Struttura dei Sistemi Operativi

- Componenti del sistema
- Servizi del Sistema Operativo
- Chiamate di sistema (*system calls*)
- Programmi di Sistema
- Struttura del Sistema
- Macchine Virtuali

Componenti comuni dei sistemi

1. Gestione dei processi
2. Gestione della Memoria Principale
3. Gestione della Memoria Secondaria
4. Gestione dell'I/O
5. Gestione dei file
6. Sistemi di protezione
7. Connessioni di rete (*networking*)
8. Sistema di interpretazione dei comandi

Gestione dei processi

- Un *processo* è un programma in esecuzione. Un processo necessita di certe risorse, tra cui tempo di CPU, memoria, file, dispositivi di I/O, per assolvere il suo compito.

- Il sistema operativo è responsabile delle seguenti attività, relative alla gestione dei processi:
  - creazione e cancellazione dei processi
  - sospensione e riesumazione dei processi
  - fornire meccanismi per
    * sincronizzazione dei processi
    * comunicazione tra processi
    * evitare, prevenire e risolvere i *deadlock*

Gestione della Memoria Principale

- La *memoria principale* è un (grande) array di parole (byte, words. . .), ognuna identificata da un preciso indirizzo. È un deposito di dati rapidamente accessibili dalla CPU e dai dispositivi di I/O.

- La memoria principale è *volatile*. Perde il suo contenuto in caso di *system failure.*

- Il sistema operativo è responsabile delle seguenti attività relative alla gestione della memoria:

  - Tener traccia di quali parti della memoria sono correntemente utilizzate, e da chi.

  - Decidere quale processo caricare in memoria, quando dello spazio si rende disponibile.

  - Allocare e deallocare spazio in memoria, su richiesta.

Gestione della memoria secondaria

- Dal momento che la memoria principale è volatile e troppo piccola per contenere tutti i dati e programmi permanentemente, il calcolatore deve prevedere anche una *memoria secondaria* di supporto a quella principale.

- La maggior parte dei calcolatori moderni utilizza *dischi* come principale supporto per la memoria secondaria, sia per i programmi che per i dati.

- Il sistema operativo è responsabile delle seguenti attività relative alla gestione dei dischi:
  - Gestione dello spazio libero
  - Allocazione dello spazio
  - Schedulazione dei dischi

Gestione del sistema di I/O

- Il sistema di I/O consiste in
  - un sistema di cache a buffer
  - una interfaccia generale ai gestori dei dispositivi (*device driver*)
  - i driver per ogni specifico dispositivo hardware (controller)

Gestione dei File

- Un *file* è una collezione di informazioni correlate, definite dal suo creatore. Comunemente, i file rappresentano programmi (sia sorgenti che eseguibili (*oggetti*)) e dati.

- Il sistema operativo è responsabile delle seguenti attività connesse alla gestione dei file:
  - Creazione e cancellazione dei file
  - Creazione e cancellazione delle directory
  - Supporto di primitive per la manipolazione di file e directory
  - Allocazione dei file nella memoria secondaria
  - Salvataggio dei dati su supporti non volatili

Sistemi di protezione

- Per *Protezione* si intende un meccanismo per controllare l'accesso da programmi, processi e utenti sia al sistema, sia alle risorse degli utenti.

- Il meccanismo di protezione deve:
  - distinguere tra uso autorizzato e non autorizzato.
  - fornire un modo per specificare i controlli da imporre
  - forzare gli utenti e i processi a sottostare ai controlli richiesti

Networking (Sistemi Distribuiti)

- Un *sistema distribuito* è una collezione di processori che non condividono memoria o clock. Ogni processore ha una memoria propria.

- I processori del sistema sono connessi attraverso una *rete di comunicazione.*

- Un sistema distribuito fornisce agli utenti l'accesso a diverse risorse di sistema.

- L'accesso ad una risorsa condivisa permette:
  - Aumento delle prestazioni computazionali
  - Incremento della quantità di dati disponibili
  - Aumento dell'affidabilità

Interprete dei comandi

- Molti comandi sono dati al sistema operativo attraverso *control statement* che servono per
  - creare e gestire i processi
  - gestione dell'I/O
  - gestione della memoria secondaria
  - gestione della memoria principale
  - accesso al file system
  - protezione
  - networking

Interprete dei comandi (Cont.)

