1 - Introduzione ai Sistemi Operativi

Sommario

Cos'è un Sistema Operativo?
- macchina astratta
- gestore di risorse

Storia dei S.O.
- generazioni 1-5 dei S.O.
- Storia di Internet e World Wide Web

Componenti dei S.O.
architetture Hardware

Tipi di S.O. e scopi dei S.O.

Concetti base dei S.O.

Strutture di S.O.
- Monolitica
- a Livelli
- Microkernel
- S.O. di rete e S. O. Distribuiti

Sistemi Operativi – concetti base

- Le architetture di sistemi includono
  - Caratteristiche che svolgono funzioni del sistema operativo rapidamente in hardware per migliorare le **prestazioni**
  - Caratteristiche che consentono al sistema operativo per far rispettare rigidamente la **protezione**

Sistemi Operativi – concetti base – protezione

- Un processore implementa i **meccanismi di protezione** di un sistema operativo
  - Impedisce ai processi di accedere a istruzioni privilegiate o a zone di memoria
  - I sistemi in genere hanno diverse modalità di esecuzione
    - **Modalità utente** (stato utente)
      - L'utente può eseguire solo un sottoinsieme di istruzioni
    - **Kernel mode** (stato supervisore)
      - Processore può accedere alle istruzioni e alle risorse privilegiate per conto dei processi
  - Principio del **privilegio minimo** – ad ogni utente minimi privilegi e accessi

Sistemi Operativi – concetti base – protezione

- Protezione e gestione della **memoria**
  - Previene che i processi accedano alla memoria che non è stata loro assegnati
  - Implementato utilizzando registri del processore modificati solo da istruzioni privilegiate

- Interrupts e Eccezioni
  - La maggior parte dei dispositivi inviano un segnale chiamato un interrupt al processore quando accade un evento
  - Le eccezioni sono interrupt generati in risposta agli errori
  - Il S.O. può rispondere ad un interrupt notificandolo ai processi che sono in attesa su questi eventi

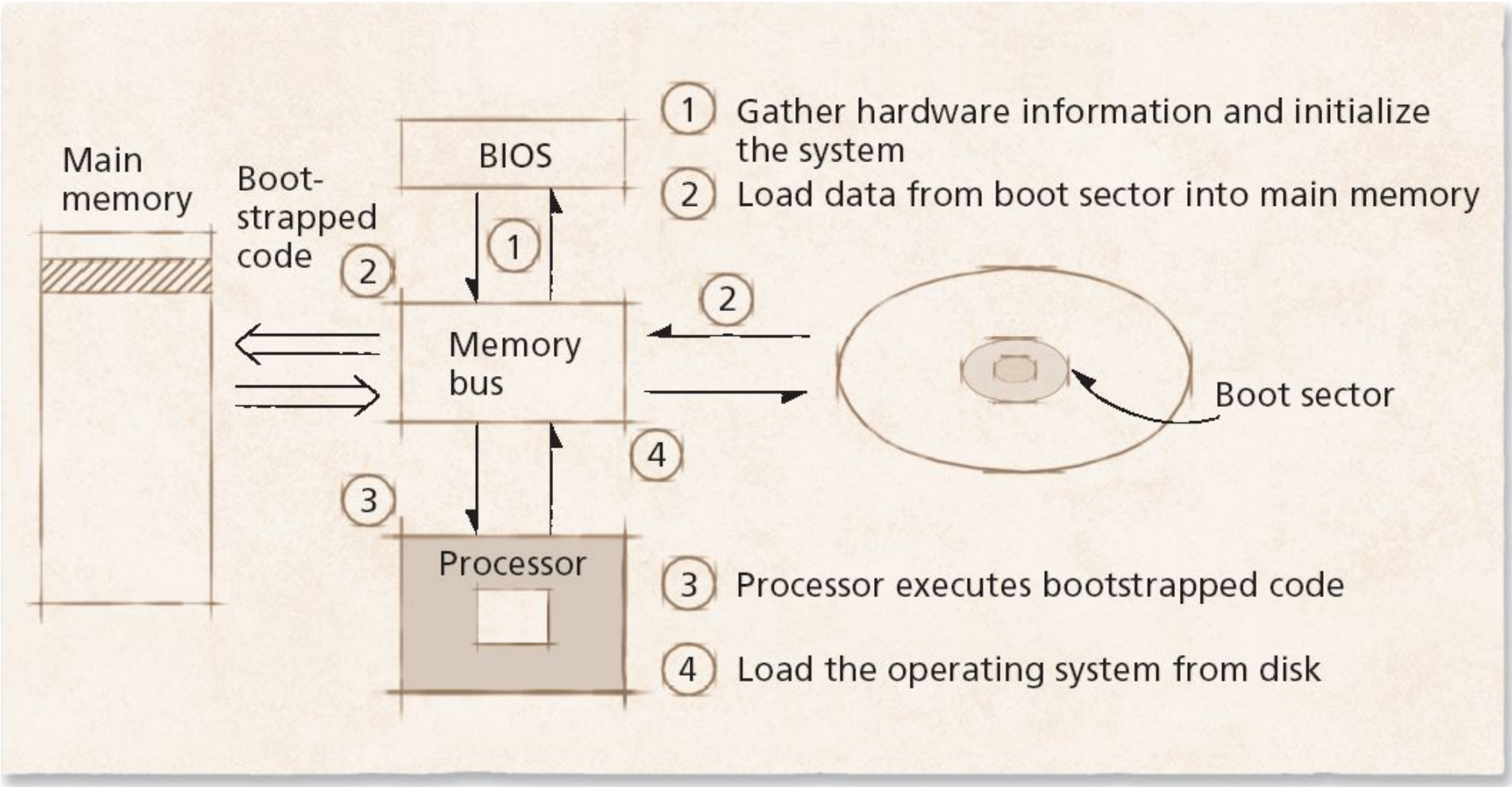
Sistemi Operativi – concetti base – interruzioni

- Timer
  - Un timer a intervalli genera periodicamente un interrupt
  - I sistemi operativi utilizzano timer a intervalli per evitare che i processi monopolizzino il processore
- Clocks
  - Forniscono una misura di continuità
  - Un orologio *ora-del-giorno* permette al S.O. di determinare il tempo e la data corrente

Sistemi Operativi – concetti base – avvio

- *Bootstrapping*:
  caricamento in memoria di componenti del sistema operativo iniziali
  - Eseguita dal *Basic Input/Output System* (BIOS)
    - Inizializza l'hardware di sistema
    - Carica le istruzioni in memoria principale da una regione di memoria secondaria chiamata il settore di avvio (*boot*)
  - Se il sistema non è caricato, l'utente non può accedere ad alcuna componente hardware del computer
  - Evoluzione: EFI (*Extensible Firmware Interface*) con interfaccia testuale (*shell*) e *driver.* L'utente può accedere ai dispositivi, dischi rigidi e rete.

Sistemi Operativi – concetti base – avvio

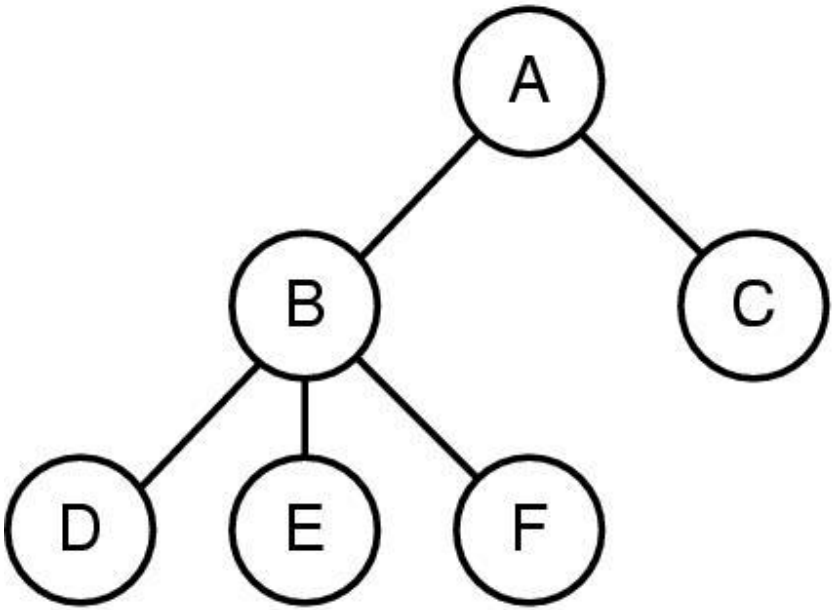


© Deitel & Ass. Inc.

