

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

S. Balsamo – Università Ca' Foscari Venezia – SO1.

78 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**

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO1.78

79 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

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO1.79

80 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

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO1.80

Sistemi Operativi – concetti base – interruzioni

81

- **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

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO1.81

Sistemi Operativi – concetti base – avvio

82

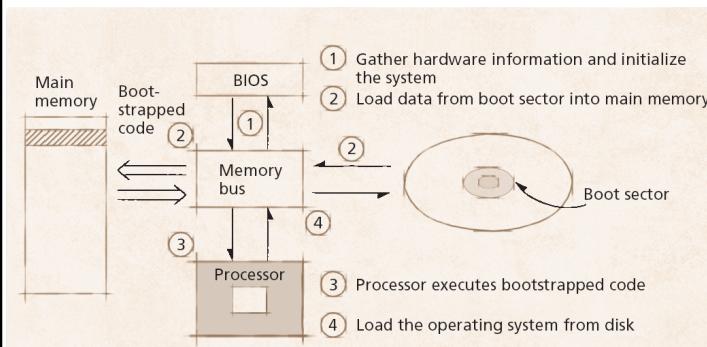
- **Bootstrapping:**
caricamento in memoria di componenti del sistema operativo **iniziali**
 - Eseguita dal *Basic Input/Output System (BIOS)*
 - **Inizializza** l'hardware di sistema
 - Carica le istruzioni in memoria principale da una regione di memoria secondaria chiamata il **settore di avvio (boot)**
 - 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.

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO1.82

Sistemi Operativi – concetti base – avvio

83



© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO1.83

Sistemi Operativi – concetti base – processi

84

- **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)**

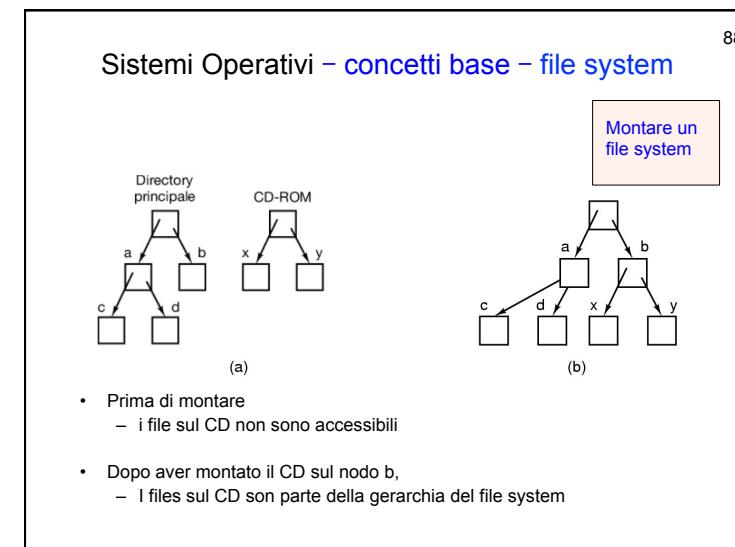
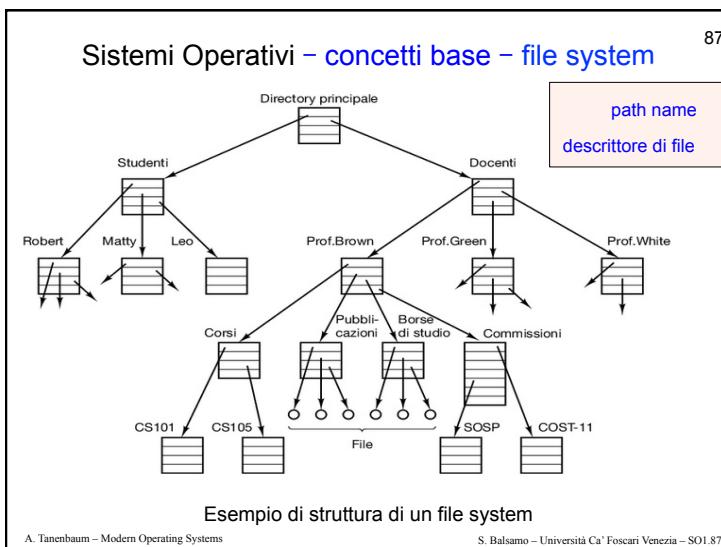
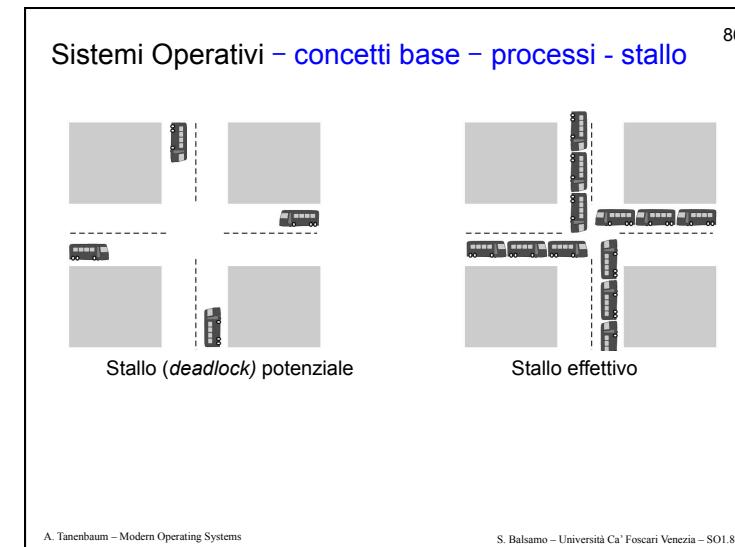
A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO1.84

