Putra Pandu Adikara, S.Kom

Sistem Operasi

Struktur Sistem Operasi

Struktur Sistem Operasi

- Layanan (Service) Sistem Operasi
- Antarmuka pengguna Sistem Operasi
- Sistem Calls
- Tipe-Tipe Sistem Call
- Program Sistem
- Perancangan dan Implementasi Operasi Sistem
- Struktur Sistem Operasi
- Virtual Machine
- Generasi Sistem Operasi
- Boot Sistem

Tujuan

- Untuk menggambarkan pelayanan (service) sistem operasi yang disediakan kepada pengguna, proses, dan sistem lain
- Untuk membahas berbagai cara strukturisasi sistem operasi
- Untuk menjelaskan bagaimana sistem operasi diinstal dan disesuaikan dan bagaimana caranya boot

Layanan Sistem Operasi

- Satu set layanan (service) sistem operasi menyediakan fungsi yang membantu pengguna:
  - **User interface** - Hampir semua sistem operasi memiliki antarmuka pengguna (UI)
    - Bervariasi antara Command-Line (CLI), Grafik User Interface (GUI), Batch
  - **Eksekusi program** - Sistem harus dapat memuat program ke memori dan menjalankan program, mengakhiri eksekusi, secara normal atau abnormal (menunjukkan kesalahan)
  - **Operasi I/O** - program berjalan mungkin memerlukan I/O, yang mungkin melibatkan file atau peranti I/O.
  - **Manipulasi File sistem** - Program perlu membaca dan menulis file dan direktori, membuat dan menghapus mereka, pencarian mereka, daftar file Informasi, manajemen izin.

Layanan Sistem Operasi (Lanj.)

- Satu set layanan (service) sistem operasi menyediakan fungsi yang membantu pengguna:
  - **Komunikasi** - Proses dapat bertukar informasi, pada komputer yang sama atau antara komputer melalui jaringan
    - Komunikasi dapat melalui memori bersama atau melalui message passing (paket dipindahkan oleh OS)
  - **Deteksi kesalahan** - OS perlu selalu menyadari kemungkinan kesalahan
    - Dapat terjadi di CPU dan perangkat keras memori, peranti I/O, dalam program pengguna
    - Untuk setiap jenis kesalahan, OS harus mengambil tindakan yang sesuai untuk memastikan komputasi yang benar dan konsisten
    - Fasilitas Debugging dapat sangat meningkatkan pengguna dan kemampuan programmer untuk efisien menggunakan sistem

Layanan Sistem Operasi (Lanj.)

- Set lain dari fungsi OS untuk menjamin operasi yang efisien dari sistem itu sendiri melalui pembagian sumber daya
  - **Alokasi sumber daya** - Ketika multiple user atau beberapa pekerjaan berjalan bersamaan, sumber daya harus dialokasikan untuk masing-masing
    - Banyak jenis sumber daya - Beberapa (seperti siklus CPU, memoriu tama, dan penyimpanan file) mungkin memiliki kode alokasi khusus, yang lain (seperti peranti I/O) mungkin punya permintaan umum dan rilis kode.
  - **Akuntansi** - Untuk melacak pengguna mana yang menggunakan berapa banyak dan apa jenisnya dari sumber daya komputer
  - **Proteksi dan keamanan** - Pemilik informasi yang disimpan dalam sebuah sistem komputer multiuser atau jaringan mungkin ingin mengontrol penggunaan informasi tersebut, proses konkuren tidak boleh saling mengganggu
    - **Proteksi/Perlindungan** melibatkan memastikan bahwa semua akses ke sumber daya sistem dikendalikan
    - **Keamanan** sistem dari luar memerlukan otentikasi pengguna, diperluas untuk mempertahankan peranti I/O eksternal dari upaya akses tidak valid
    - Jika sistem harus dilindungi dan supaya aman, tindakan pencegahan harus dilembagakan keseluruhan. Sebuah rantai hanya sekuat pada hubungan yang terlemah

Antarmuka Pengguna Sistem Operasi - CLI

- CLI memungkinkan pemasukan perintah langsung
  - Kadang-kadang diimplementasikan di kernel, kadang-kadang oleh program sistem
  - Kadang-kadang beberapa selera diimplementasikan – **shells**
  - Utamanya mengambil perintah dari pengguna dan mengeksekusinya
    - Kadang-kadang perintah built-in, kadang-kadang hanya nama program
      - Jika yang terakhir, menambahkan fitur baru tidak memerlukan modifikasi shell

❖ Linux

Shell Name	Developed by	Where	Remark
BASH (Bourne-Again SHell)	Brian Fox and Chet Ramey	Free Software Foundation	Most common shell in Linux. It's Freeware shell.
CSH (C SHell)	Bill Joy	University of California (For BSD)	The C shell's syntax and usage are very similar to the C programming language.
KSH (Korn SHell)	David Korn	AT & T Bell Labs	--
TCSH	See the man page. Type $ man tcsh	--	TCSH is an enhanced but completely compatible version of the Berkeley UNIX C shell (CSH).

❖ MS-DOS

- shell-nya adalah COMMAND.COM

User Operating System Interface - GUI

- Antarmuka metafora **desktop** yg user-friendly
  - Biasanya mouse, keyboard, dan monitor
  - **Ikon** mewakili file, program, tindakan, dll
  - Berbagai tombol mouse di atas objek dalam antarmuka menyebabkan berbagai tindakan (memberikan informasi, pilihan, melaksanakan fungsinya, membuka direktori (dikenal sebagai **folder**)
  - Diciptakan di Xerox PARC
- Banyak sistem sekarang mencakup antarmuka CLI dan GUI
  - Microsoft Windows GUI dengan shell CLI "command“, “PowerShell”
  - Apple Mac OS X sebagai interface GUI "Aqua" dengan kernel UNIX yang tersedia di bawah dan shell
  - Solaris CLI dengan antarmuka GUI opsional (Java Desktop, KDE)