Sistemi Operativi – concetti base – processi

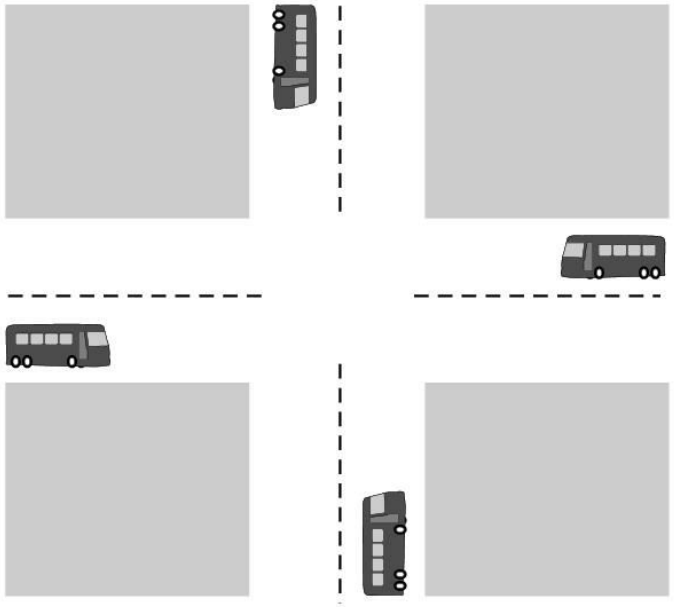
- **Processo**: programma in esecuzione
  - Spazio degli indirizzi
  - Insieme di risorse
    registri, file, segnali,…
  - Descrittore di processo
  - UID Identificatore unico di utente
  - Ogni processo ha un UID
  - Tabella di processo
  - Possibilità di creare processi ‘figli’
  - Comunicazione e sincronizzazione fra processi
  - IPC (*interprocess communication*)

Sistemi Operativi – concetti base – processi

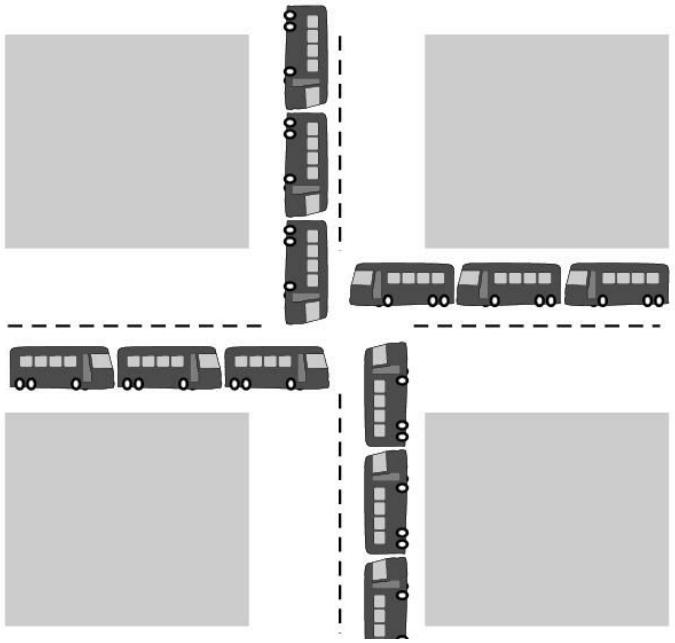


- Un albero di processi
  - A crea due processi figli, B e C
  - B crea tre processi figli, D, E, e F

Sistemi Operativi – concetti base – processi - stallo



Stallo (*deadlock)* potenziale

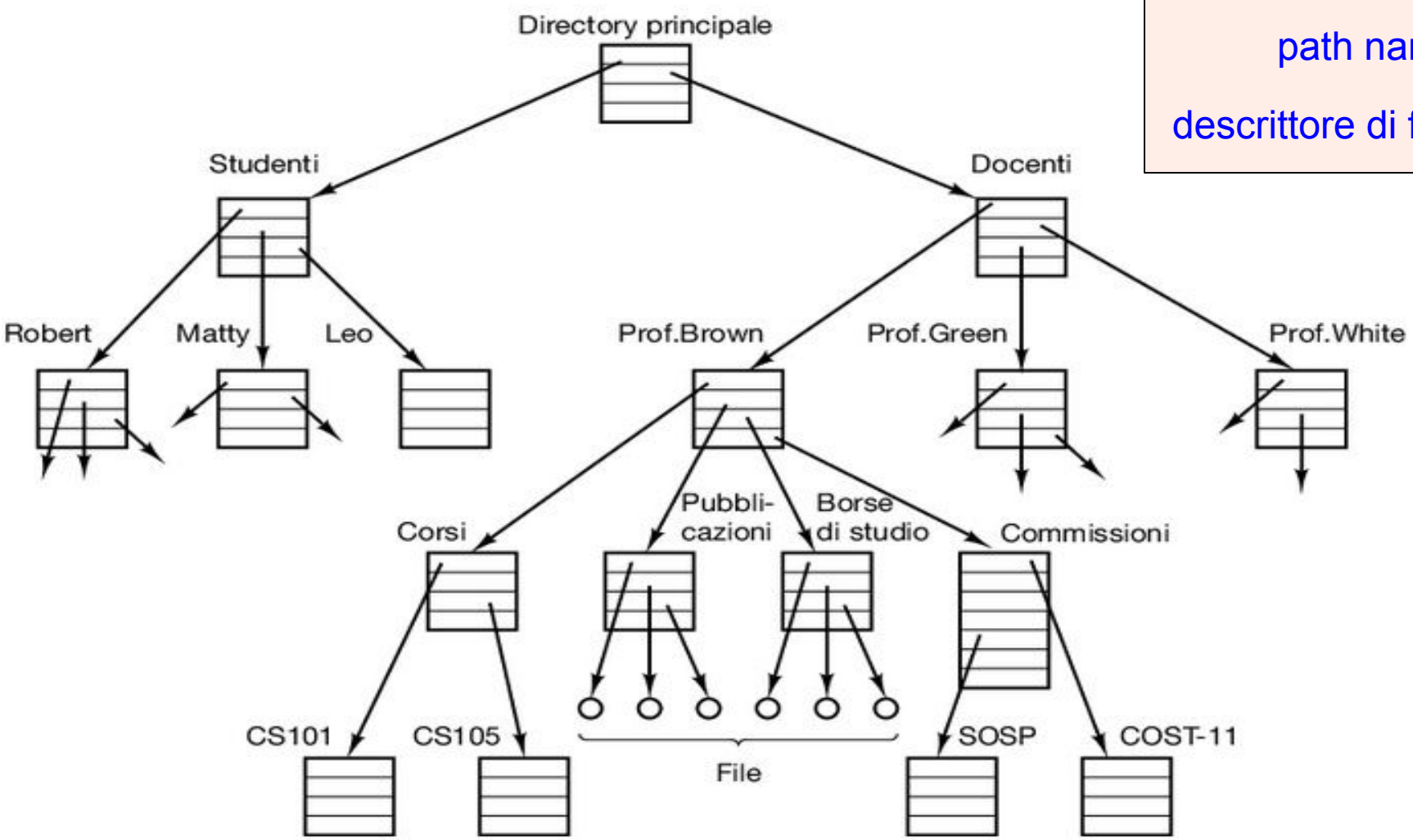


Stallo effettivo

Sistemi Operativi – concetti base – file system

path name

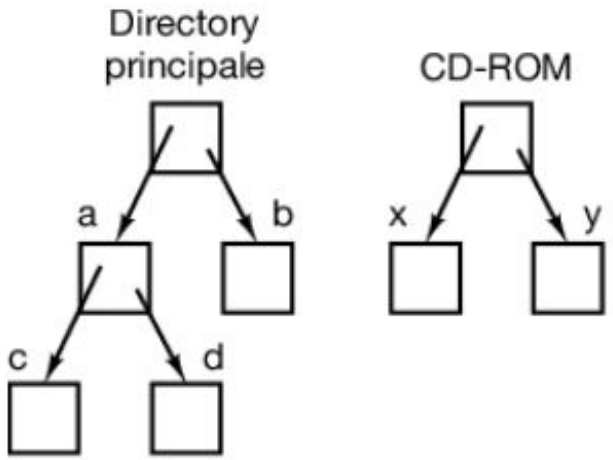
descrittore di file



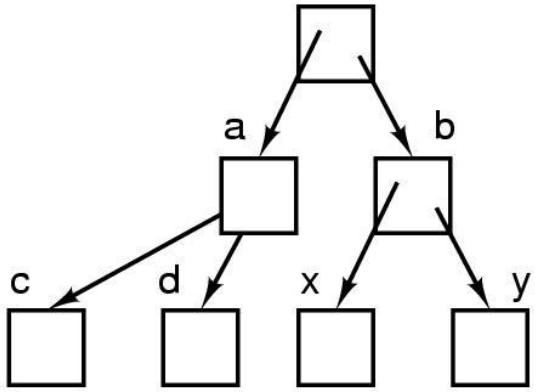
Esempio di struttura di un file system

Sistemi Operativi – concetti base – file system

Montare un file system



(a)