- Il programma che legge e interpreta i comandi di controllo ha diversi nomi:
  - interprete delle schede di controllo (sistemi batch)
  - interprete della linea di comando (DOS, Windows)
  - shell (in UNIX)
  - interfaccia grafica: Finder in MacOS, Explorer in Windows, gnome-session in Unix. . .

  La sua funzione è di ricevere un comando, eseguirlo, e ripetere.

Servizi dei Sistemi Operativi

- Esecuzione dei programmi: caricamento dei programmi in memoria ed esecuzione.

- Operazioni di I/O: il sistema operativo deve fornire un modo per condurre le operazioni di I/O, dato che gli utenti non possono eseguirle direttamente,

- Manipolazione del file system: capacità di creare, cancellare, leggere, scrivere file e directory.

- Comunicazioni: scambio di informazioni tra processi in esecuzione sullo stesso computer o su sistemi diversi collegati da una rete. Implementati attraverso *memoria condivisa* o *passaggio di messaggi*.

- Individuazione di errori: garantire una computazione corretta individuando errori nell'hardware della CPU o della memoria, nei dispositivi di I/O, o nei programmi degli utenti.

Funzionalità addizionali dei sistemi operativi

Le funzionalità addizionali esistono per assicurare l'efficienza del sistema, piuttosto che per aiutare l'utente

- Allocazione delle risorse: allocare risorse a più utenti o processi, allo stesso momento
- Accounting: tener traccia di chi usa cosa, a scopi statistici o di rendicontazione
- Protezione: assicurare che tutti gli accessi alle risorse di sistema siano controllate

Chiamate di Sistema (System Calls)

- Le chiamate di sistema formano l'interfaccia tra un programma in esecuzione e il sistema operativo.

- Generalmente, sono disponibili come speciali istruzioni assembler

- Alcuni linguaggi pensati per programmazione di sistema permettono di eseguire direttamente system call (e.g., Bliss e PL 360 (anni 70), C).

- In questo il Run Time System (RTS) (insieme di funzioni predefinite associate ad una particolare piattaforma fornite con il compilatore) fornisce l'interfaccia per la chiamata a basso livello

- Tre metodi generali per passare parametri tra il programma e il sistema operativo:

– Passare i parametri nei *registri*.

– Memorizzare i parametri in una tabella in memoria, il cui indirizzo è passato come parametro in un registro

– Il programma mette i parametri sullo *stack*, da cui il sistema operativo li recupera.

Tipi di chiamate di sistema

Controllo dei processi: creazione/terminazione processi, esecuzione programmi, (de)allocazione memoria, attesa di eventi, impostazione degli attributi,. . .

Gestione dei file: creazione/cancellazione, apertura/chiusura, lettura/scrittura, impostazione degli attributi,. . .

Gestione dei dispositivi: allocazione/rilascio dispositivi, lettura/scrittura, collegamento logico dei dispositivi (e.g. mounting). . .

Informazioni di sistema: leggere/scrivere data e ora del sistema, informazioni sull'hardware/software installato,. . .

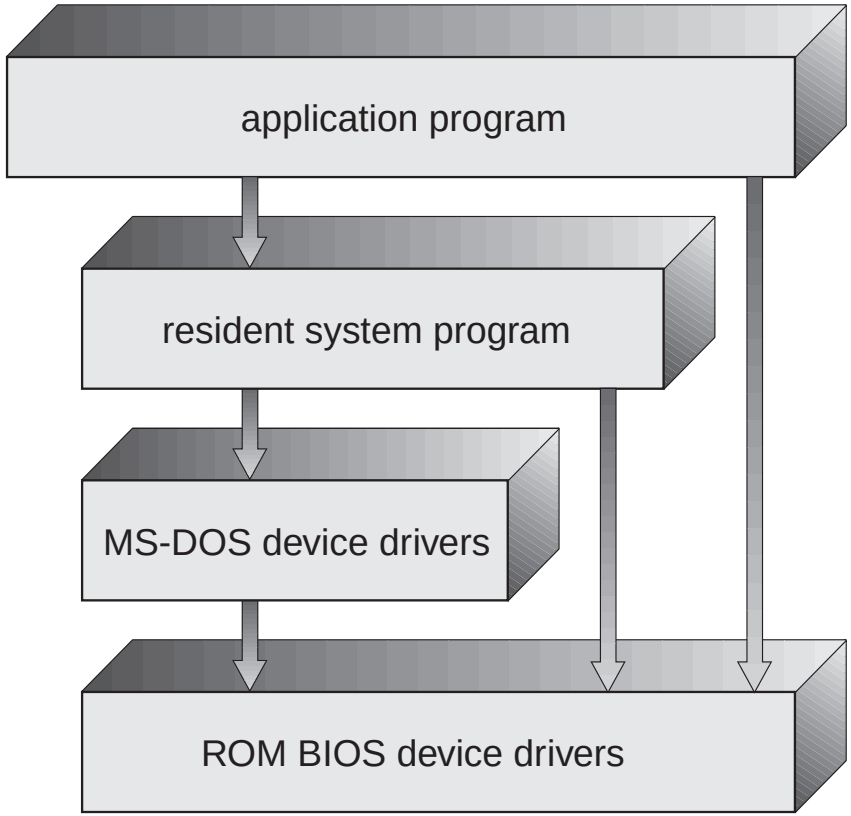
Comunicazioni: creare/cancellare connessioni, spedire/ricevere messaggi,. . .