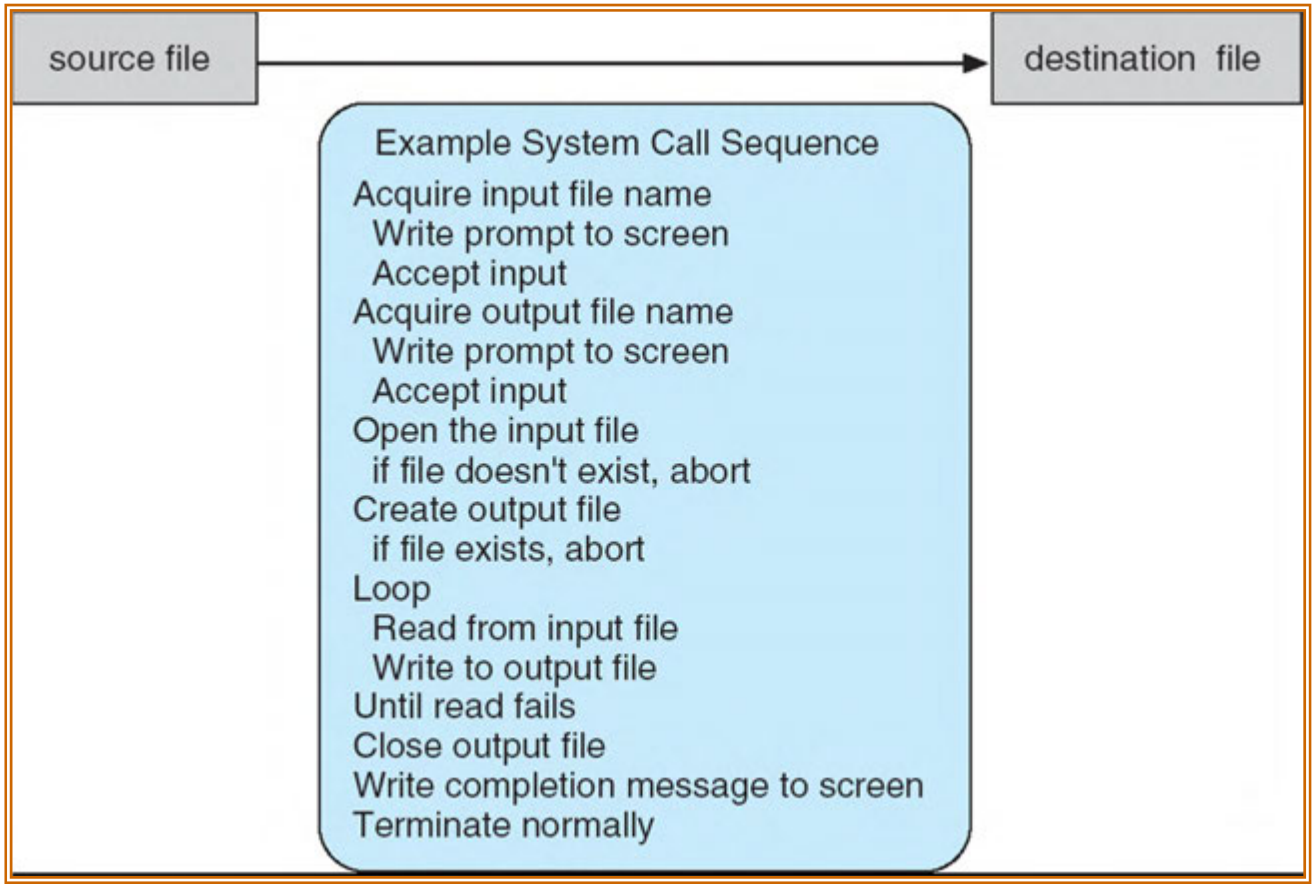
System Calls

- Antarmuka pemrograman untuk layanan yang diberikan oleh OS
- Biasanya ditulis dalam bahasa tingkat tinggi (C atau C + +)
- Sebagian besar diakses oleh program melalui **Application Program Interface (API)** tingkat-tinggi daripada menggunakan sistem direct call
- Tiga API yang paling umum adalah:
  - Win32 API untuk Windows,
  - POSIX API untuk sistem berbasis POSIX (termasuk hampir semua versi UNIX, Linux, dan Mac OS X), dan
  - Java API untuk Java virtual machine (JVM)
- Mengapa menggunakan API daripada system calls?

(Catatan bahwa nama-nama sistem call yang digunakan di seluruh teks ini adalah generik)

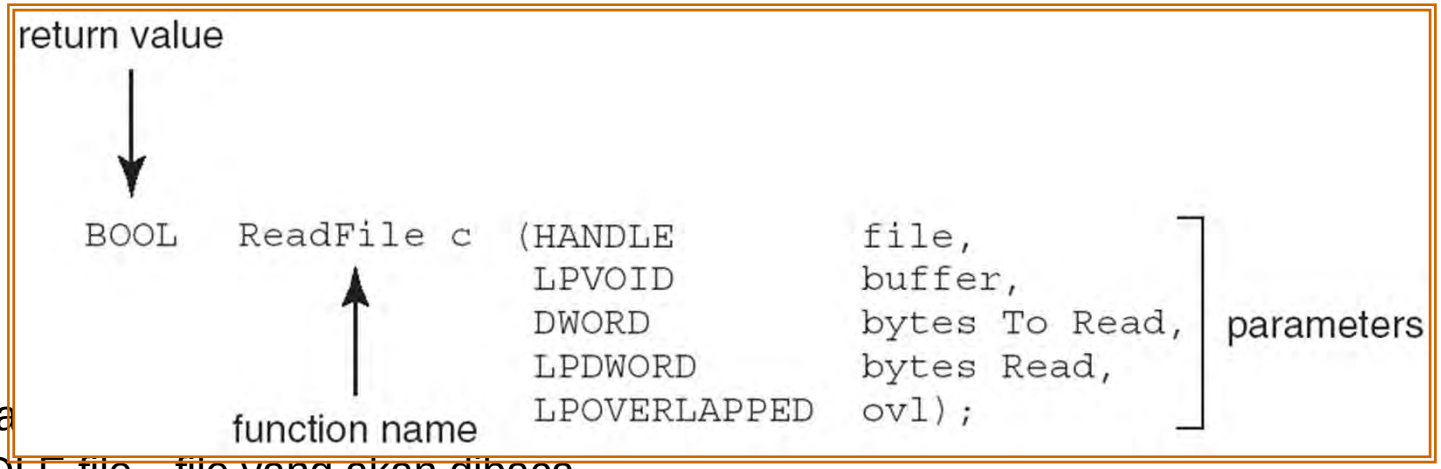
Contoh System Calls

- Urutan system call sequence untuk mengkopi isi dari satu file ke file lain



Contoh Standard API

- Tinjau function **ReadFile()** di
- Win32 API—fungsi pembacaan dari file

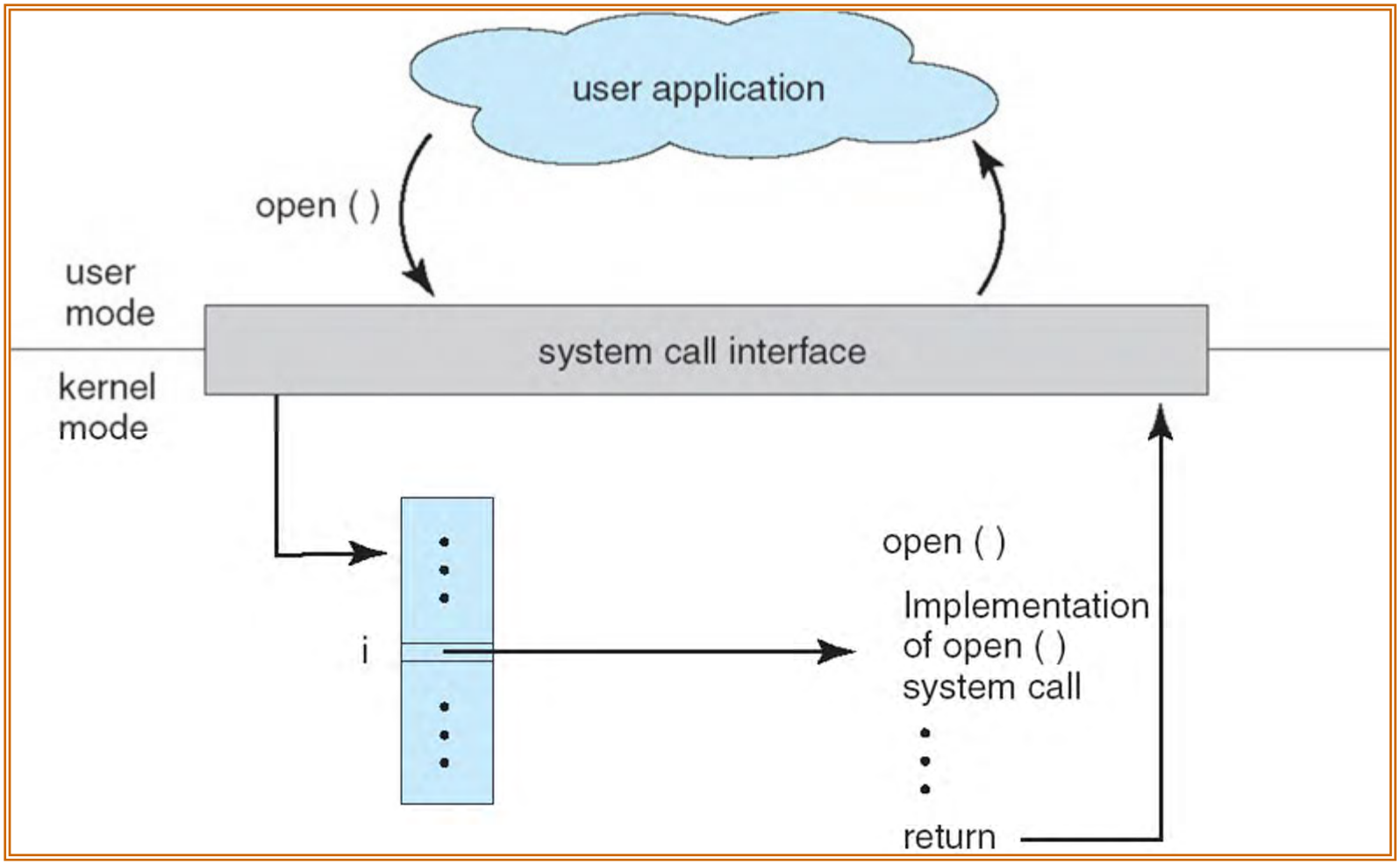


- Keteranga
  - HANDLE file—file yang akan dibaca
  - LPVOID buffer—buffer dimana data akan dibaca ke dan ditulis dari
  - DWORD bytesToRead—ukuran jumlah bytes yang akan dibaca ke buffer
  - LPDWORD bytesRead—ukuran jumlah bytes yang dibaca sejak pembacaan terakhir
  - LPOVERLAPPED ovl—indikasi jika overlap I/O sedang digunakan