(b)

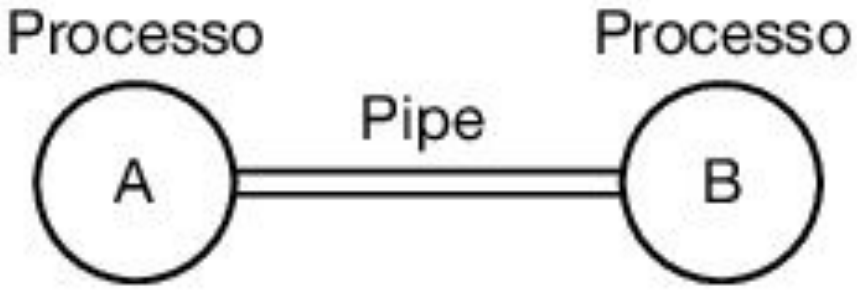
- Prima di montare
  - i file sul CD non sono accessibili
- Dopo aver montato il CD sul nodo b,
  - I files sul CD son parte della gerarchia del file system

Sistemi Operativi – concetti base – file system

In Unix

- File speciali per vedere e trattare dispositivi di I/O come file
  - a blocchi es. per dischi
  - a caratteri es. per stampanti, modem
  - nella directory /dev

- Pipe pseudofile per connessione fra due processi



Due processi connessi da *pipe*

Sistemi Operativi – concetti base – plug and play

Tecnologia *Plug and Play*

Consente ai sistemi operativi di configurare l'hardware di nuova installazione senza l'interazione dell'utente

Per supportare plug and play, un dispositivo hardware deve:

- Identificarsi unicamente al sistema operativo
- Comunicare con il sistema operativo per indicare le risorse e i servizi che il dispositivo richiede per funzionare correttamente
- Identificare il driver che supporta il dispositivo e consente al software di configurare il dispositivo (ad esempio, assegnare il dispositivo a un canale DMA)

Sistemi Operativi – concetti base – cache, buffer, spool

Caches

Memorie relativamente veloci
Mantiene copie di dati che saranno presto richiesti
Aumenta la velocità di esecuzione di un programma
Cache miss/hit: dati non presenti/presenti in cache
Algoritmi per ottimizzare l'uso della cache. Spesso euristiche.

Buffers

area di memorizzazione temporanea che contiene i dati durante I / O
Usato per:
- Coordinamento delle comunicazioni tra i dispositivi a diverse velocità
- Memorizzare dei dati per l'elaborazione asincrona
- Permette ai segnali di essere consegnati in modo asincrono

Spooling (*Simultaneous Peripheral Operations On Line*)

Tecnica di buffering in cui un dispositivo intermedio (es. disco) è interposto tra un processo e una periferica I/O lenta

Permette ai processi di inviare operazioni di richiesta da una periferica senza aspettare che il dispositivo sia pronto a servire la richiesta

Sistemi Operativi – concetti base – memoria Virtuale

- Memoria Virtuale
  - Possibilità di eseguire programmi con richieste di memoria maggiori rispetto alla memoria fisica
  - Primo sistema operativo con memoria virtuale: MULTICS,
  - Oggi in Unix e Windows

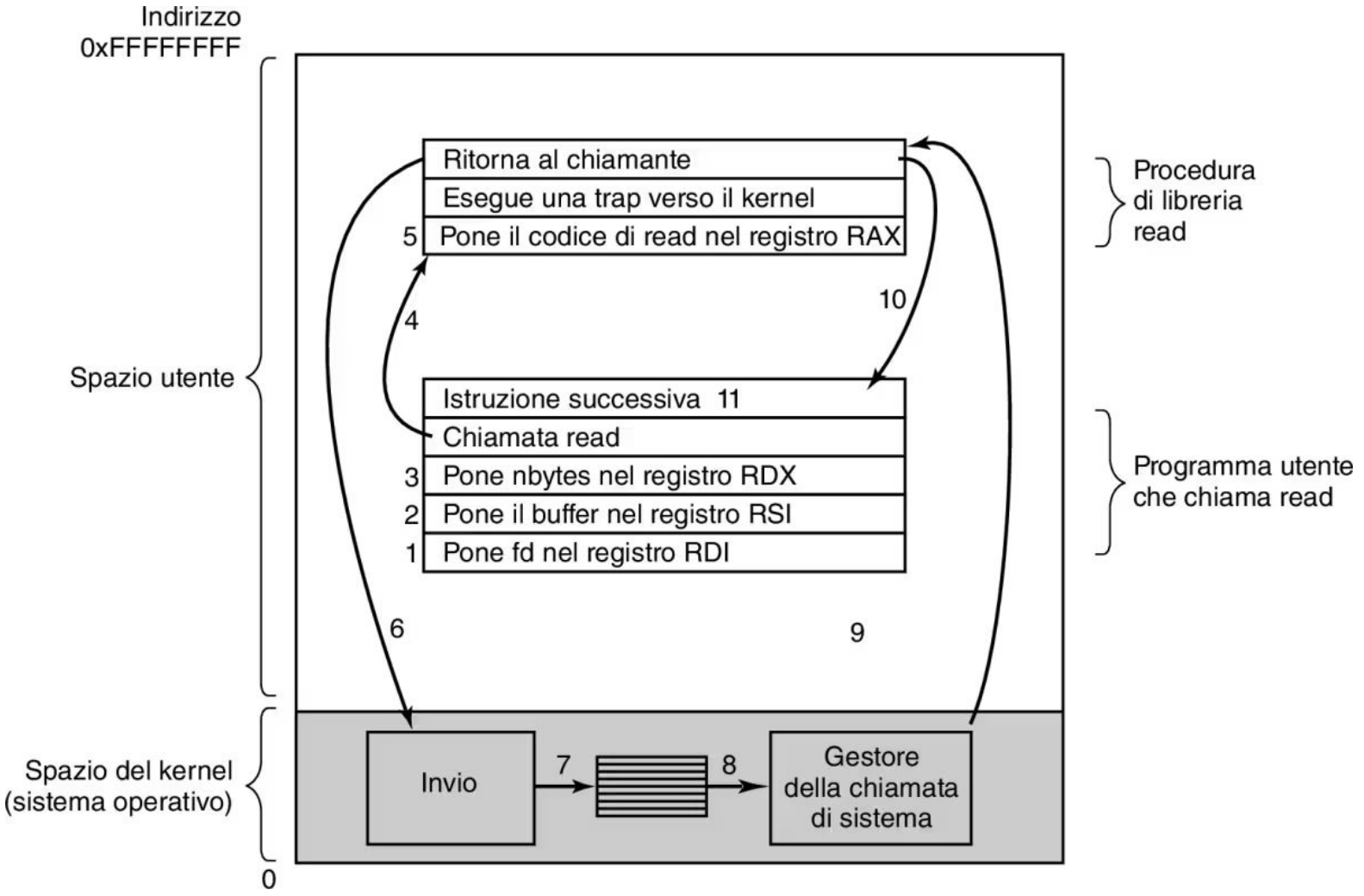
Sistemi Operativi – concetti base – chiamate di sistema

- Chiamate di sistema
  - Nell'interfaccia del S.O.
  - Un processo utente attiva attraverso una *TRAP*
  - Il controllo passa al sistema operativo
  - al termine il controllo ritorna all'istruzione successiva del processo utente