Programmi di sistema

- I programmi di sistema forniscono un ambiente per lo sviluppo e l'esecuzione dei programmi. Si dividono in
  - Gestione dei file
  - Modifiche dei file
  - Informazioni sullo stato del sistema e dell'utente
  - Supporto dei linguaggi di programmazione
  - Caricamento ed esecuzione dei programmi
  - Comunicazioni
  - Programmi applicativi
- La maggior parte di ciò che un utente vede di un sistema operativo è definito dai programmi di sistema, non dalle reali chiamate di sistema.

Struttura dei Sistemi Operativi -Approccio semplice

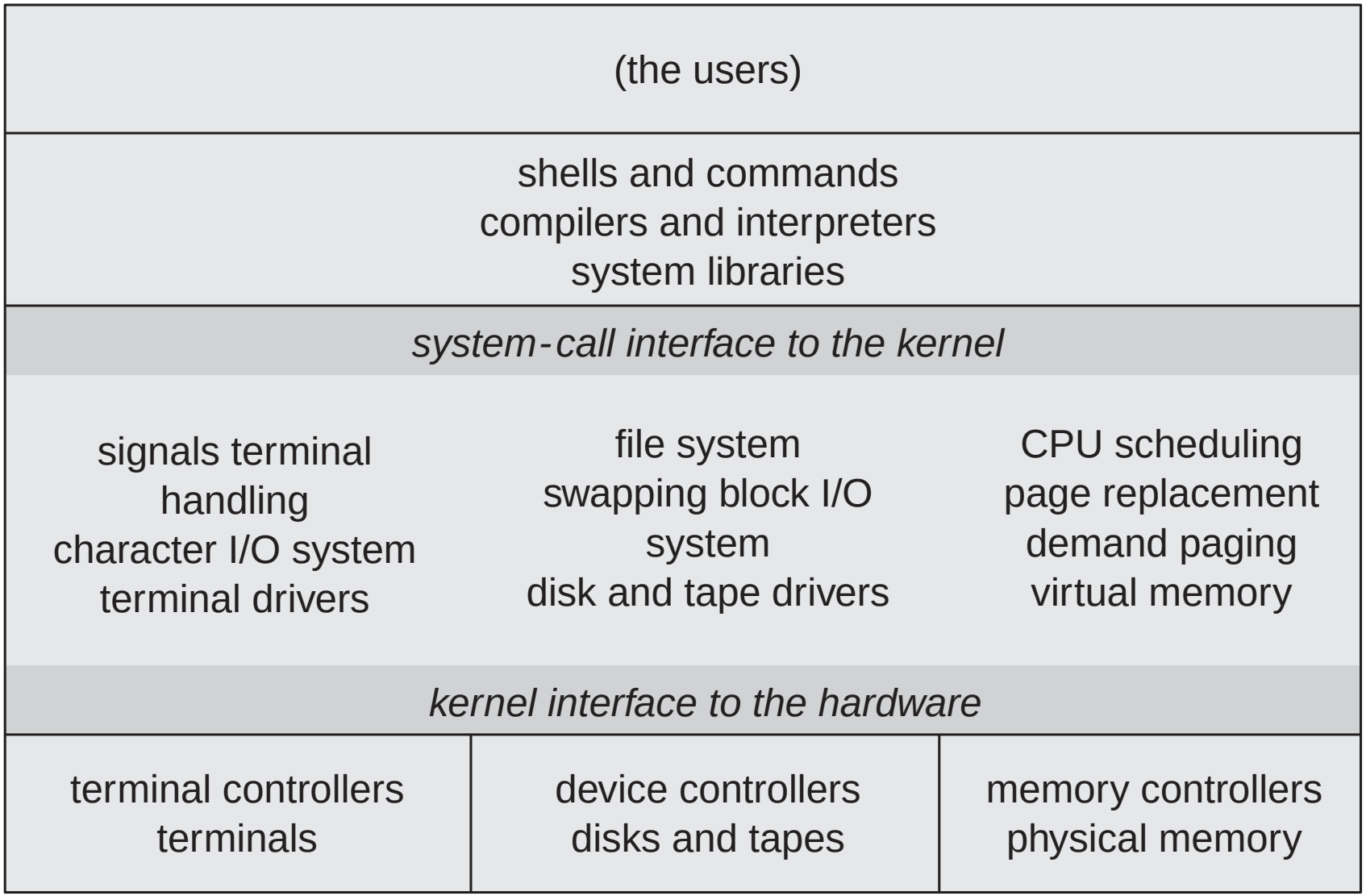
- MS-DOS – pensato per fornire le massime funzionalità nel minore spazio possibile.
  - non è diviso in moduli (è cresciuto oltre il previsto)
  - nonostante ci sia un po' di struttura, le sue interfacce e livelli funzionali non sono ben separati.



