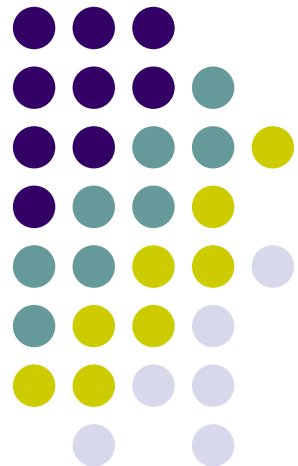
3

Struktur Sistem Operasi



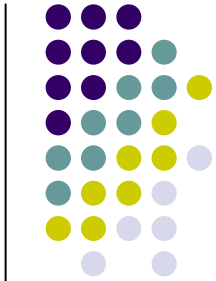
❋ Struktur Sistem Operasi

- Komponen Sistem
- Layanan Sistem Operasi
- System Calls
- Program System
- Struktur System
- Virtual Machines
- System Design dan Implementation
- System Generation

Komponen Sistem Umum

- Manajemen Proses
- Manajemen Main Memory
- Manajemen File
- Manajaemen I/O System
- Manajemen Secondary Storage
- Command-Interpreter System

Manajemen Proses



- Proses adalah sebuah program yang sedang dijalankan (eksekusi).
  - Suatu proses memerlukan resources pada saat ekesekusi: CPU time, memory, files dan I/O devices
- Sistim operasi bertanggung jawab terhadap aktifitas yang berhubungan dengan manajemen proses:
  - Process creation & deletion.
  - Process suspension (block) & resumption.
  - Mekanisme:
    - Sinkronisasi antar proses
    - Komunikasi antar proses

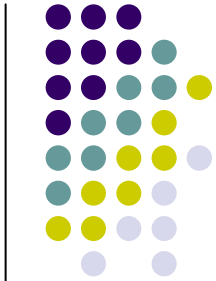
Manajemen Main-Memory

- Memori sebagai tempat penyimpanan instruksi/data dari program
  - Storage yang cepat sehingga dapat mengimbangi kecepatan eksekusi instruksi CPU
  - Terdiri dari “array of words/bytes” yang besar
  - Address digunakan untuk mengakses data (shared oleh CPU dan I/O devices)
- Umumnya main memory bersifat “volatile” – tidak permanent. Isinya akan hilang jika komputer di matikan.
- Manajemen memori:
  - Melacak pemakaian memori (siapa dan berapa besar?).
  - Memilih program mana yang akan diload ke memori.
  - Alokasi dan De-alokasi memori fisik untuk program.

Manajemen Secondary-Storage

- Secondary Storage: penyimpanan permanen
  - Kapasitas harus besar untuk menyimpan semua program dan data.
  - Secondary storage dapat dijadikan “backup” storage main memory supaya dapat enjalankan banyak program.
  - Umumnya menggunakan “magnetic disks” (hard disk).
- OS bertanggung jawab untuk manajemen disk:
  - Manajemen ruang kosong
  - Alokasi storage
  - Penjadualan disk

Manajemen I/O System

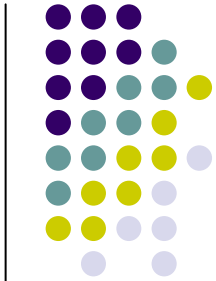


- Sering disebut device manager
  - Menyediakan “device driver” yang umum sehingga operasi I/O dapat seragam (open, read, write, close)
    - Contoh: user menggunakan operasi yang sama untuk read file pada hard disk, CD-ROM dan floppy disk sama.
- Komponen OS untuk sistim I/O:
  - Buffer: menampung sementarai data dari/ke I/O devices
  - Spooling: melakukan scheduling pemakaian I/O sistim supaya lebih efisien (antrian dsb)
  - Menyediakan “driver” untuk dapat melakukan operasi “rinci” (detail) untuk hardware I/O tertentu.

Manajemen File

- File: kumpulan informasi yang berhubungan (sesuai dengan tujuan pembuat file tsb).
  - File dapat mempunya struktur yang bersifat hirarkis (direktori, volume dll).
- OS bertanggung jawab:
  - Membuat dan menghapus file.
  - Membuat dan menghapus directory.
  - Dukungan primitif untuk manipulasi file dan directory.
  - Pemetaan file ke dalam secondary storage.
  - Backup file ke media storage yang stabil (nonvolatile).