Esempio: read in Unix

read (fd, buffer, nbytes)

Sistemi Operativi – concetti base – chiamate di sistema



read (fd, buffer, nbytes)

Sistemi Operativi – concetti base – chiamate di sistema

Chiamate di sistema per la gestione di processi in POSIX

Call	Description
pid = fork( )	Create a child process identical to the parent
pid = waitpid(pid, &statloc, options)	Wait for a child to terminate
s = execve(name, argv, environp)	Replace a process' core image
exit(status)	Terminate process execution and return status

Chiamate di sistema per la gestione di file

Call	Description
fd = open(file, how, ...)	Open a file for reading, writing or both
s = close(fd)	Close an open file
n = read(fd, buffer, nbytes)	Read data from a file into a buffer
n = write(fd, buffer, nbytes)	Write data from a buffer into a file
position = lseek(fd, offset, whence)	Move the file pointer
s = stat(name, &buf)	Get a file's status information

Sistemi Operativi – concetti base – chiamate di sistema

Chiamate di sistema per la gestione del file system e delle directories

Call	Description
s = mkdir(name, mode)	Create a new directory
s = rmdir(name)	Remove an empty directory
s = link(name1, name2)	Create a new entry, name2, pointing to name1
s = unlink(name)	Remove a directory entry
s = mount(special, name, flag)	Mount a file system
s = umount(special)	Unmount a file system

Altre chiamate di sistema

Call	Description
s = chdir(dirname)	Change the working directory
s = chmod(name, mode)	Change a file's protection bits
s = kill(pid, signal)	Send a signal to a process
seconds = time(&seconds)	Get the elapsed time since Jan. 1, 1970

Sistemi Operativi – concetti base – chiamate di sistema

Alcune WIn32 API (Application Program Interface) per ottenere servizi del sistema

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

Sistemi Operativi - Componenti e Obbiettivi

- Sviluppo dei sistemi di elaborazione
  - I sistemi di elaborazione sono evoluti
  - I primi sistemi non contenevano alcun sistema operativo
  - Successivamente sono stati sviluppati i sistemi con multiprogrammazione e timesharing
  - Quindi i personal computer e poi i veri e propri sistemi distribuiti
  - Nuove funzioni dipendenti
    - dalla crescita della domanda
    - dalla diversificazione della domanda
    - dai requisiti degli utenti

Componenti centrali dei Sistemi Operativi

- Interazione utente-sistema operativo
  - Spesso, attraverso l'applicazione speciale chiamata shell (interprete di comandi del S.O.)
  - Nucleo (Kernel)
    - Software che contiene le componenti fondamentali del SO
  - Modalità supervisore (privilegiata) vs modalità utente

- Tipici componenti di un SO comprendono:
  - Processor scheduler
  - Gestore della memoria
  - Gestore della I/O
  - Gestore della Interprocess communication (IPC)
  - Gestore del File system

Componenti centrali dei Sistemi Operativi

- Gli ambienti multiprogrammati sono ora molto diffusi
  - Il nucleo gestisce l'esecuzione dei processi
  - **Thread**: componenti dei programmi eseguite in modo indipendente, ma che utilizzano un unico spazio condiviso di memoria per i dati
  - Per accedere ad un dispositivo I/O, un processo effettua una chiamata di sistema
    - Gestita dal driver della periferica
    - Componente software che interagisce direttamente con l'hardware
    - Spesso contiene comandi specifici del dispositivo

Sistemi Operativi - Obbiettivi

- Proprietà dei S.O. desiderabili dagli utenti
  - Efficienza
  - Robustezza
  - Scalabilità
  - Estensibilità
  - Portabilità
  - Sicurezza
  - Protezione
  - Interattività
  - Usabilità

Sistemi Operativi - Architetture

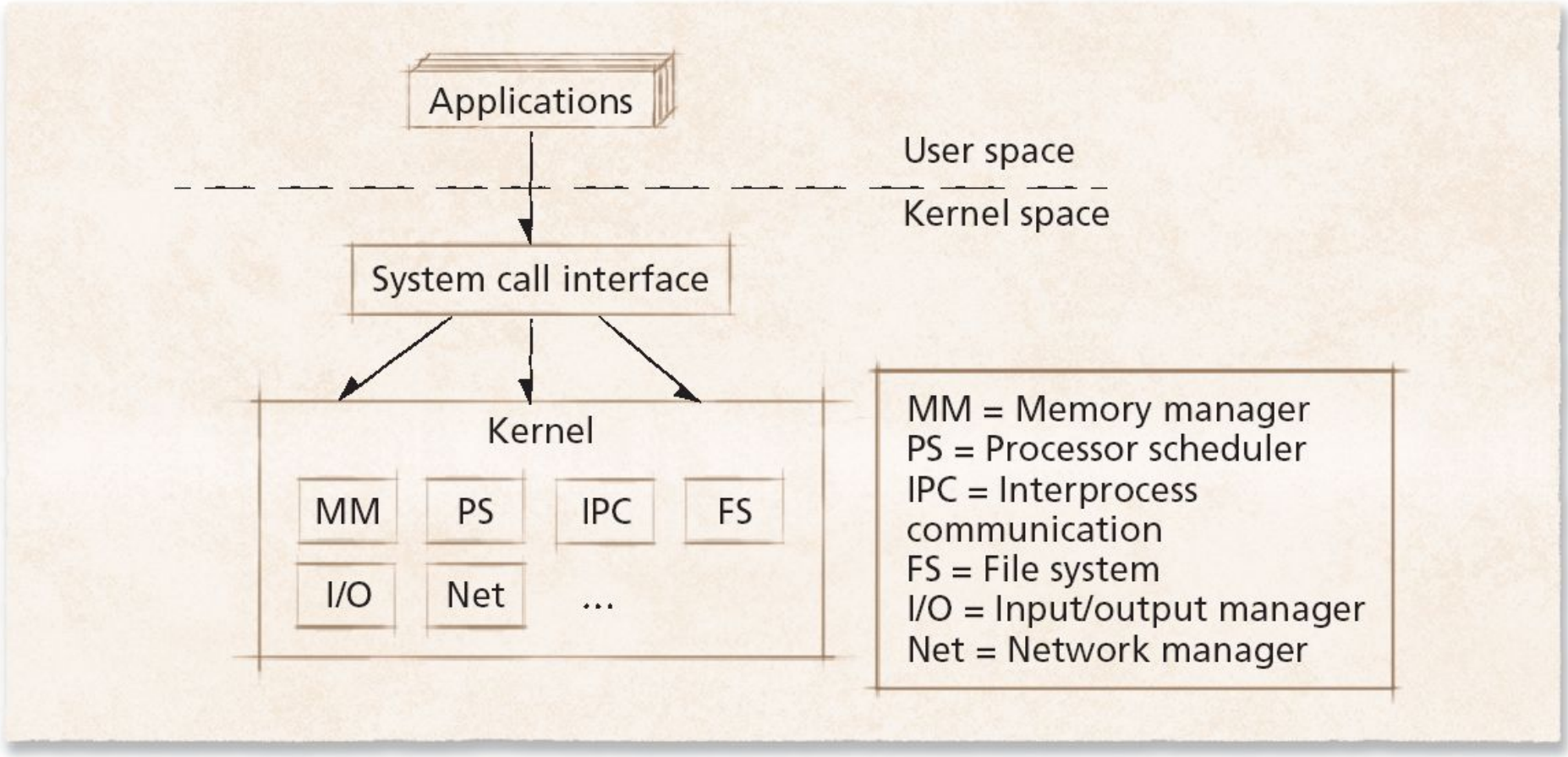
- I SO attuali tendono ad essere complessi
  - Forniscono molti servizi
  - Eterogeneità: supportano hardware e software diversi
  - Le architetture di SO aiutano a gestire tale complessità
    - Organizzazione delle componenti del sistema operativo
    - Specifica della priorità di esecuzione con cui ogni componente può essere eseguita

Sistemi Operativi – Architettura Monolitica

- Sistemi Operativi **monolitici**
  - Ogni componente è contenuta nel nucleo
  - Ogni componente può comunicare direttamente con qualsiasi altra
  - Obbiettivo: elevata efficienza
  - Principale svantaggio: difficoltà a identificare eventuali fonti di errori