Implementasi System Call

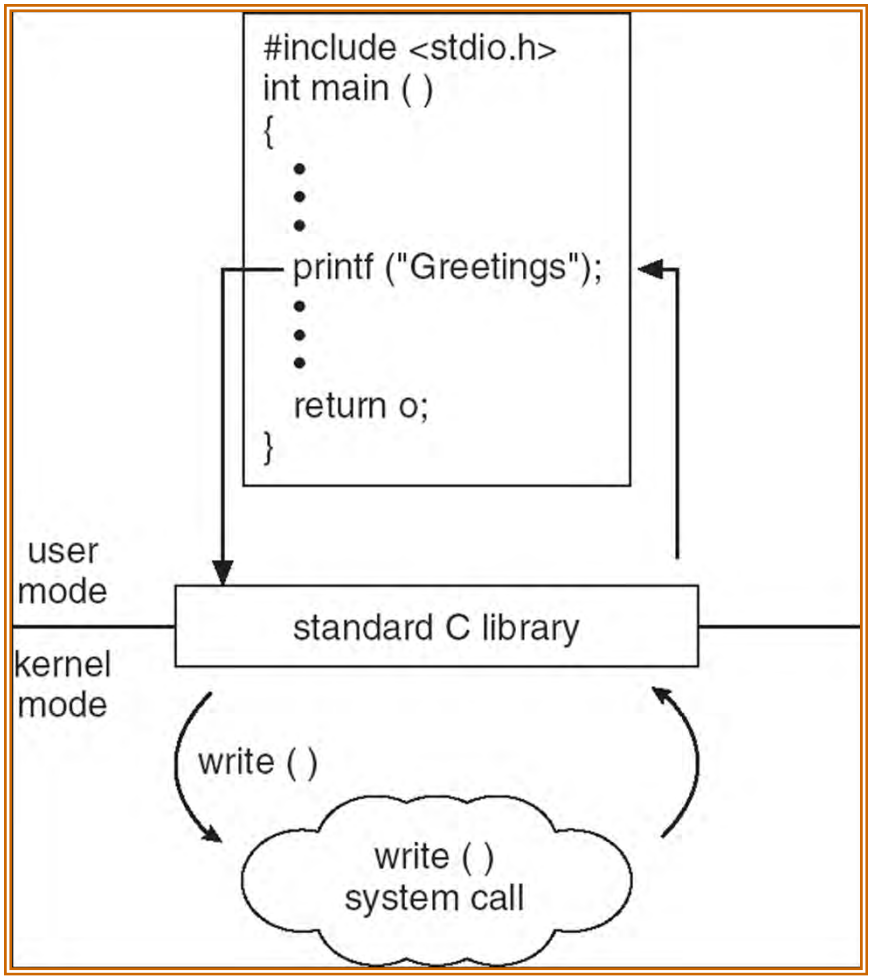
- Biasanya, nomor dikaitkan dengan setiap sistem call
  - System-call interface memelihara sebuah tabel berindeks menurut nomor-nomor ini
- Antarmuka system call memanggil system call yang diinginkan di kernel OS dan mengembalikan status dari sistem call dan nilai-nilai hasil baliknya
- Caller tidak perlu tahu bagaimana system call diimplementasikan
  - Hanya perlu mematuhi API dan mengerti apa OS akan melakukannya sebagai suatu hasil panggilan
  - Sebagian besar rincian antarmuka OS disembunyikan dari programmer menggunakan API
    - Dikelola oleh library yg mendukung run-time (set fungsi yang dibangun dalam library yg disertakan dengan compiler)

API – System Call – OS Relationship



Contoh Standar C Library

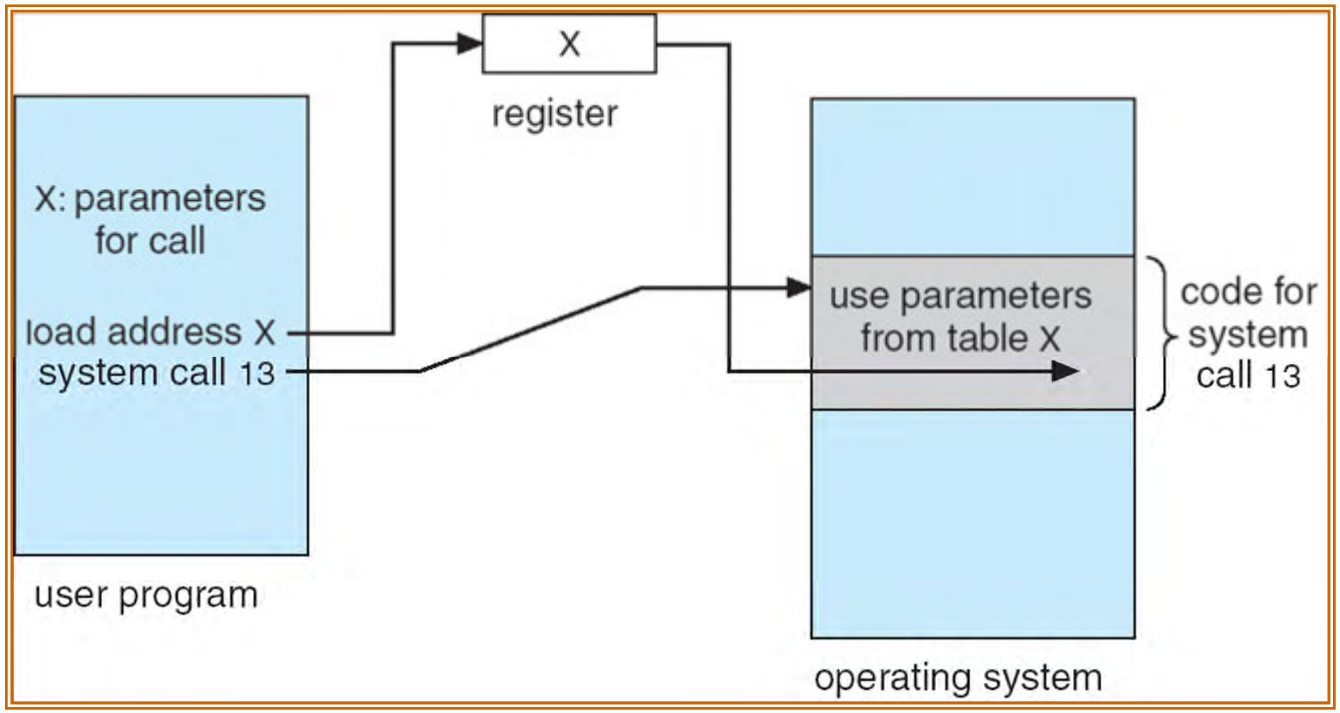
- Program C memanggil call library printf(), yang mana memanggil system call write()



System Call Parameter Passing

- Seringkali, informasi lebih diperlukan dari sekedar identitas system call yang diinginkan
  - Tipe pasti tepat dan jumlah informasi yang bervariasi sesuai dengan OS dan call nya
- Tiga metode umum yang digunakan untuk passing parameter ke OS
  - Sederhana: pass parameter dalam register
    - Dalam beberapa kasus, mungkin parameter lebih dari register
  - Parameter disimpan dalam *bloc*k, atau table, dalam memori, dan alamat block dilewatkan sebagai parameter di register
    - Pendekatan yang diambil oleh Linux dan Solaris
  - Parameter ditempatkan, atau di-*push*, ke *stack* oleh program dan di-*pop* dari stack oleh sistem operasi
  - Metode Block dan stack tidak membatasi jumlah atau panjang parameter yang dilewatkan