Struttura dei Sistemi Operativi - Approccio semplice

- UNIX – limitato dalle funzionalità hardware, lo UNIX originale aveva una debole strutturazione. Consiste almeno in due parti ben separate:
  - Programmi di sistema
  - Il kernel
    * consiste in tutto ciò che sta tra le system call e l'hardware
    * implementa il file system, lo scheduling della CPU, gestione della memoria e altre funzioni del sistema operativo: molte funzionalità in un solo livello.

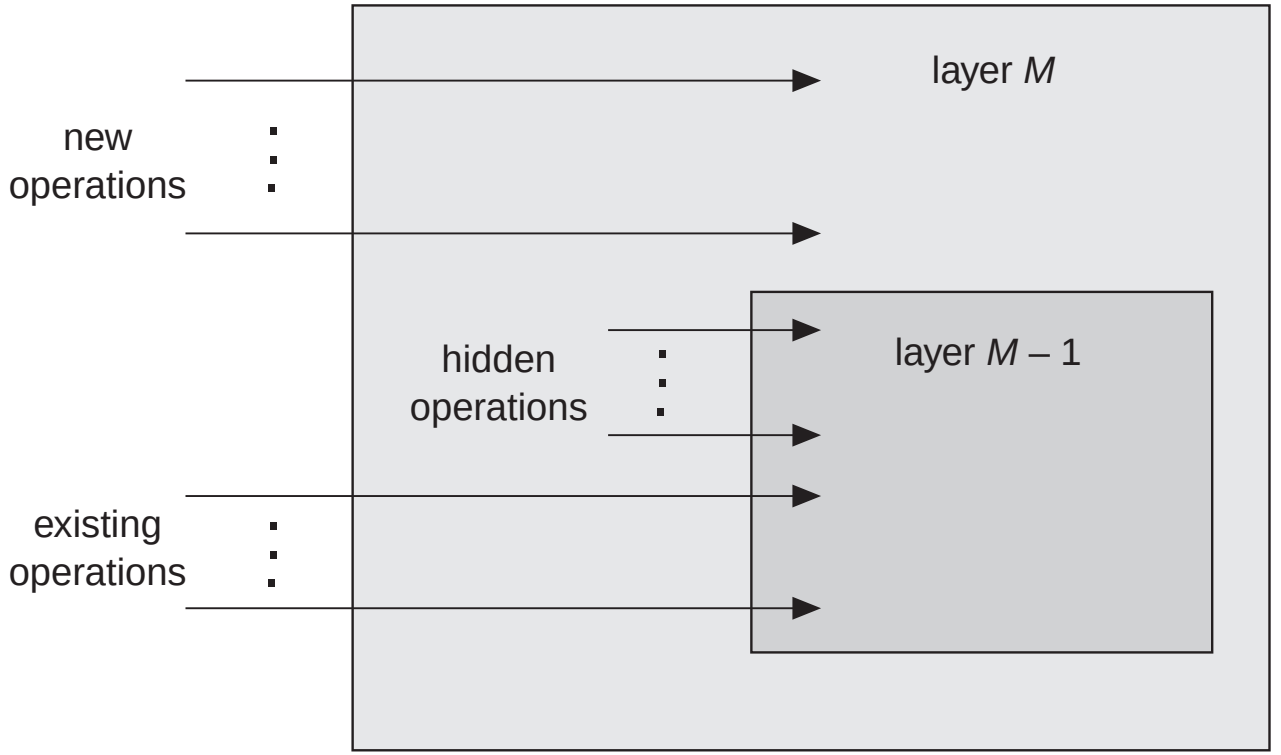
Struttura dei Sistemi Operativi – Unix originale



Struttura dei sistemi operativi – Approccio stratificato

- Il sistema operativo è diviso in un certo numero di strati (livelli); ogni strato è costruito su quelli inferiori. Lo strato di base (livello 0) è l'hardware; il più alto è l'interfaccia utente.