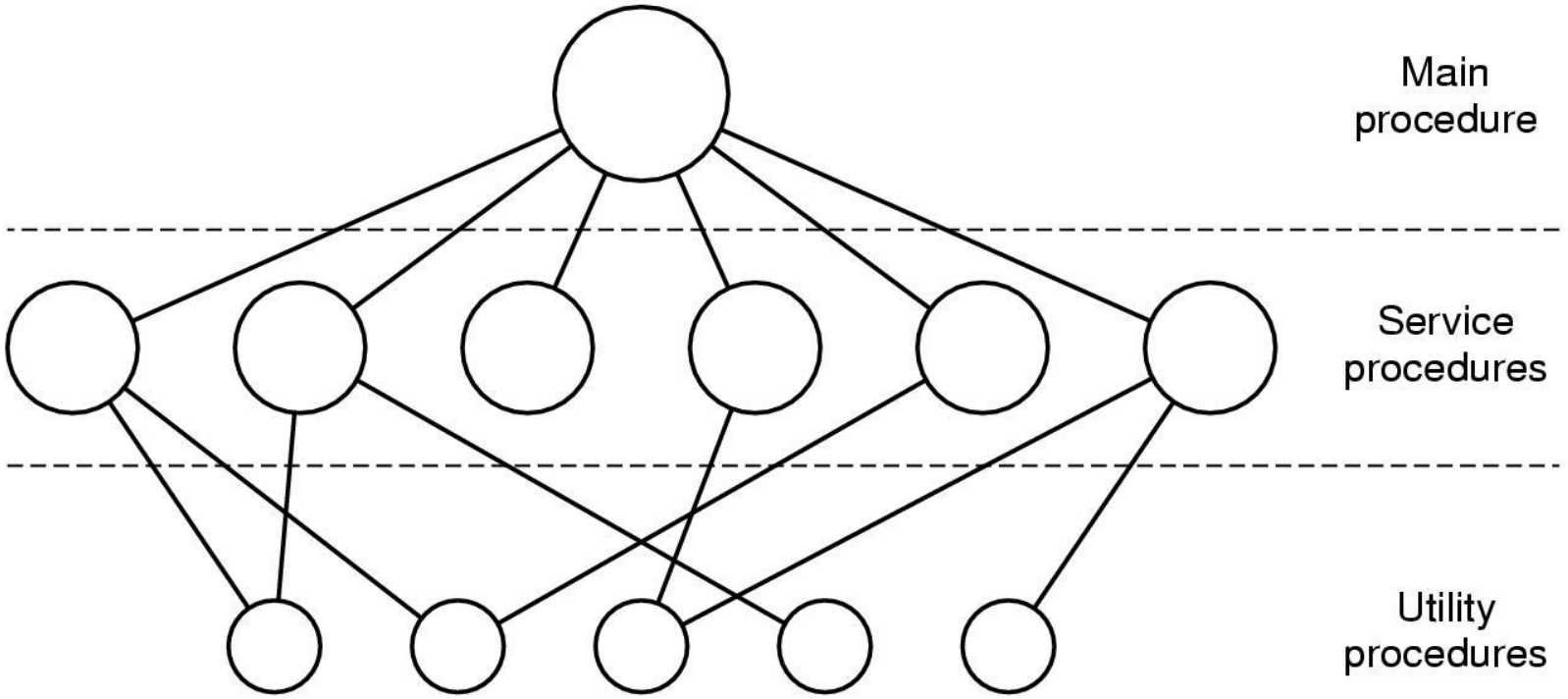
Sistemi Operativi – Architettura Monolitica

Architettura di un nucleo di sistema operativo monolitico



© Deitel & Ass. Inc.

Sistemi Operativi – Architettura Monolitica - esempio



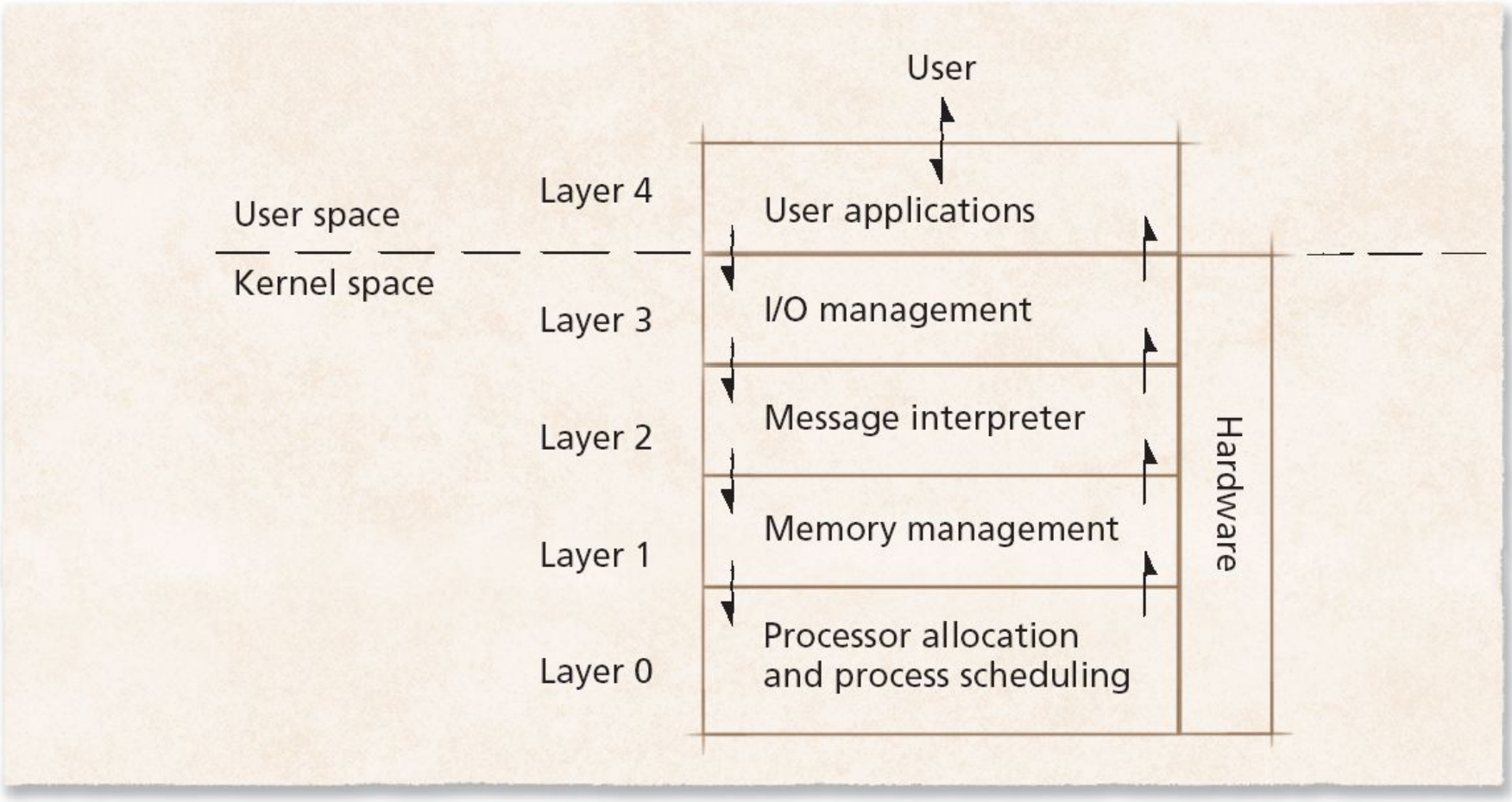
Semplice modello strutturale di un sistema monolitico

Sistemi Operativi – Architettura a livelli

- Sistemi Operativi con approccio **a livelli**
  - Obbiettivo: migliorare il progetto del nucleo monolitico
    - Nei livelli: gruppi di componenti che svolgono funzioni simili
  - **Isolamento**: ogni livello comunica solo con strati immediatamente sopra e sotto
  - Le richieste dei processi eventualmente attraversano diversi strati prima del completamento
  - Il throughput di sistema può essere meno efficiente dei sistemi con nuclei di SO monolitici
    - Ulteriori metodi devono essere invocati per passare i dati e il controllo

Sistemi Operativi – Architettura a livelli

Livelli del sistema operativo THE



© Deitel & Ass. Inc.