Command-Interpreter System



- OS: menunggu instruksi dari user (command driven)
- Program yang membaca instruksi dan mengartikan keinginal user (lebih dari sejenis).
  - Contoh:
    - control-card interpreter
    - command-line interpreter
    - shell (in UNIX)
  - Sangat bervariasi dari satu OS ke OS yang lain dan disesuaikan dengan tujuan, teknologi I/O devices yang ada.
    - CLI, Windows, Pen-based (touch) etc.

❃ Layanan Sistem Operasi

- Eksekusi Program
  - Kemampuan sistim untuk “load” program ke memori dan menjalankan program.
- Operasi I/O
  - User tidak dapat secara langsung mengakses H/W resources, OS harus menyediakan mekanisme untuk melakukan operasi I/O atas nama user
- Manipulasi File-system
  - Kemampuan program untuk operasi pada file (to read, write, create, and delete files).

Layanan Sistem Operasi (Cont.)

- Komunikasi
  - Pertukaran data/informasi antar dua atau lebih proses yang berada pada satu komputer (atau lebih).
- Deteksi Error
  - Menjaga kestabilan sistim dengan mendeteksi “error”: hardware maupun operasi.
- Penggunaan System yang Efisien
  - Proteksi : menjamin akses ke system resources dikendalikan (user dikontrol akses ke sistim).
  - Accounting: merekam kegiatan users, jatah pemakaian resources (fairness atau policy).

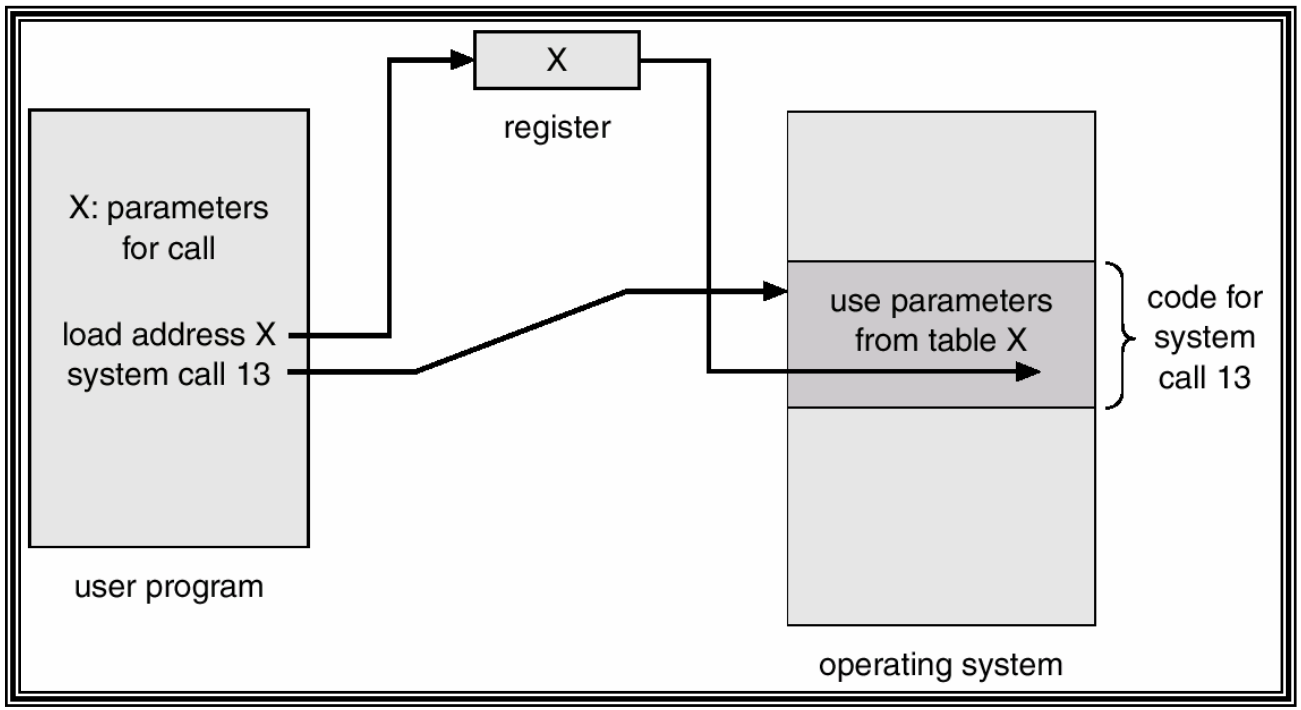
✵ System Call

- System call:
  - Menyediakan interface antara program (user program yang berjalan) dan bagian OS.
- System call menjadi jembatan antara proses dan OS.
  - System call ditulis dalam assembly language (machine specific) atau bahasa tingkat tinggi yang dapat mengendalikan mesin (C).
  - Contoh: UNIX menyediakan system call: read, write => operasi I/O untuk file.