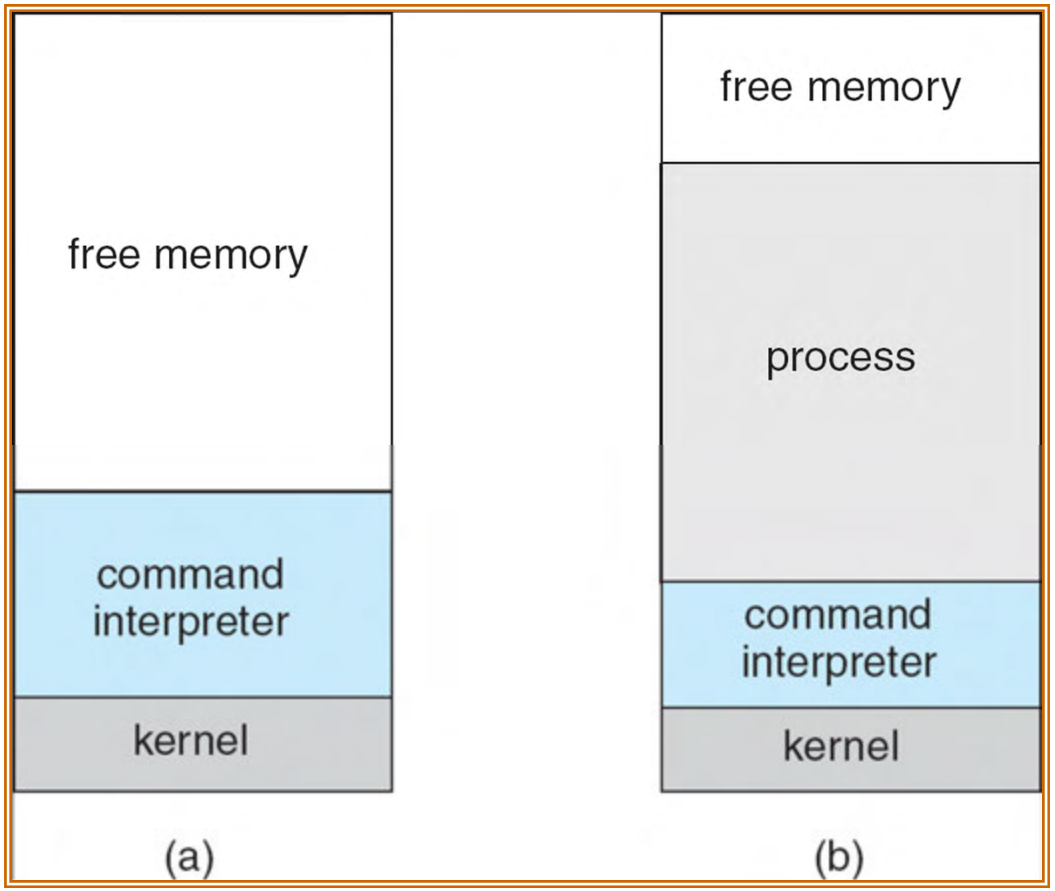
Parameter Passing via Table



Tipe-Tipe System Calls

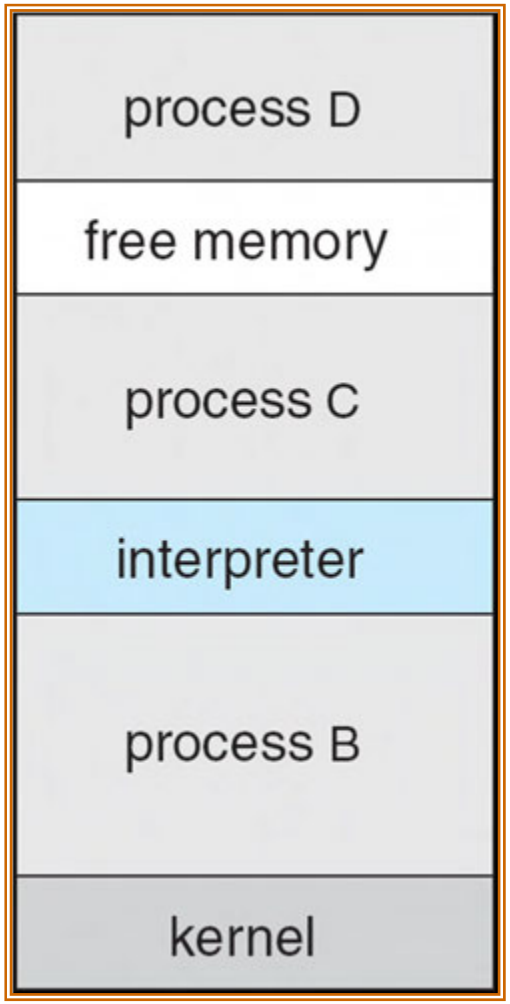
- Kontrol proses
- Manajemen file
- Manajemen peranti
- Pemeliharaan informasi
- Komunikasi

Execution MS-DOS



(a) At system startup (b) running a program

FreeBSD Running Multiple Programs



System Programs

- Program sistem menyediakan lingkungan yang nyaman untuk pengembangan program dan eksekusi. Ini dapat dibagi menjadi:
  - Manipulasi Berkas
  - Informasi status
  - Modifikasi File
  - Dukungan bahasa pemrograman
  - Pemuatan dan eksekusi program (program loading dan execution)
  - Komunikasi
  - Program aplikasi
- Pengguna sebagian besar memandang tampilan sistem operasi didefinisikan oleh program sistem, bukan sistem call sebenarnya

Solaris 10 dtrace Following System Call

```
# ./all.d `pgrep xclock` XEventsQueued
dtrace: script './all.d' matched 52377 probes
CPU FUNCTION
  0 -> XEventsQueued                       U
  0   -> _XEventsQueued                    U
  0     -> _X11TransBytesReadable          U
  0     <- _X11TransBytesReadable          U
  0     -> _X11TransSocketBytesReadable    U
  0     <- _X11TransSocketBytesreadable    U
  0     -> ioctl                           U
  0       -> ioctl                         K
  0         -> getf                        K
  0           -> set_active_fd             K
  0           <- set_active_fd             K
  0         <- getf                        K
  0         -> get_udatamodel              K
  0         <- get_udatamodel              K
...
  0         -> releasef                    K
  0           -> clear_active_fd           K
  0           <- clear_active_fd           K
  0           -> cv_broadcast              K
  0           <- cv_broadcast              K
  0         <- releasef                    K
  0       <- ioctl                         K
  0     <- ioctl                           U
  0   <- _XEventsQueued                    U
  0 <- XEventsQueued                       U
```

System Programs

- Menyediakan lingkungan yang nyaman untuk pengembangan dan eksekusi program
  - Beberapa darinya hanya user interface untuk sistem call; lainnya jauh lebih kompleks
- Manajemen file - Membuat, menghapus, menyalin, mengubah nama, mencetak, dump, daftar, dan umumnya memanipulasi file dan direktori
- Informasi status
  - Beberapa meminta sistem untuk informasi - tanggal, waktu, jumlah memori yang tersedia, ruang disk, jumlah pengguna
  - Lainnya menyediakan informasi rincian performa, logging, dan debugging
  - Biasanya, program-program ini memformat dan mencetak output ke terminal atau perangkat output lainnya
  - Beberapa sistem menerapkan registri - digunakan untuk menyimpan dan mengambil informasi konfigurasi

Program Sistem (cont'd)

- Modifikasi File
  - Editor Teks untuk membuat dan memodifikasi file
  - Perintah khusus untuk mencari isi dari file atau melakukan transformasi dari teks
- Dukungan bahasa pemrograman – compiler, assembler, debugger dan interpreter kadang-kadang diberikan
- Pemuatan dan eksekusi program – Absolute loader, relocatable loader, linkage editor, dan overlay loader, sistem debugging untuk tingkat yang lebih tinggi dan bahasa mesin
- Komunikasi - Menyediakan mekanisme untuk membuat hubungan virtual antar proses, pengguna, dan sistem komputer
  - Memungkinkan pengguna untuk mengirim pesan dari satu layar ke layar lain, browse halaman web, mengirim pesan email, login jarak jauh, transfer file dari satu mesin yang lain

Desain and Implementasi Sistem Operasi

- Desain dan Implementasi OS tidak “solvable", tetapi beberapa pendekatan telah terbukti sukses
- Struktur internal Sistem Operasi yang berbeda dapat sangat bervariasi
- Mulailah dengan menentukan tujuan dan spesifikasi
- Dipengaruhi oleh pilihan perangkat keras, jenis sistem
- Tujuan User dan tujuan System
  - Tujuan User - sistem operasi harus nyaman digunakan, mudah dipelajari, dapat diandalkan, aman, dan cepat
  - Tujuan System - sistem operasi harus mudah untuk dirancang, diimplementasikan, dan dipelihara, serta fleksibel, handal, bebas kesalahan, dan efisien