Sistemi Operativi – Architettura a livelli - esempio

Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

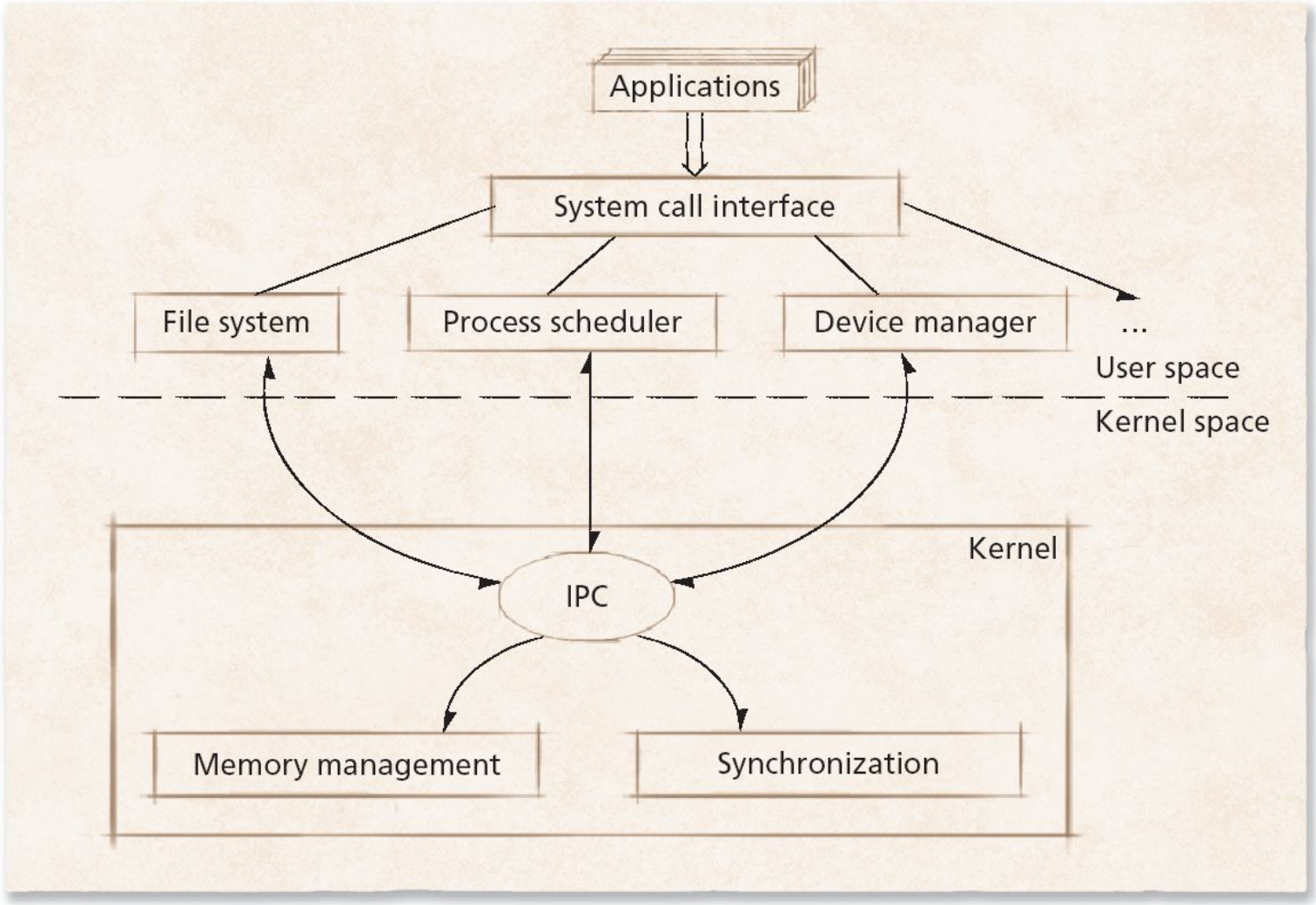
Struttura del sistema operativo THE

Sistemi Operativi – Architettura Microkernel

- Architettura di un SO **Microkernel**
  - Fornisce solo **servizi limitati**
    - Tentativo di contenere le dimensioni del kernel e garantire la scalabilità
  - Elevato grado di modularità
    - Estensibile, portabile e scalabile
  - Aumento del livello di comunicazione fra moduli
    - Può portare ad una degradazione delle prestazioni del sistema

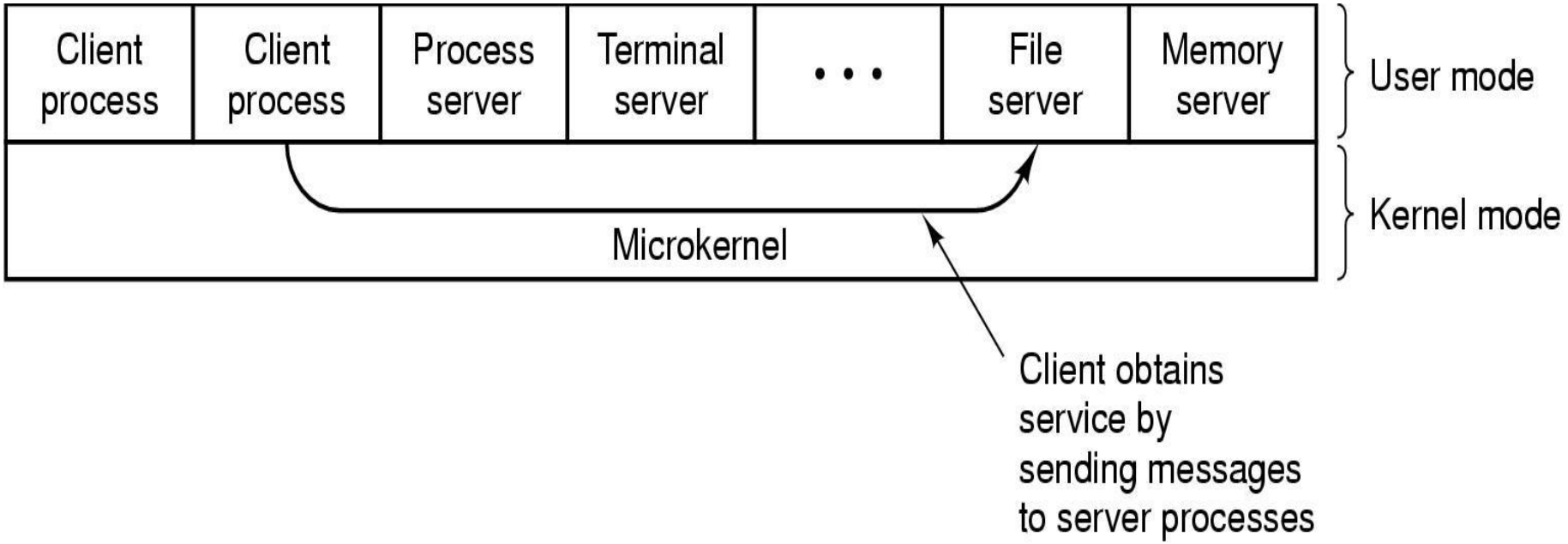
Sistemi Operativi – Architettura Microkernel

Architettura di un sistema operativo Microkernel



© Deitel & Ass. Inc.