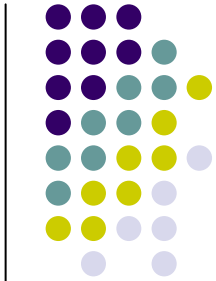
System Call : Passing Parameter

- Sering user program harus memberikan data (parameter) ke rutin OS yang akan dipanggil.
  - UNIX: read(buffer, max_size, file_id);
- Tiga cara memberikan parameter dari program ke sistim operasi:
  - Melalui registers (resources di CPU).
  - Menyimpan parameter pada data struktur (table) di memory, dan alamat table tsb ditunjuk oleh pointer yang disimpan di register.
  - Push (store) melalui “stack” pada memori dan OS mengambilnya melalui pop pada stack tsb.

Tabel Passing Parameter



Tipe System Calls

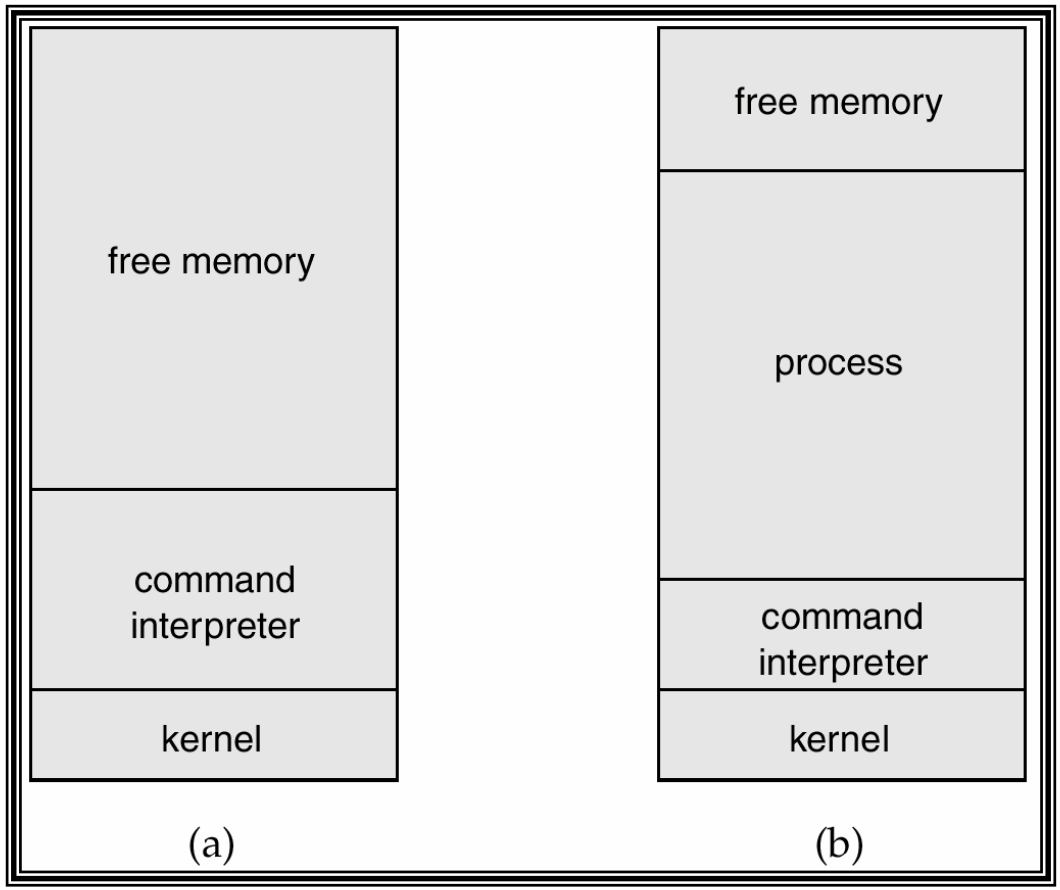


- Kontrol Proses
- Manipulasi File
- Managemen Device
- Informasi Lingkungan
- Komunikasi