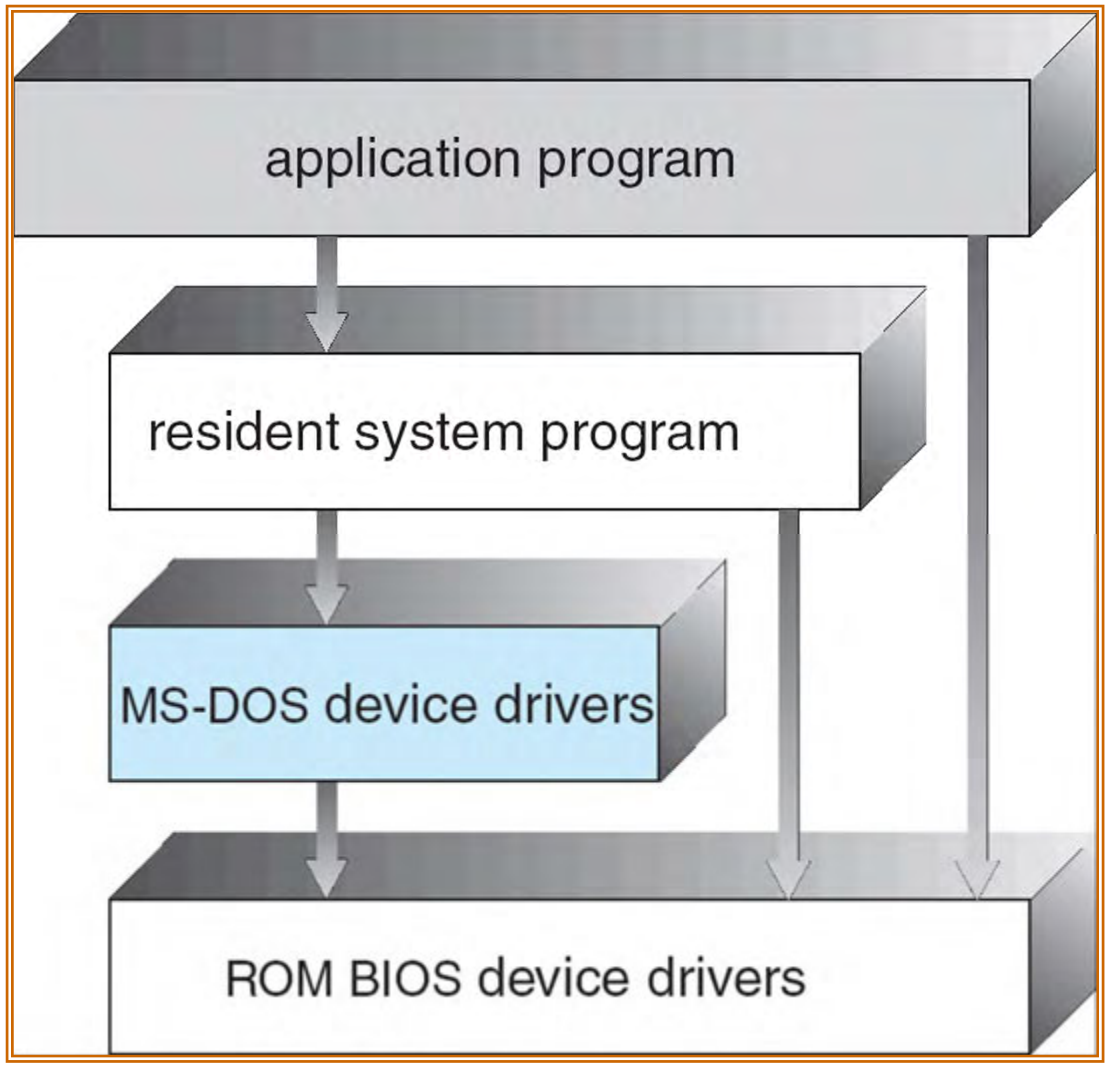
Desain and Implementasi Sistem Operasi (Lanj.)

- Prinsip penting untuk memisahkan
  - **Kebijakan (Policy)**: Apa yang akan dilakukan?
  - **Mekanisme (Mechanism)**: Bagaimana melakukannya?
- Mekanisme menentukan bagaimana melakukan sesuatu, kebijakan memutuskan apa yang akan dilakukan
  - Pemisahan kebijakan dari mekanisme adalah prinsip yang sangat penting, hal itu memungkinkan fleksibilitas maksimal jika keputusan kebijakan harus diubah nanti

Simple Structure

- MS-DOS - dibuat untuk memberikan fungsionalitas paling banyak dalam ruang sedikit
  - Tidak dibagi dalam modul-modul
  - Meskipun MS-DOS memiliki struktur tertentu, tingkat antarmuka dan fungsionalitas tidak dipisahkan secara baik

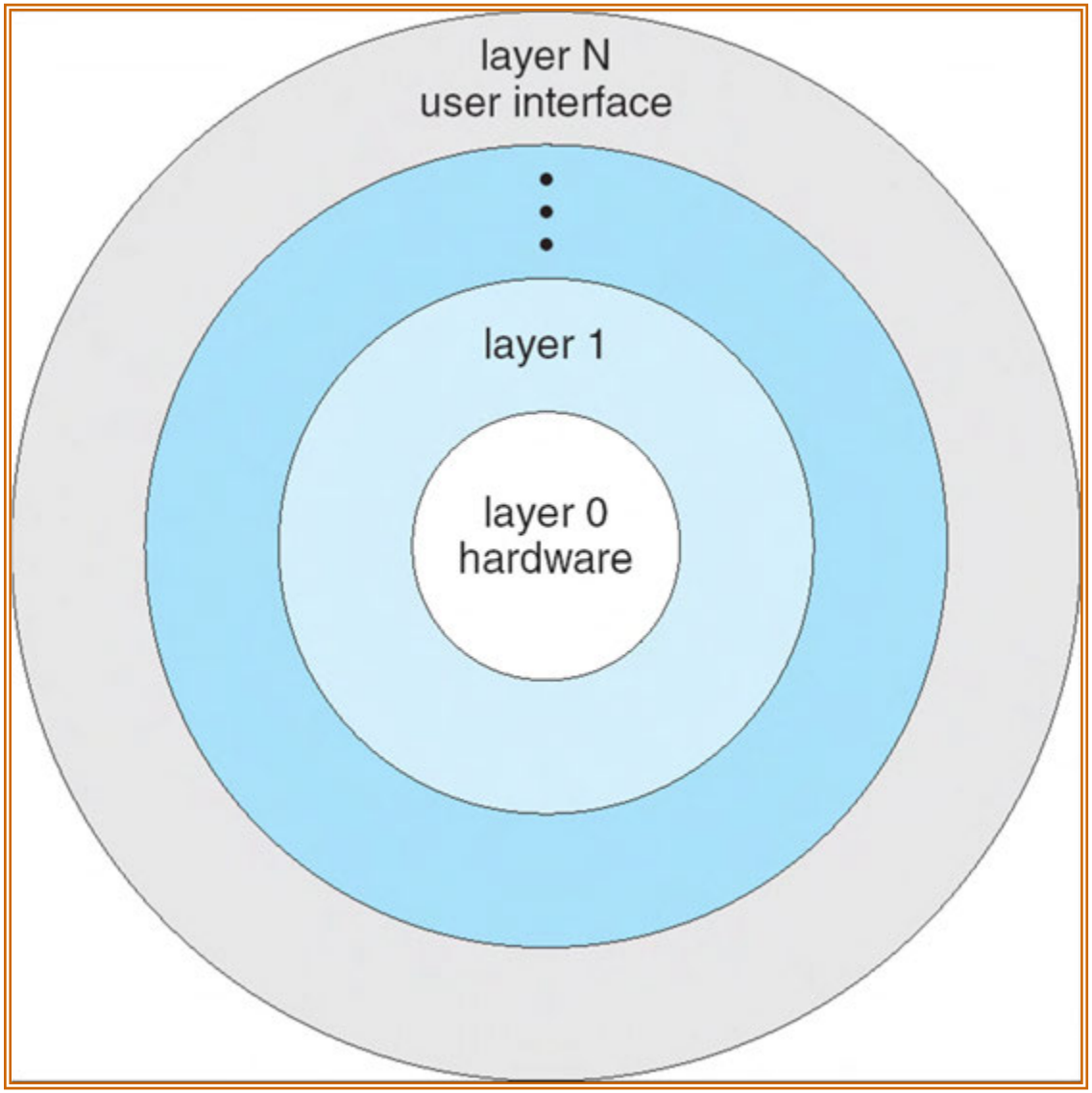
Struktur Lapisan MS-DOS



Pendekatan Berlapis

- Sistem operasi dibagi menjadi beberapa lapisan (tingkat), masing-masing dibangun di atas lapisan bawahnya. Lapisan bawah (layer 0), adalah perangkat keras, sedangkan (lapisan N) tertinggi adalah user interface.
- Dengan modularitas, lapisan yang dipilih sedemikian rupa sehingga masing-masing menggunakan fungsi (operasi) dan layanan (service) hanya lapisan ditingkat bawahnya