Sistemi Operativi – Architettura Microkernel - esempio



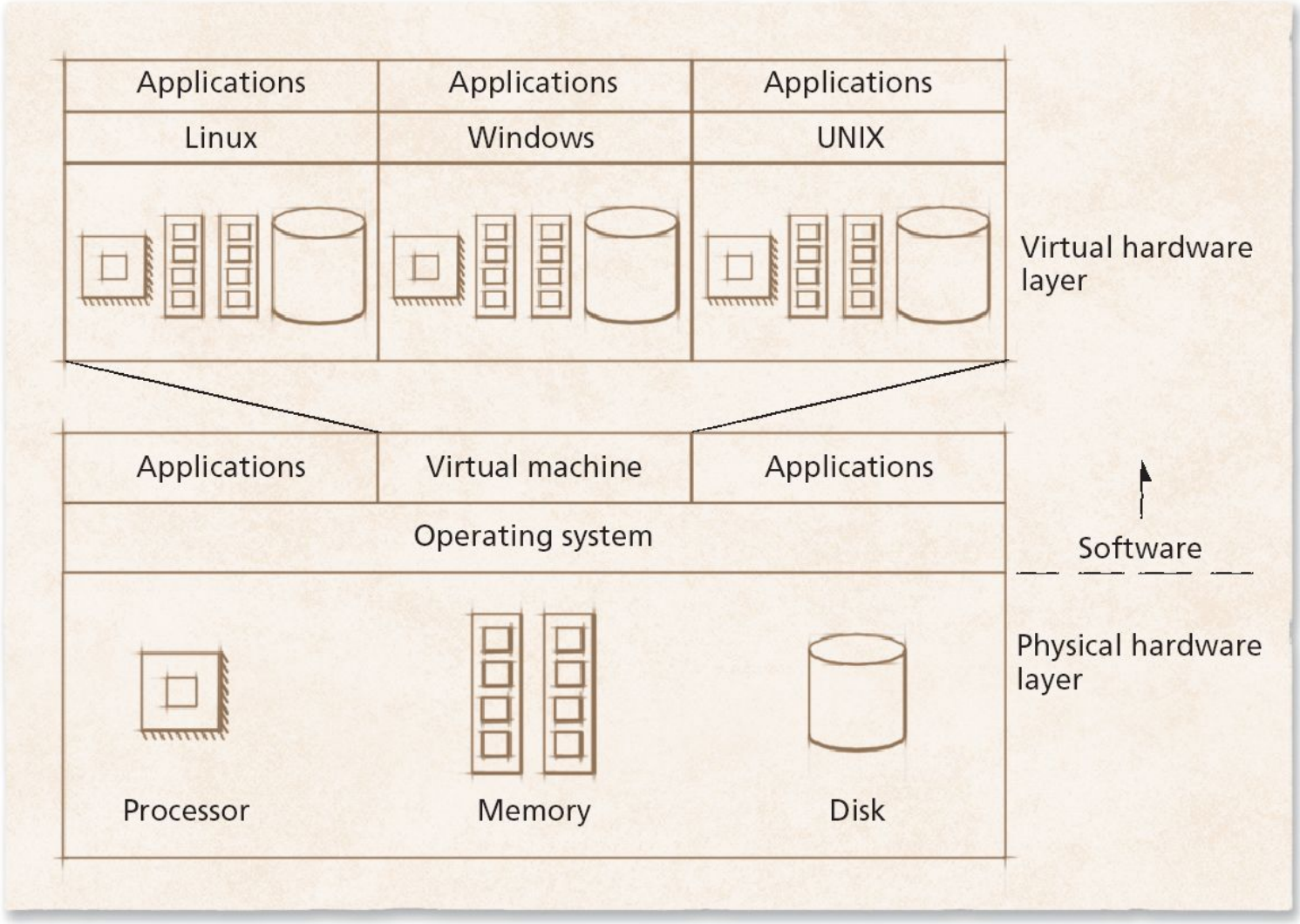
Modello cliente-servente

Sistemi Operativi –Macchina Virtuale

- Macchine Virtuali (VMs)
  - Astrazione software di un sistema di elaborazione
  - Spesso in esecuzione sopra ad un SO nativo
- Sistema Operativo a Macchine Virtuali
  - Gestisce le risorse fornite dalla macchina virtuale
- Applicazioni delle Macchine Virtuali
  - Permette la coesistenza di diverse istanze di un SO eseguibili contemporaneamente
  - Emulazione
    - Software o hardware che imita le funzionalità di hardware o software non presente nel sistema
  - Facilita la portabilità

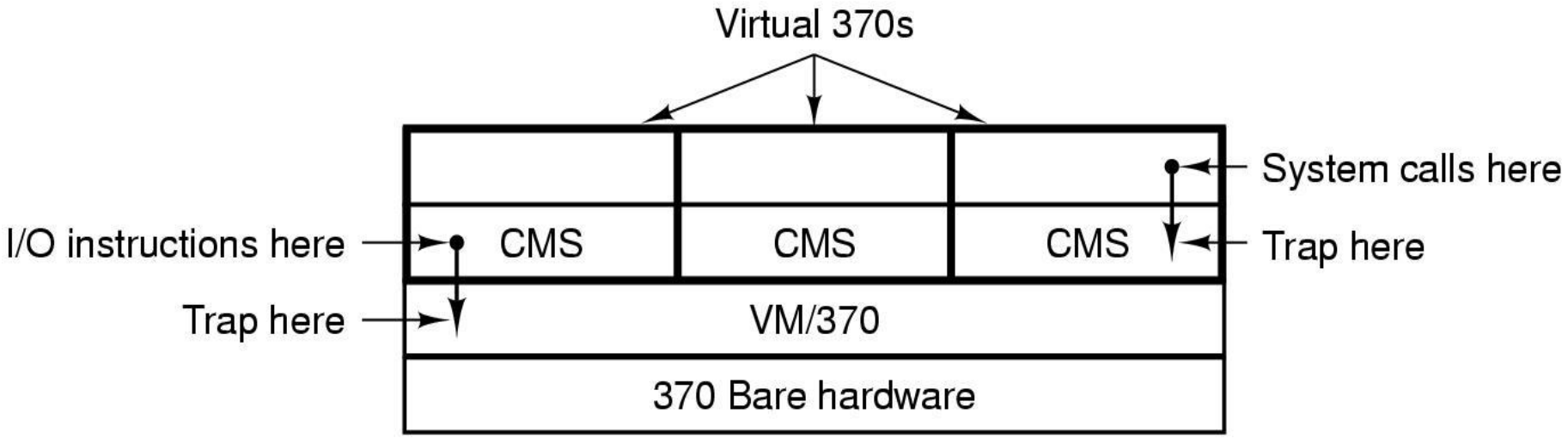
Sistemi Operativi –Macchina Virtuale

Schema di una macchina virtuale



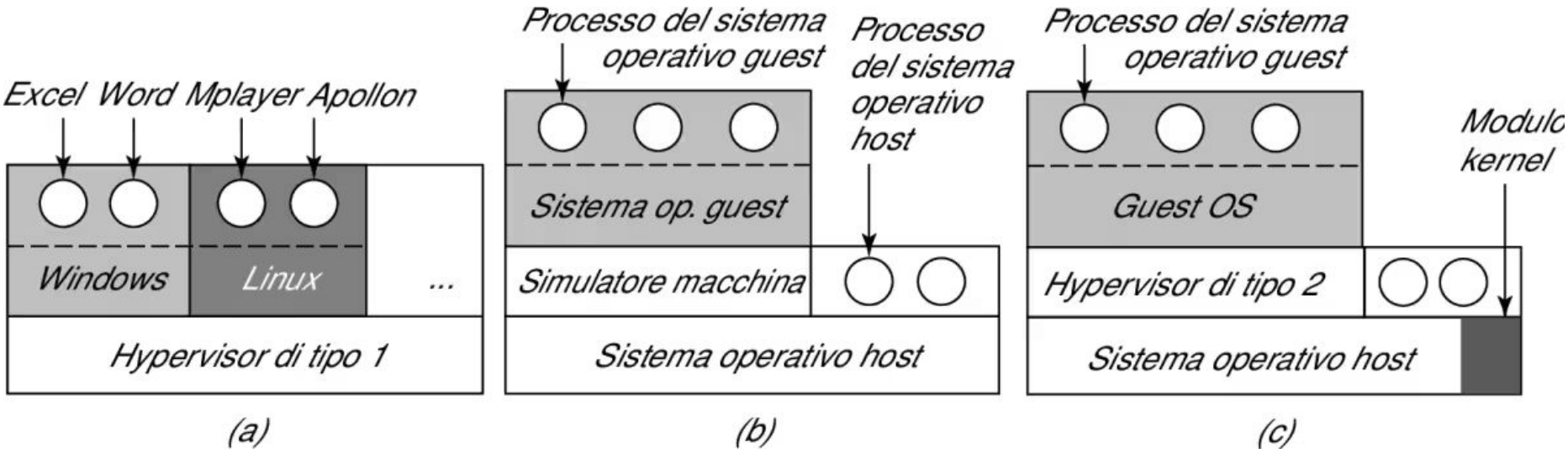
© Deitel & Ass. Inc.