Kontrol Proses

- Mengakhiri (end) dan membatalkan (abort)
- Mengambil (load) dan eksekusi (execute)
- Membuat dan mengakhiri proses
- Menentukan dan mengeset atribut proses
- Wait for time
- Wait event, signal event
- Mengalokasikan dan membebaskan memori

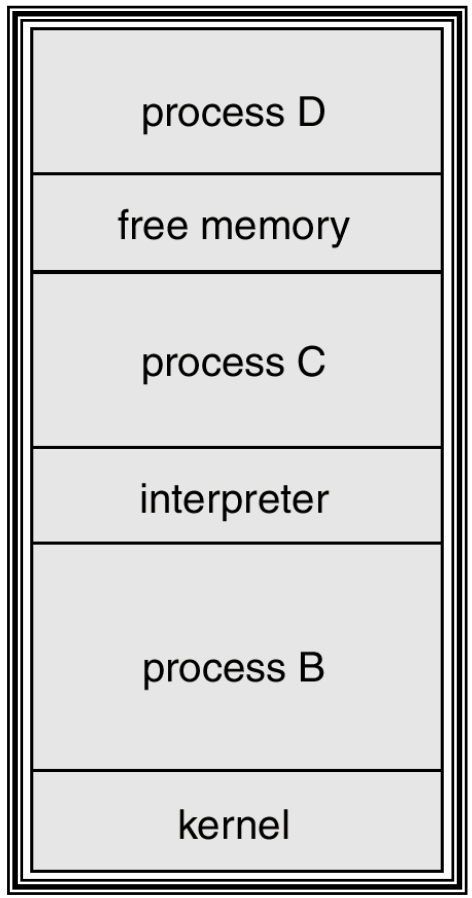
Eksekusi MS-DOS



At System Start-up

Running a Program

UNIX Menjalankan Multiple Program



Manipulasi File

- Membuat dan menghapus file
- Membuka dan menutup file
- Membaca, menulis dan mereposisi file
- Menentukan dan mengeset atribut file

Mangemen Device

- Meminta dan membebaskan device
- Membaca, menulis dan mereposisi device
- Menentukan dan mengeset atribut device

Informasi Lingkungan

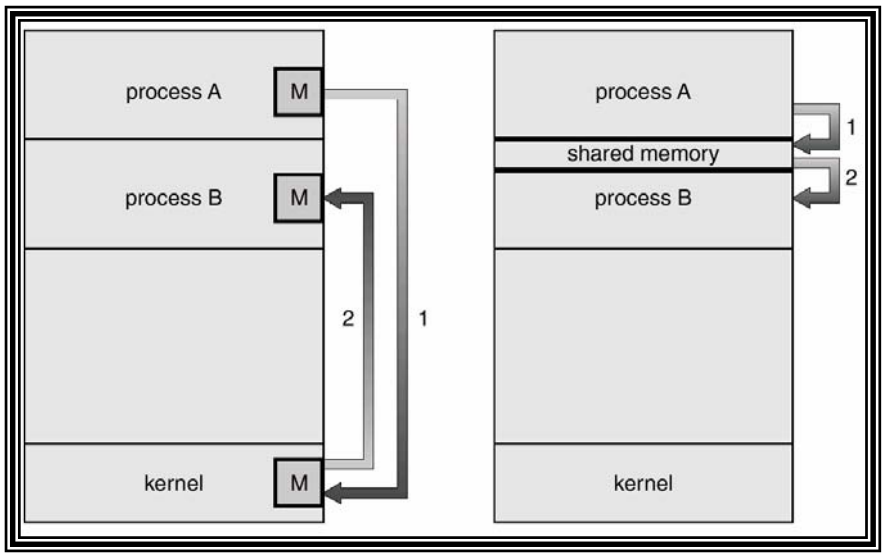
- Mengambil atau mengeset waktu atau tanggal
- Mengambil atau mengeset sistem data
- Mengambil atau mengeset proses, file atau atribut-atribut device

Komunikasi

- Membuat dan mengahpus sambungan komunikasi
- Mengirim dan menerima pesan
- Mentransfer status informasi