Sistemi Operativi – concetti base – processi 85

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

A. Tanenbaum – Modern Operating Systems S. Balsamo – Università Ca' Foscari Venezia – SOI.85

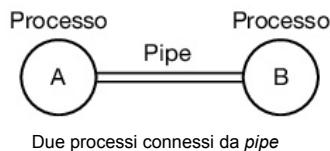


89

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



90

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)

91

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

92

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

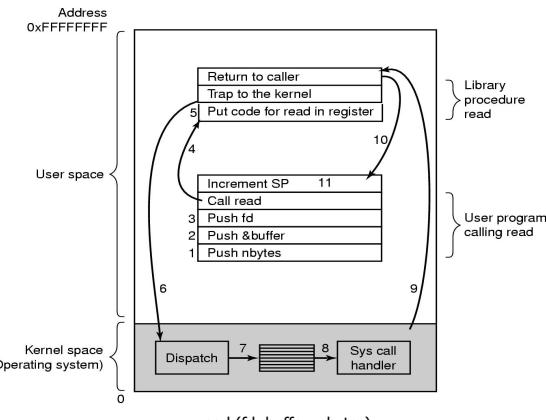
Esempio: *read* in Unix

read(fd, buffer, nbytes)

S. Balsamo – Università Ca' Foscari Venezia – SO1.93

93

Sistemi Operativi – concetti base – chiamate di sistema



S. Balsamo – Università Ca' Foscari Venezia – SO1.94

94

Sistemi Operativi – concetti base – chiamate di sistema

Chiamate di sistema per la gestione di processi in POSIX

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

Chiamate di sistema per la gestione di file

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

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO1.95

95

Sistemi Operativi – concetti base – chiamate di sistema

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

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

Altre chiamate di sistema

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

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO1.96

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
Iseek	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