Sistemi Operativi – Macchina Virtuale - Esempio



Struttura del sistema VM/370 con CMS

Sistemi Operativi – Macchina Virtuale - Hypervisor



(a) Type 1 (b) Type 2 Puro (c) Type 2 Pratico

JVM e Container

- Java Virtual Machine
  - Il compilatore Java produce codice per la JVM che viene interpretato
  - Più sicuro (se correttamente implementato) e maggiore controllo sulle risorse

- Container
  - Eseguono codice solo in modalità Utente
  - Tutti i container condividono il Kernel dell'host
  - Non contiene un sistema operativo completo
  - Non è possibile eseguire un container con un SO diverso dall'host
  - Non esiste partizionamento delle risorse (isolati solo per processo, se compromette il Kernel il sistema crasha)

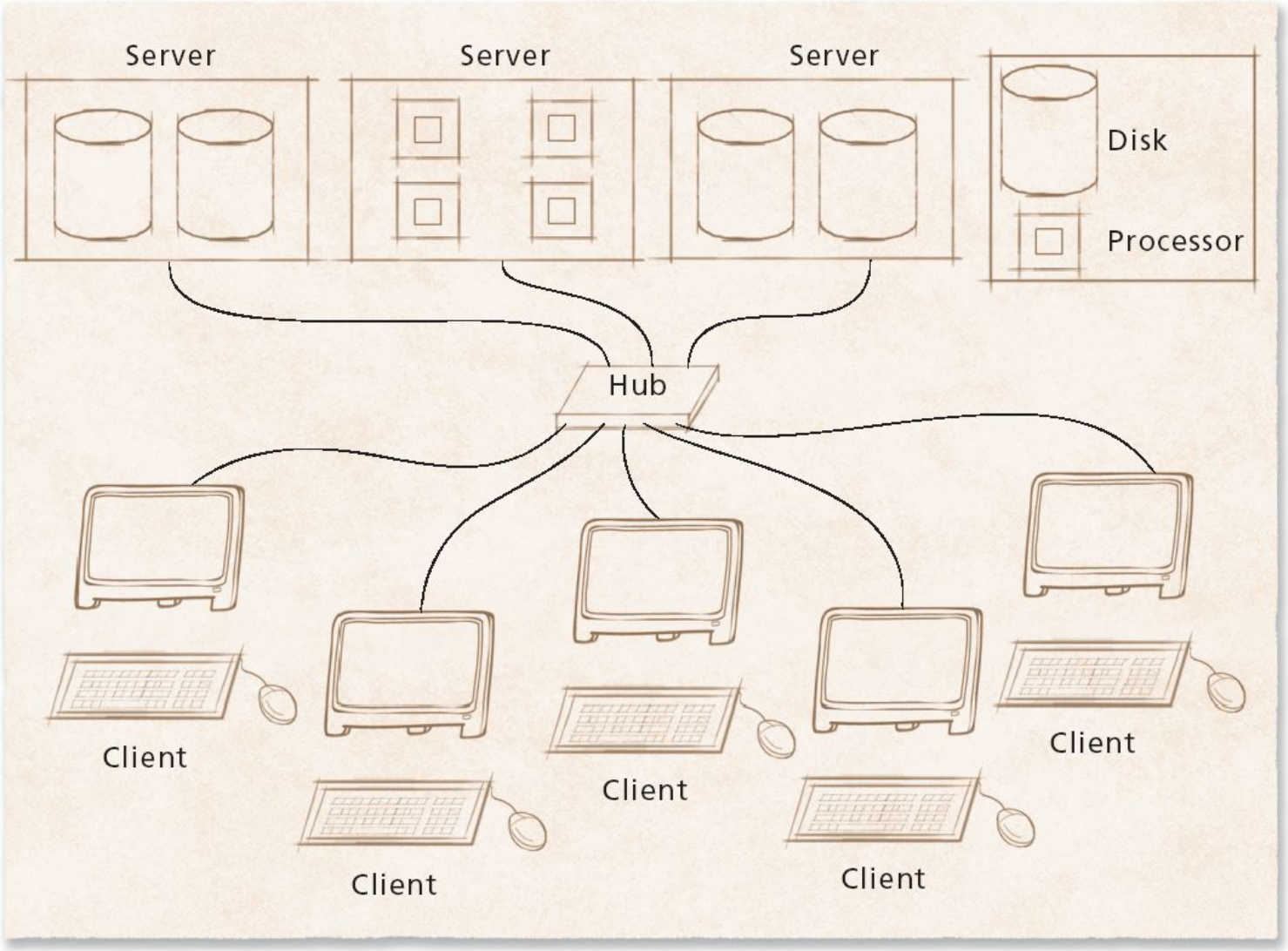
Sistemi Operativi di Rete e Sistemi Operativi Distribuiti

- Sistema Operativo di Rete
  - Eseguito su un computer
  - Permette ai suoi processi di accedere alle risorse sui computer remoti

- Sistema Operativo Distribuito
  - Sistema Operativo unico
  - Gestisce le risorse su un insieme di sistemi (computer)
  - Gli obiettivi includono:
    - Trasparenza di accesso, di uso, di prestazioni, di replicazione,…
    - Scalabilità
    - Tolleranza ai guasti
    - Consistenza

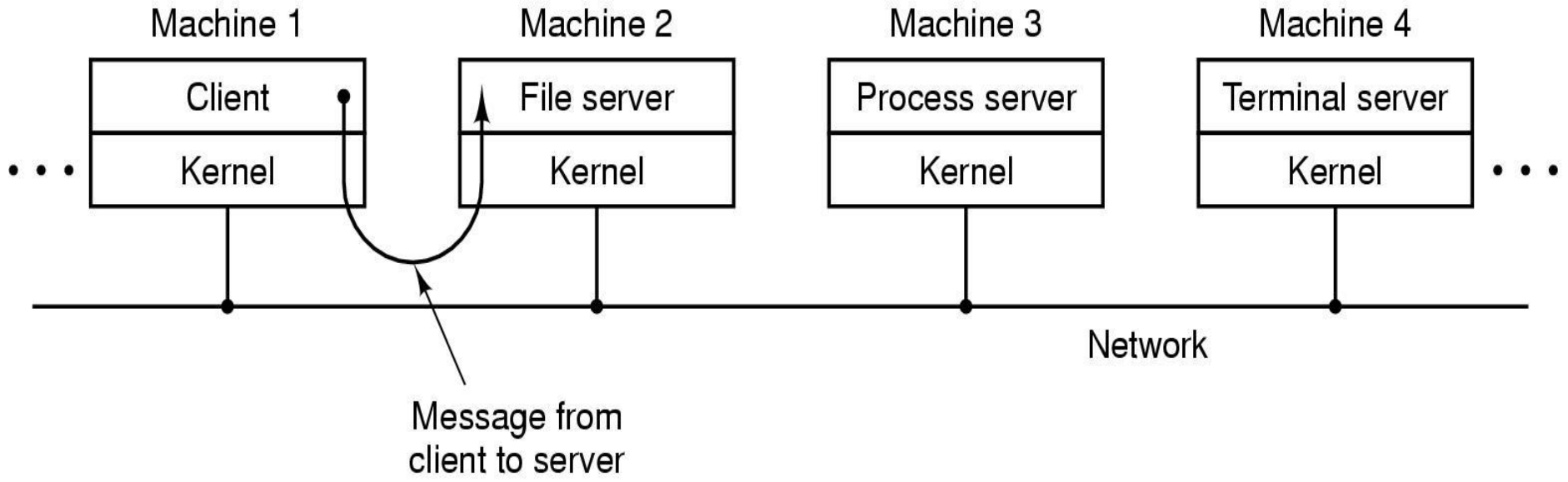
Sistemi Operativi di Rete e Sistemi Operativi Distribuiti

Modello di un sistema operativo di rete di tipo cliente/servente



© Deitel & Ass. Inc.

Esempi di strutture di Sistemi Operativi



Modello cliente-servente in un sistema distribuito

Sistemi Operativi Distribuiti e Middleware

- **Middleware** è un software per sistemi distribuiti
  - Permette interazioni tra più processi in esecuzione su uno o più computer in una rete
  - Facilita sistemi distribuiti **eterogenei**
  - Semplifica la programmazione delle applicazioni
  - Es: *Open Data Base Connectivity* (ODBC)
    - API per accesso a database
    - Permette alle applicazioni di accedere ai database tramite middleware chiamato un driver ODBC