Komunikasi (2)

- Komunikasi dilakukan dengan melewatkan pesan atau sharing memori



Message Passing Shared Memory

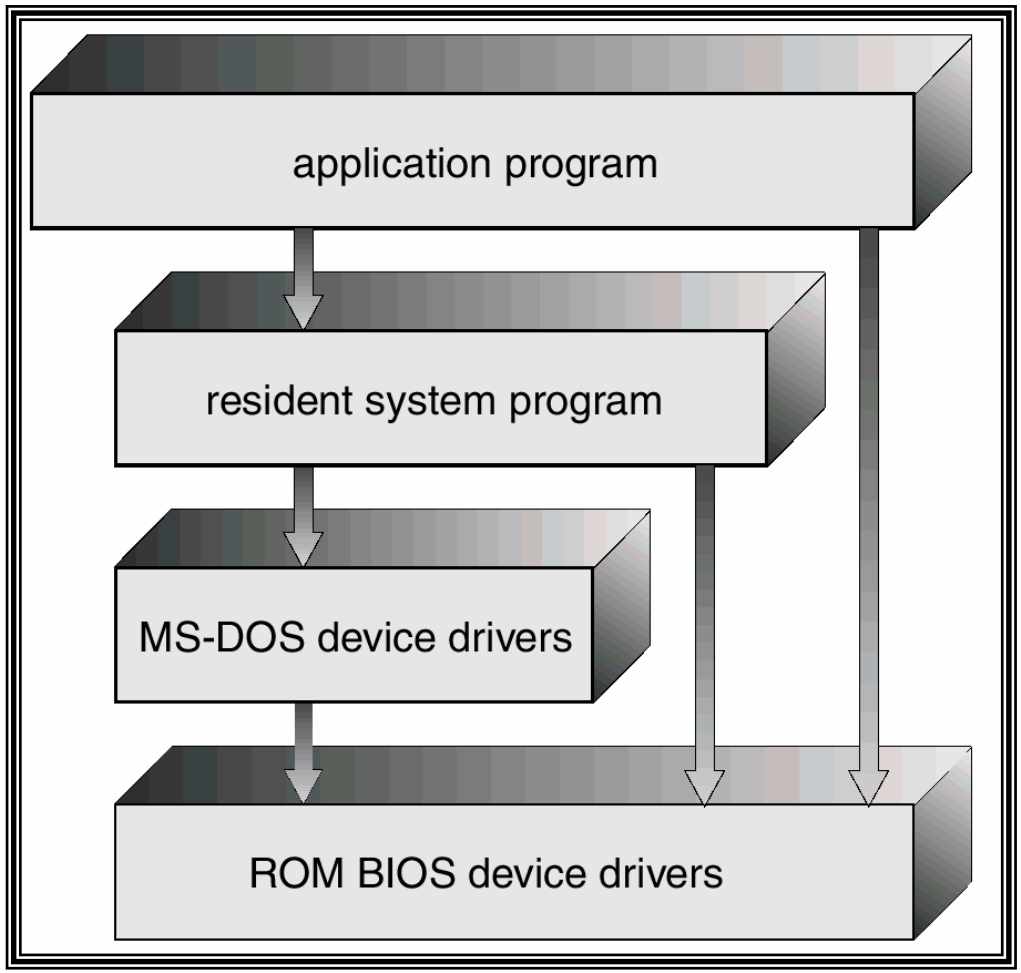
✵ Program Sistem

- Program sistem menyediakan kemudahan pembangunan program dan eksekusi.
  - Manipulasi File
  - Informasi status
  - Modifikasi File
  - Dukungan bahasa pemrograman
  - Loading dan eksekusi program
  - Komunikasi
  - Aplikasi program
- Kebanyakan user memandang sistem operasi sebagai program sistem, bukan sebagai “actual system calls”.