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO1.97

97

Sistemi Operativi - Componenti e Obiettivi

- 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

S. Balsamo – Università Ca' Foscari Venezia – SO1.98

98

Componenti centrali dei Sistemi Operativi

99

- 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

S. Balsamo – Università Ca' Foscari Venezia – SO1.99

Componenti centrali dei Sistemi Operativi

100

- 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

S. Balsamo – Università Ca' Foscari Venezia – SO1.100

Sistemi Operativi - Obbiettivi

101

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

S. Balsamo – Università Ca' Foscari Venezia – SOI.101

Sistemi Operativi - Architetture

102

- 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

S. Balsamo – Università Ca' Foscari Venezia – SOI.102

Sistemi Operativi – Architettura Monolitica

103

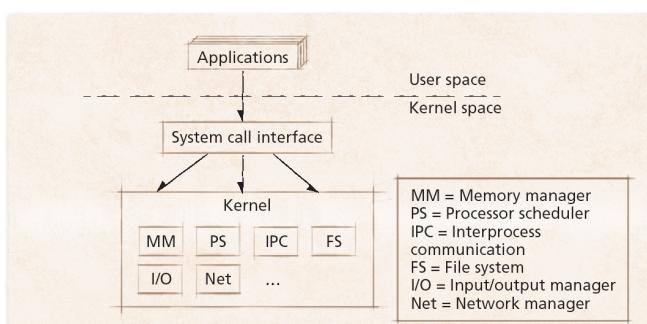
- Sistemi Operativi monolitici
 - Ogni componente è contenuta nel nucleo
 - Ogni componente può comunicare direttamente con qualsiasi altra
 - Obiettivo: elevata efficienza
 - Principale svantaggio: difficoltà a identificare eventuali fonti di errori

S. Balsamo – Università Ca' Foscari Venezia – SOI.103

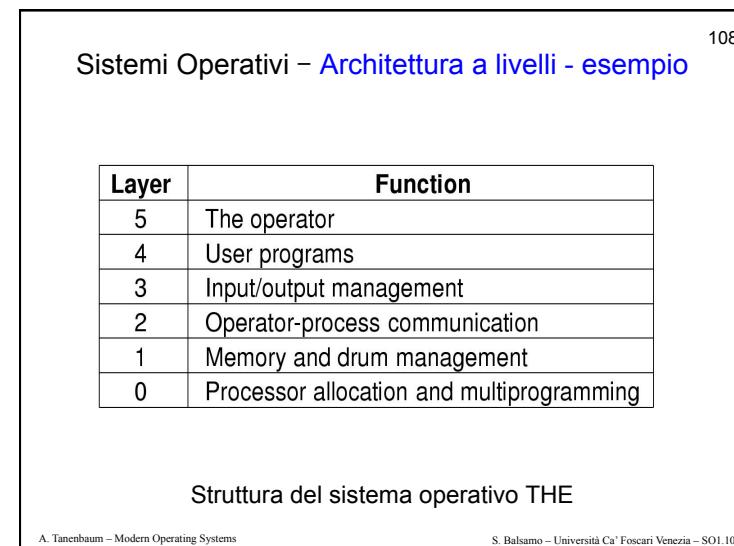
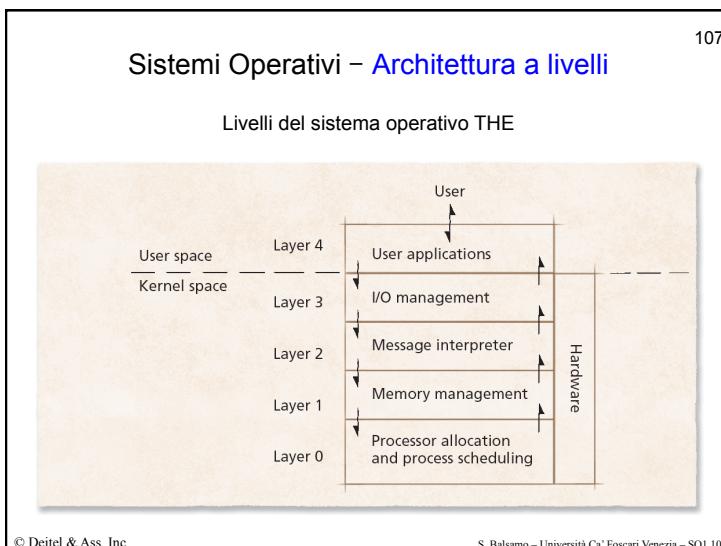
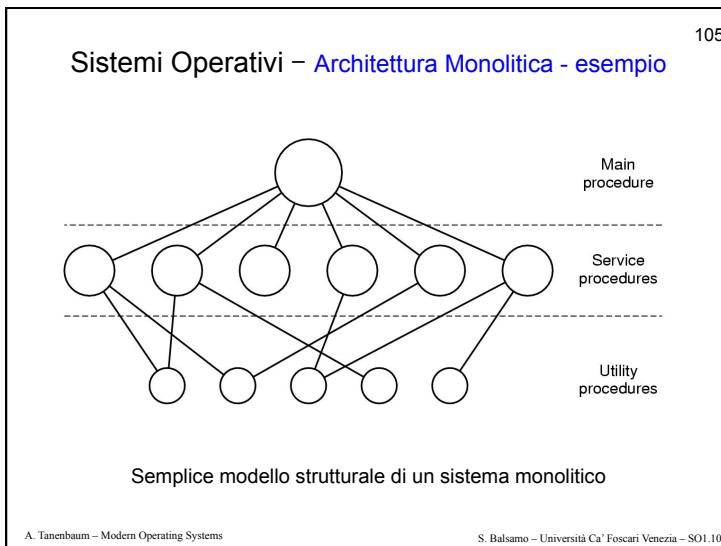
Sistemi Operativi – Architettura Monolitica