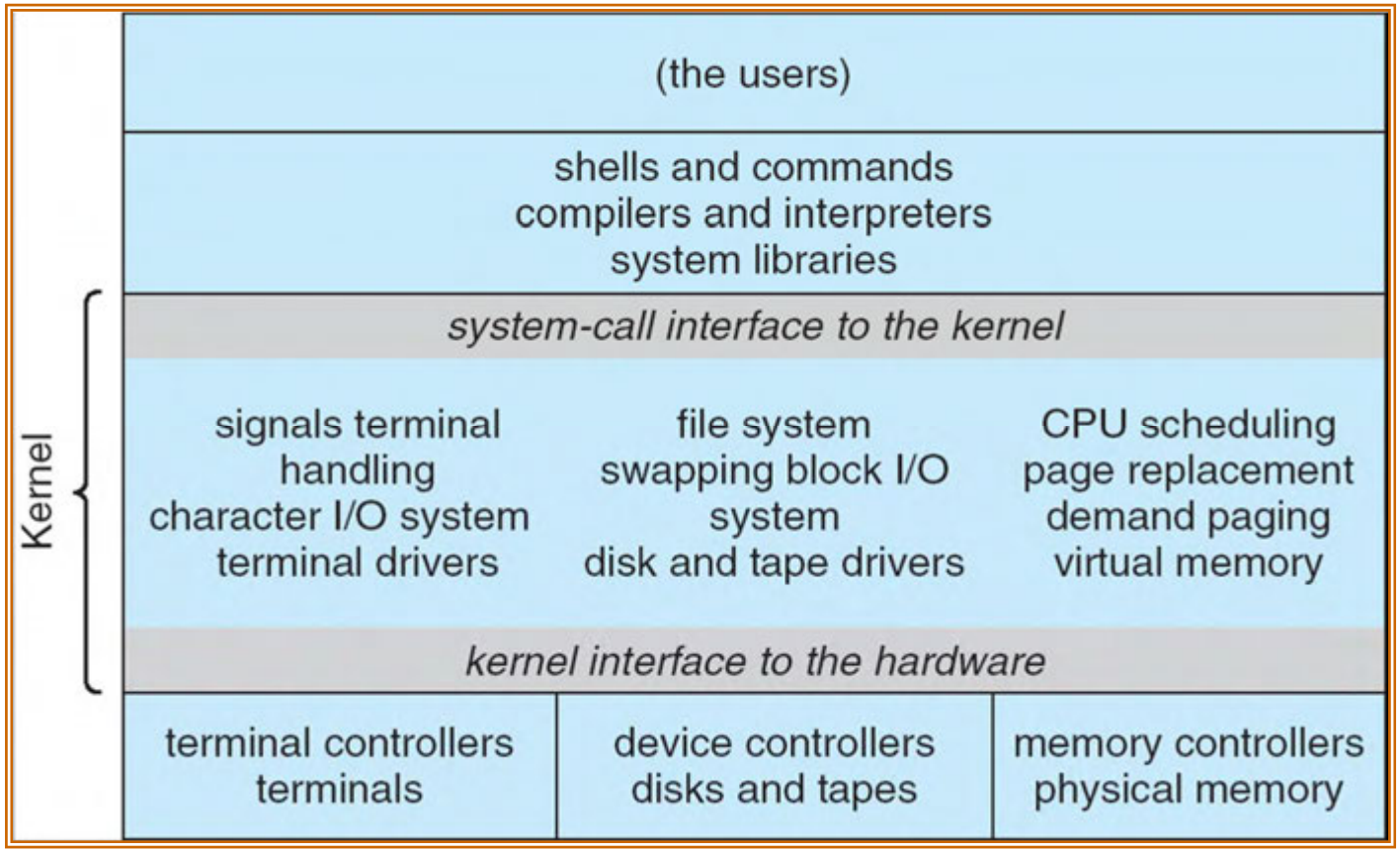
Layered Operating System



UNIX

- UNIX - dibatasi oleh fungsionalitas perangkat keras, sistem operasi UNIX asli memiliki struktur terbatas. OS UNIX terdiri dari dua bagian terpisah
  - Sistem program
  - Kernel
    - Terdiri dari segala sesuatu di bawah antarmuka sistem -call dan di atas fisik perangkat keras
    - Menyediakan sistem berkas, penjadwalan CPU, manajemen memori, dan sistem operasi fungsi; sejumlah besar fungsi untuk satu tingkat

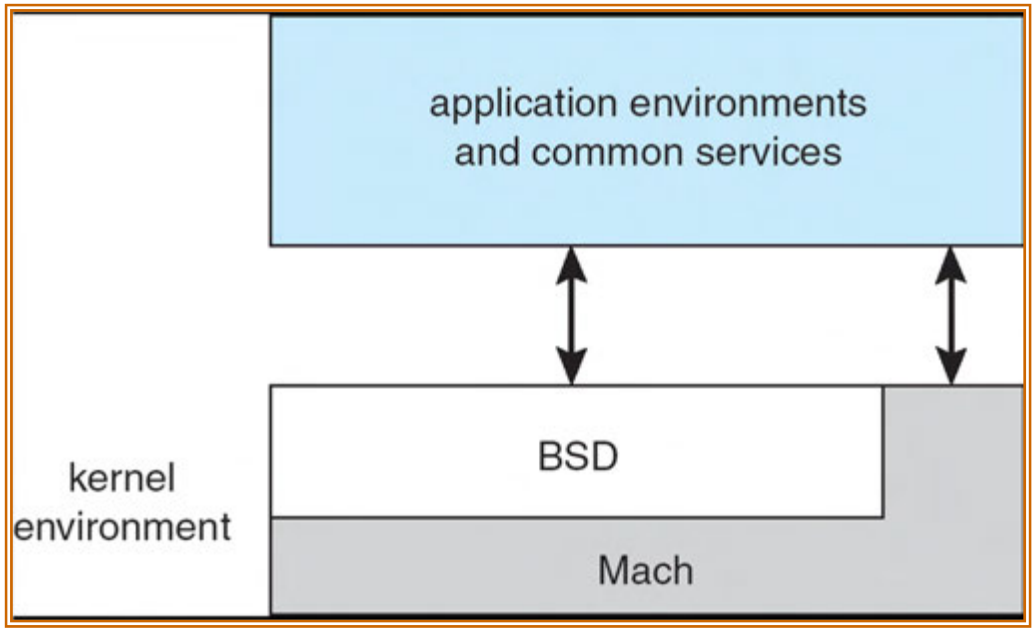
Struktur Sistem UNIX



Microkernel System Structure

- Berpindah sebisa mungkin dari kernel ke ruang “user“
- Komunikasi terjadi antara module user menggunakan passing message
- Manfaat:
  - Lebih mudah untuk mengembangkan mikrokernel
  - Lebih mudah untuk porting sistem operasi ke arsitektur baru
  - Lebih dapat diandalkan (sedikit kode berjalan dalam mode kernel)
  - Lebih aman
- Kekurangan:
  - Overhead kinerja dari komunikasi ruang pengguna ke ruang kernel