- Secondo la modularità, gli strati sono pensati in modo tale che ognuno utilizza funzionalità (operazioni) e servizi solamente di strati inferiori.



Struttura dei sistemi operativi – Stratificazione di THE

- La prima stratificazione fu usata nel sistema operativo THE per un calcolatore olandese nel 1969 da Dijkstra e dai suoi studenti.

- THE consisteva dei seguenti sei strati:

layer 5:	user programs
layer 4:	buffering for input and output devices
layer 3:	operator-console device driver
layer 2:	memory management
layer 1:	CPU scheduling
layer 0:	hardware

Stratificazione

- Il sistema MULTICS era organizzato ad annelli concentrici (livelli)
- Per accedere ad un livello piú interno occorreva una chiamata di sistema che attivava una TRAP
- L'organizzazione ad anelli si poteva estendere anche a sottosistemi utente (studente lavora a livello $n+1$, programma di correzioni lavora a livello n per evitare interferenze)

Macchine Virtuali

- Una *macchina virtuale* porta l'approccio stratificato all'estremo: tratta hardware e il sistema operativo come se fosse tutto hardware.

- Una macchina virtuale fornisce una interfaccia *identica* all'hardware nudo e crudo sottostante.

- Il sistema operativo impiega le risorse del calcolatore fisico per creare le macchine virtuali:

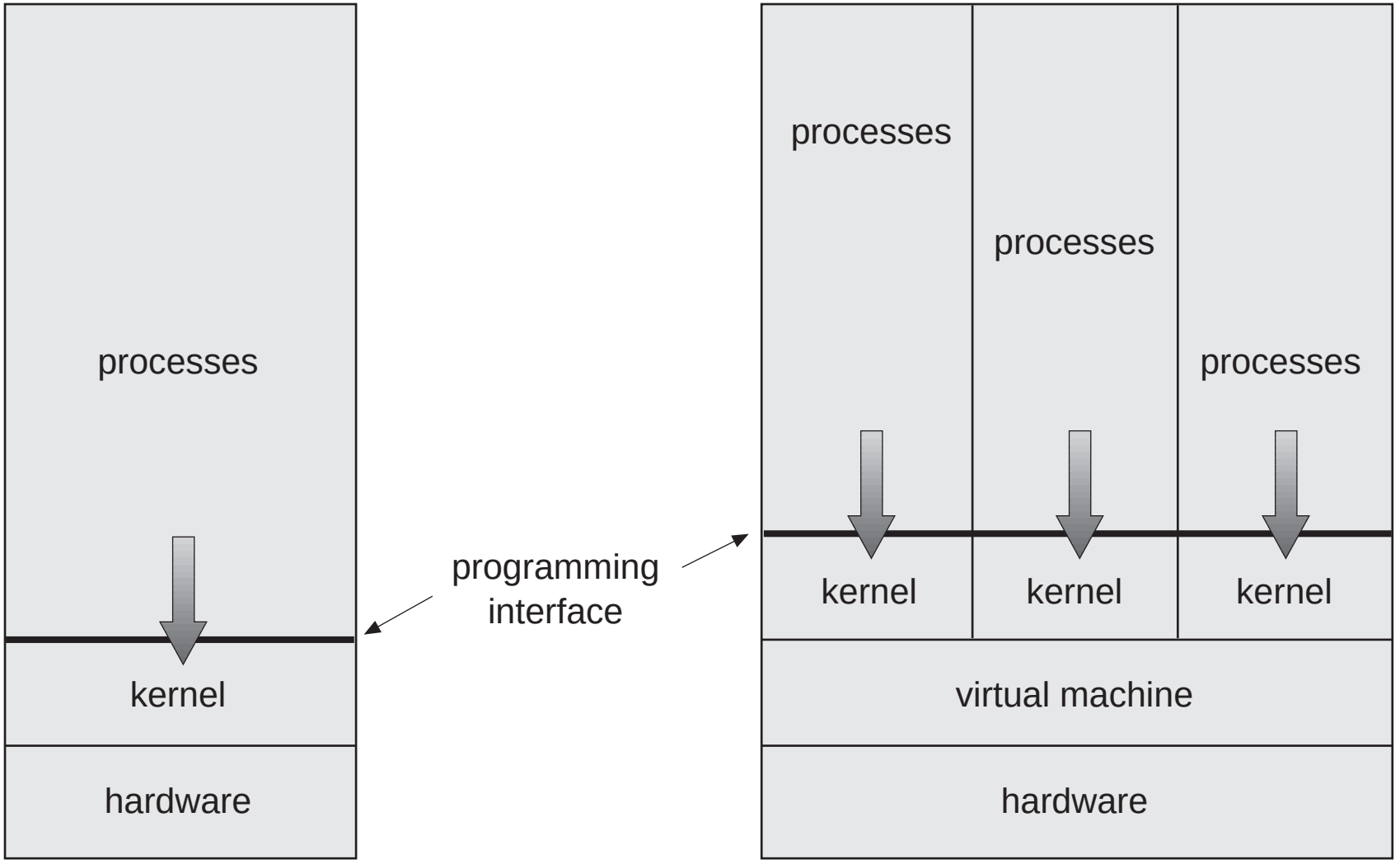
  - Lo scheduling della CPU crea l'illusione che ogni processo abbia il suo processore dedicato.