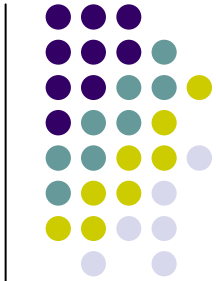
✵ Struktur Sistem Operasi

- Struktur Sistem Operasi
  - Metode untuk mengorganisasi dan membangun sistem operasi
- Contoh: MS-DOS
  - Saat dirancang kemampuan PC sangat minimal
    - Prosesor: 8086 (10 MHz), Max. memory: 640 Kb
  - MS-DOS – dibuat dengan menyediakan “fungsional” dari OS sebanyak mungkin pada resources yang sangat terbatas (memori)
  - Tidak dalam bentuk modul => monolithic (satu kesatuan):
    - MS-DOS menjadi satu kesatuan besar tanpa batasan jelas – fungsional dan interface
    - Terdapat struktur yang sangat sederhana dan “proteksi” yang longgar (single user system)

Struktur Layer MS-DOS



Struktur Monolithic

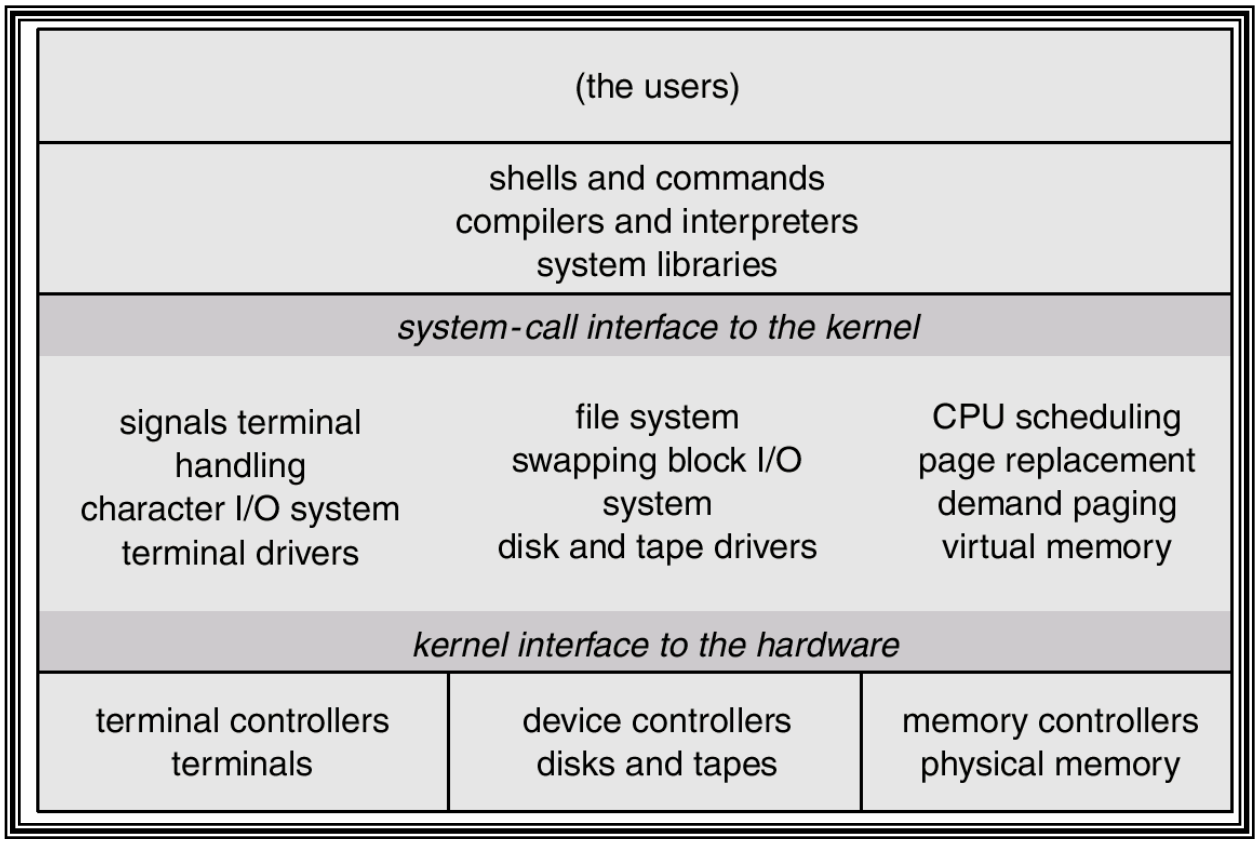


- Tidak terstuktur secara jelas
  - Kumpulan program yang menggunakan langsung resources hardware
  - Terdapat program-program yang mewakili fungsi OS: processor management, memory management
    - OS awal: satu kesatuan proses, dimana kontrol berpindah dari program-program tersebut (“procedure calls)
    - Program user (proses): menjadi satu bagian rutin dari (loop) program utama jika tidak melakukan fungsi OS
    - User program dijalankan “call” dari OS => eksekusi pada user mode akan berhenti:
      - timeout (timer interrupt)
      - kembali ke OS (service)
      - Interrupt (hardware)