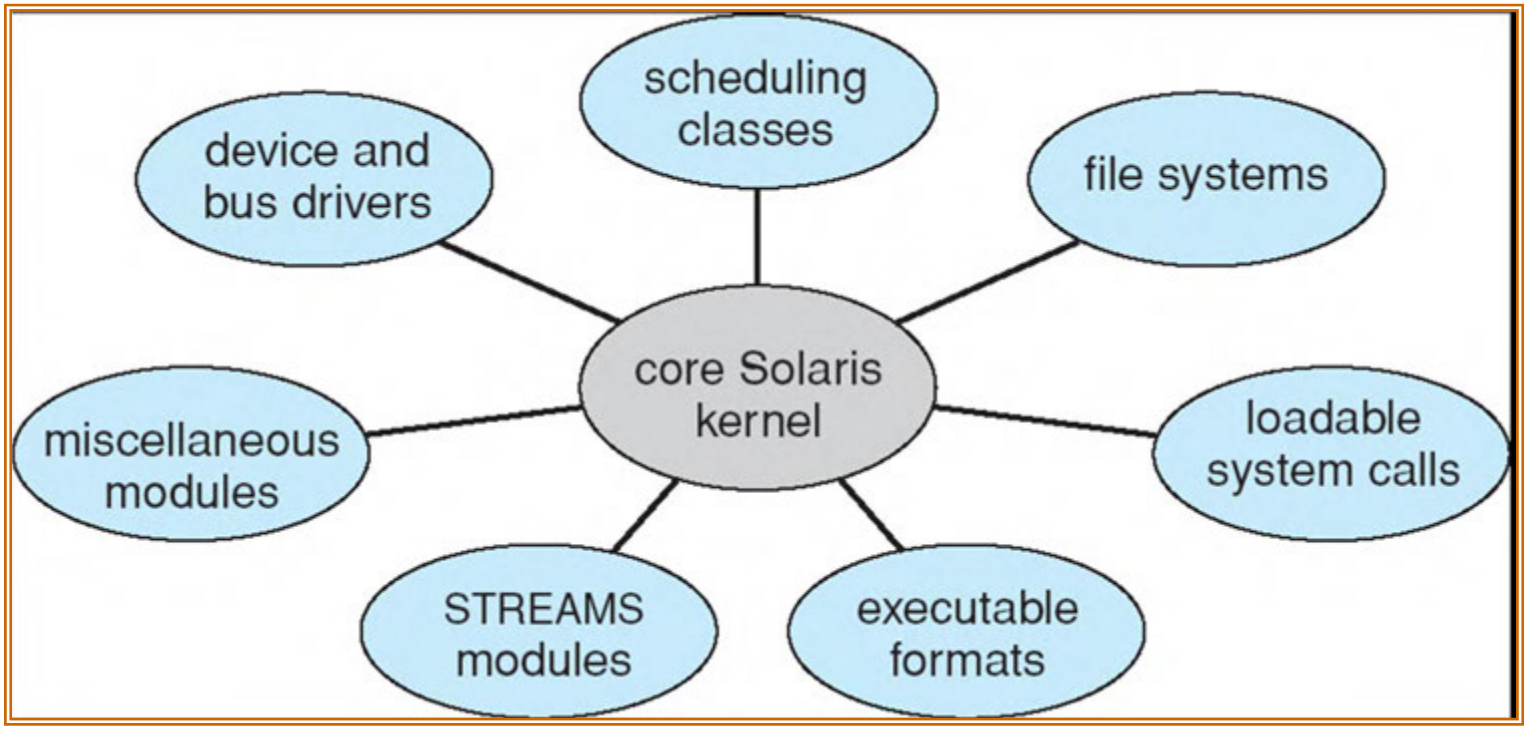
Struktur Mac OS X



Modul-Modul

- Sebagian besar sistem operasi modern menerapkan modul-modul kernel
  - Menggunakan pendekatan berorientasi obyek
  - Tiap modul komponen inti dipisahkan
  - Tiap modul “berbicara” ke yang lainnya melalui antarmuka yg dikenal
  - Tiap modul loadable sesuai keperluan di dalam kernel
- Secara keseluruhan, mirip dengan lapisan tetapi lebih fleksibel

Pendekatan Modular Solaris



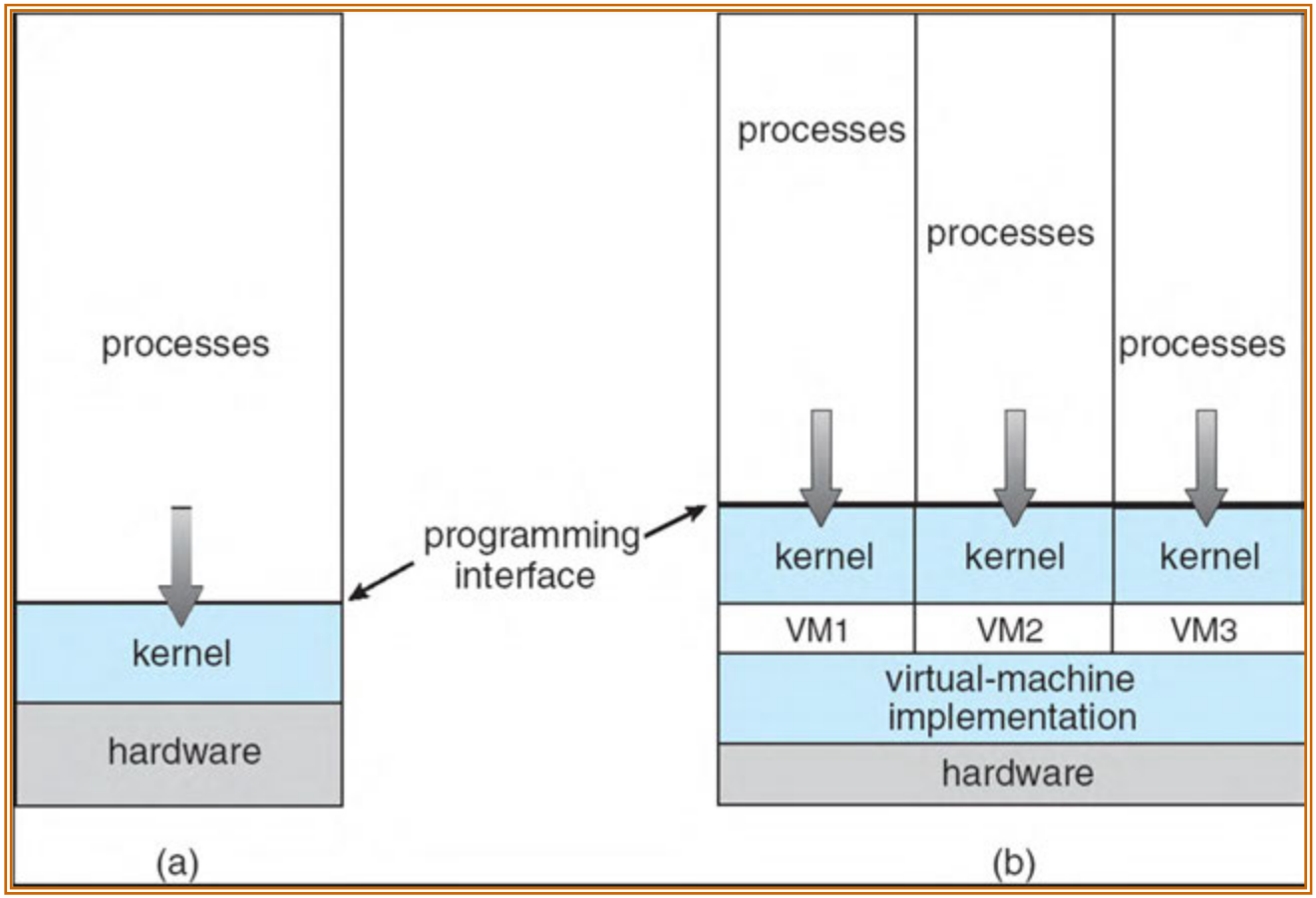
Virtual Machine

- **Virtual machine** mengambil pendekatan berlapis untuk kesimpulan logis. Memperlakukan hardware dan kernel sistem operasi seolah-olah semuanya perangkat keras
- Sebuah mesin virtual menyediakan sebuah antarmuka yang *identik* dengan perangkat keras yang mendasari secara kasat mata
- Sistem operasi menciptakan ilusi dari beberapa proses, masing-masing dieksekusi pada prosesor sendiri dengan memori (virtual) sendiri

Virtual Machine (Lanj.)

- Sumber daya dari komputer fisik dibagi untuk membuat mesin virtual
  - Penjadwalan CPU dapat membuat penampakan bahwa pengguna memiliki prosesor sendiri
  - Spooling dan sistem berkas dapat memberikan card reader virtual dan line printer virtual
  - Terminal pengguna time-sharing biasa berfungsi sebagai console operator virtual machine

Virtual Machine (Lanj.)