104

Architettura di un nucleo di sistema operativo monolitico



S. Balsamo – Università Ca' Foscari Venezia – SOI.104



Sistemi Operativi – Architettura Microkernel

109

- Architettura di un **SO Microkernel**

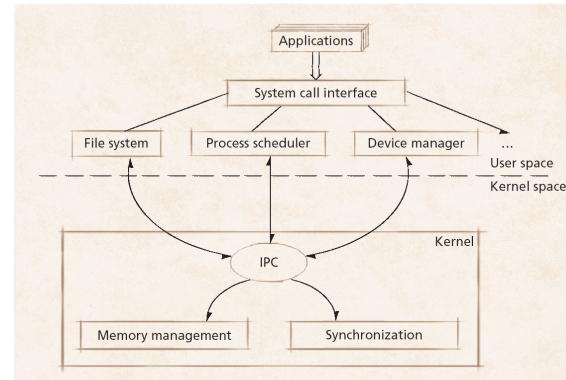
- Fornisce solo **servizi limitati**
 - Tentativo di contenere le dimensioni del kernel e garantire la scalabilità
- Elevato grado di **modularità**
 - **Estensibile**, portabile e scalabile
- Aumento del livello di comunicazione fra moduli
 - Può portare ad una degradazione delle prestazioni del sistema

S. Balsamo – Università Ca' Foscari Venezia – SOI.109

Sistemi Operativi – Architettura Microkernel

110

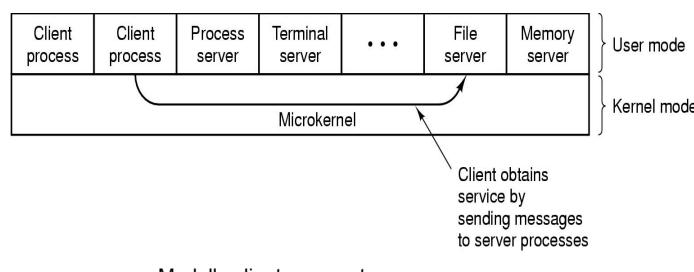
Architettura di un sistema operativo Microkernel



S. Balsamo – Università Ca' Foscari Venezia – SOI.110

Sistemi Operativi – Architettura Microkernel - esempio

111



Modello cliente-servente

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SOI.111

Sistemi Operativi – Macchina Virtuale

112

- **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**

S. Balsamo – Università Ca' Foscari Venezia – SOI.112