  - La gestione della memoria crea l'illusione di una memoria virtuale per ogni processo

  - Lo spooling può implementare delle stampanti virtuali

  - Spazio disco può essere impiegato per creare "dischi virtuali"

Macchine Virtuali (Cont.)



(a) Macchina non virtuale; (b) Macchine virtuali

Vantaggi/Svantaggi delle Macchine Virtuali

- Il concetto di macchina virtuale fornisce una protezione completa delle risorse di sistema, dal momento che ogni macchina virtuale è isolata dalle altre. Questo isolamento non permette però una condivisione diretta delle risorse.

- Un sistema a macchine virtuali è un mezzo perfetto per l'emulazione di altri sistemi operativi, o lo sviluppo di nuovi sistemi operativi: tutto si svolge sulla macchina virtuale, invece che su quella fisica, quindi non c'è pericolo di far danni.

- Implementare una macchina virtuale è complesso, in quanto si deve fornire un *perfetto* duplicato della macchina sottostante. Può essere necessario dover emulare ogni singola istruzione macchina.

- Approccio seguito in molti sistemi: Windows, Linux, MacOS, JVM,. . .

Exokernel

- Estensione dell'idea di macchina virtuale
- Ogni macchina virtuale di livello utente vede solo un *sottoinsieme* delle risorse dell'intera macchina
- Ogni macchina virtuale può eseguire il proprio sistema operativo
- Le risorse vengono richieste all'exokernel, che tiene traccia di quali risorse sono usate da chi
- Semplifica l'uso delle risorse allocate: l'exokernel deve solo tenere separati i domini di allocazione delle risorse

Meccanismi e Politiche

- I kernel tradizionali (monolitici) sono poco flessibili

- Distinguere tra *meccanismi* e *politiche*:

  – i meccanismi determinano *come* fare qualcosa;

  – le politiche determinano *cosa* deve essere fatto.

  Ad esempio: assegnare l'esecuzione ad un processo è un meccanismo; scegliere *quale* processo attivare è una politica.

- Questa separazione è un principio molto importante: permette la massima flessibilità, nel caso in cui le politiche debbano essere cambiate.

- Estremizzazione: il kernel fornisce solo i meccanismi, mentre le politiche vengono implementate in user space.

Sistemi con Microkernel

- *Microkernel*: il kernel è ridotto all'osso, fornisce soltanto i meccanismi:
  - Un meccanismo di comunicazione tra processi
  - Una minima gestione della memoria e dei processi
  - Gestione dell'hardware di basso livello (driver)
- Tutto il resto viene gestito da processi in spazio utente: ad esempio, tutte le politiche di gestione del file system, dello scheduling, della memoria sono implementate come processi.
- Meno efficiente del kernel monolitico
- Grande flessibilità; immediata scalabilità in ambiente di rete
- Sistemi operativi recenti sono basati, in diverse misure, su microkernel (AIX4, BeOS, GNU HURD, MacOS X, QNX, Tru64, Windows NT . . .)