(a) Nonvirtual machine (b) virtual machine

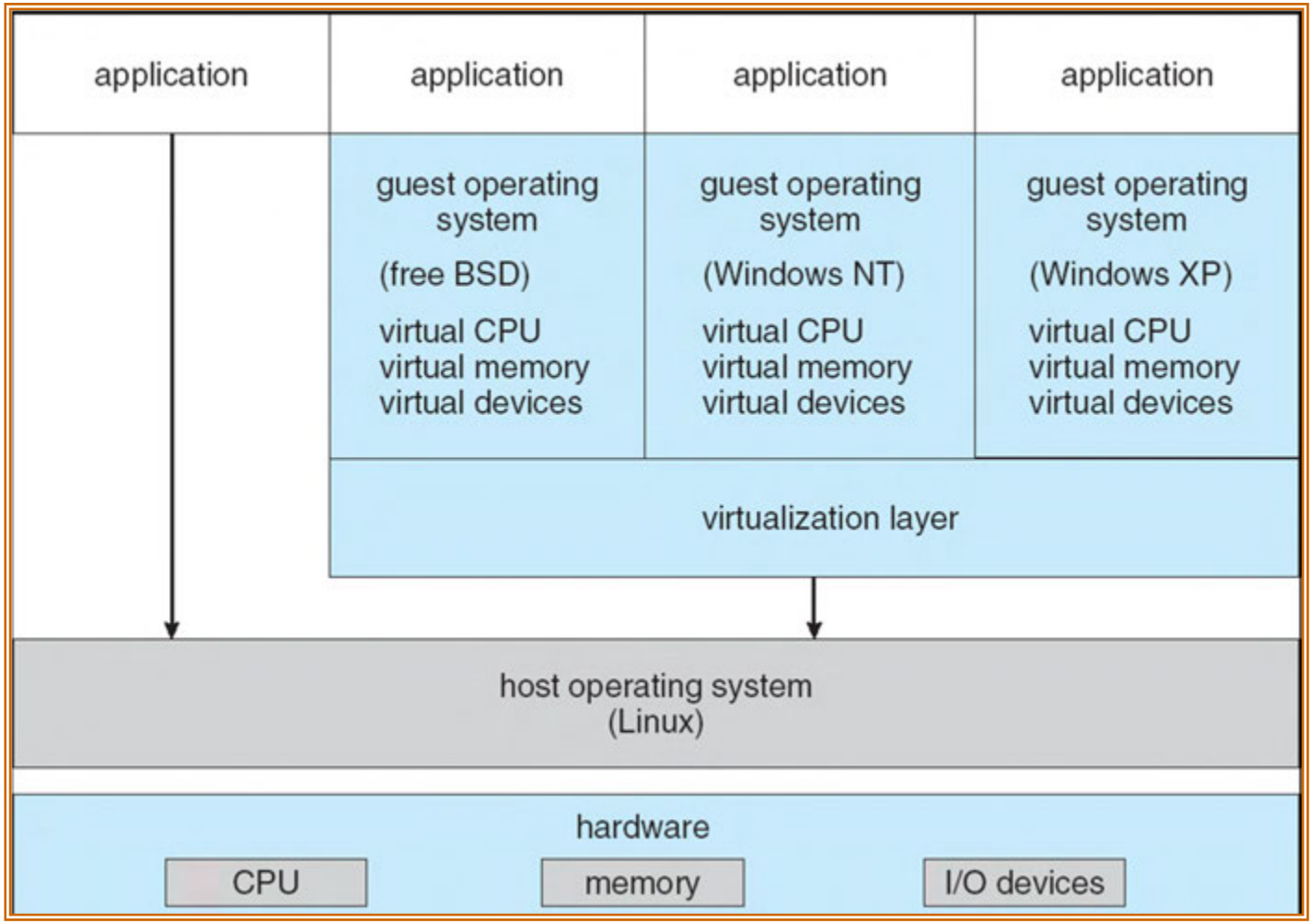
Virtual Machines (Cont.)

- Konsep virtual machine menyediakan perlindungan yang lengkap dari sumber daya sistem karena masing-masing mesin virtual terisolasi dari semua virtual machine lainnya. Isolasi ini, bagaimanapun, tidak mengizinkan berbagi sumber daya secara langsung .

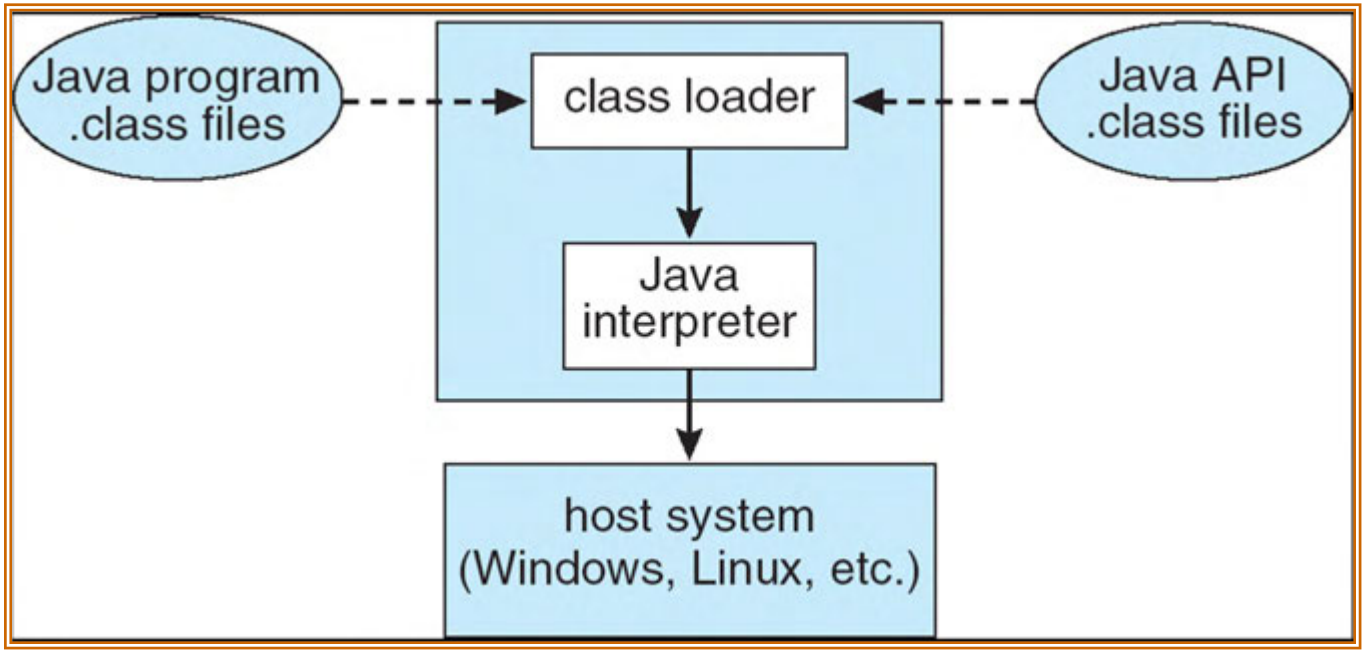
- Sebuah sistem virtual machine adalah kendaraan yang sempurna untuk penelitian sistem operasi dan development. Pengembangan sistem dilakukan pada mesin virtual, bukan pada mesin fisik sehingga tidak mengganggu operasi normal sistem.

- Konsep mesin virtual sangat sulit untuk diimplementasikan karena usaha yang dibutuhkan untuk menyediakan duplikat yang tepat untuk mesin yang mendasarinya

Arsitektur VMware



Java Virtual Machine



Operating System Generation

- Sistem operasi dirancang untuk berjalan di salah satu dari suatu kelas mesin, sistem harus dikonfigurasi untuk setiap situs komputer tertentu
- Program SYSGEN mendapatkan informasi mengenai konfigurasi khusus dari sistem perangkat keras
- ***Booting*** - memulai komputer dengan me-load kernel
- ***Bootstrap program*** - kode yang tersimpan dalam ROM yang mampu menemukan kernel, load ke memori, dan mulai eksekusi

Boot Sistem

- Sistem operasi harus dibuat tersedia untuk perangkat keras sehingga perangkat keras dapat memulainya
  - Potongan kecil kode - **bootstrap loader**, menempatkan kernel, memuat ke memori, dan memulainya
  - Kadang-kadang proses dua langkah dimana **boot blo**ck pada lokasi tetap memuat bootstrap loader
  - Ketika power diinisialisasi pada sistem, eksekusi dimulai pada lokasi memori yg tetap
    - Firmware digunakan untuk menyimpan kode boot awal (misal BIOS)