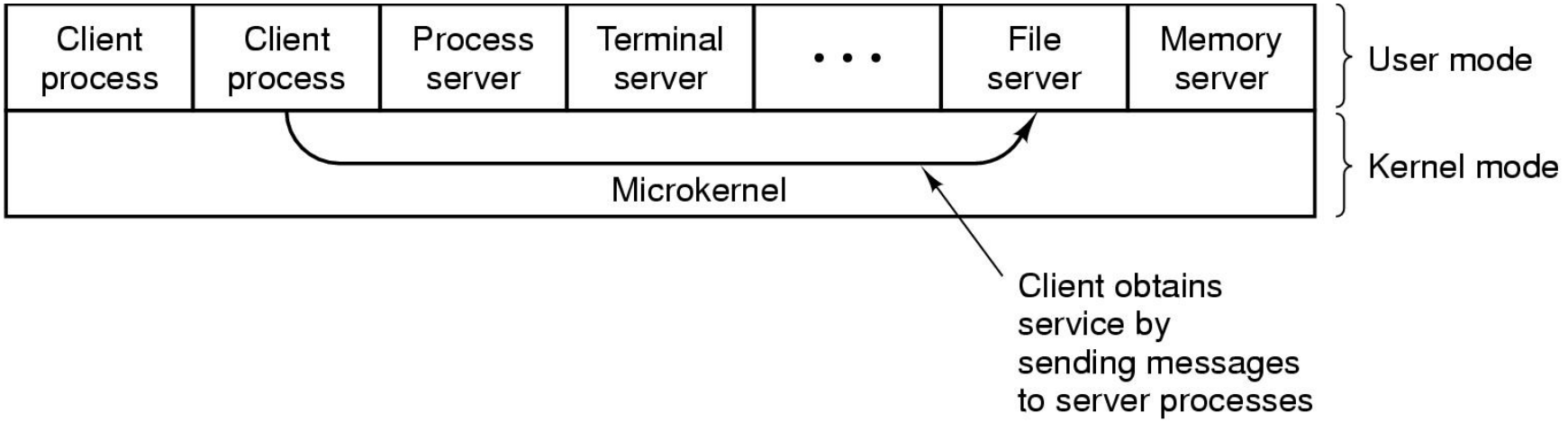
Un tipico sistema monolitico

Struttura di un sistema operativo (3)

Il modello client-server:

- minimizza le funzioni del SO che girano in modo kernel
- molte funzioni sono realizzate da processi *server* che girano in modo utente
- nucleo minimo (*microkernel*) :
  - funzioni base per la gestione dei processi e lo scambio dei messaggi
  - comunicazione con i dispositivi vista come messaggi "speciali"
- l'attesa avviene fuori dal kernel

Struttura di un sistema operativo (4)



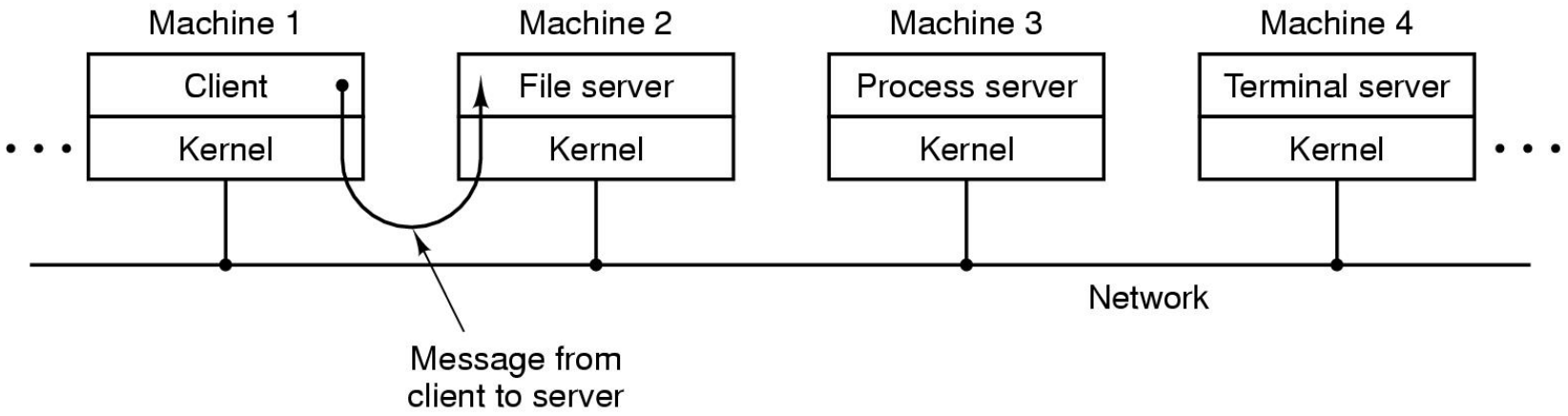
Il modello client-server

Struttura di un sistema operativo (5)

Client-server vs modello monolitico

- più sicuro
- meno efficiente
- si adatta bene ai sistemi operativi di rete
- Windows NT 3.0 adottava un modello ispirato al client/server (ibrido)
- studiato in ambito accademico
  - es: MACH, Minix

Struttura di un Sistema Operativo (6)



Il modello client-server in ambiente distribuito