Pendekatan Sederhana (Kernel)

- Struktur terbatas pada dua layer
  - Systems programs: bagian OS yang dibangun di atas kernel – extended machine
  - Kernel
    - Operasi vital yang penting dan melindungi resources hardware
    - Semua service untuk user proses melalui mekanisme system call
    - Tugas utama kernel menyediakan fasilitas untuk: multiprogramming/multitasking – dimana proses-proses dapat berjalan serentak (concurrent) dan terpisah
    - UNIX (1978)
      - Menggunakan pendekatan rancangan sederhana dengan dukungan H/W yang terbatas (PDP-11)

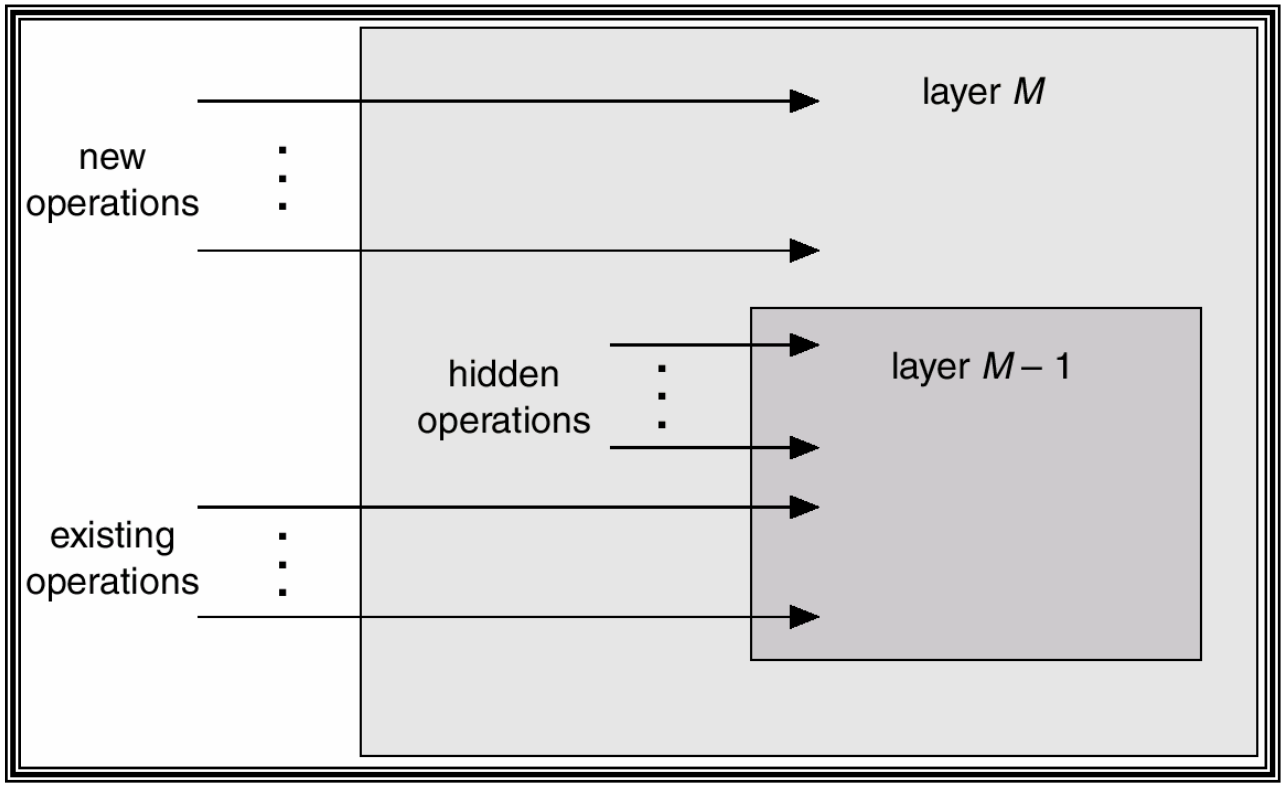
Struktur Sistem UNIX



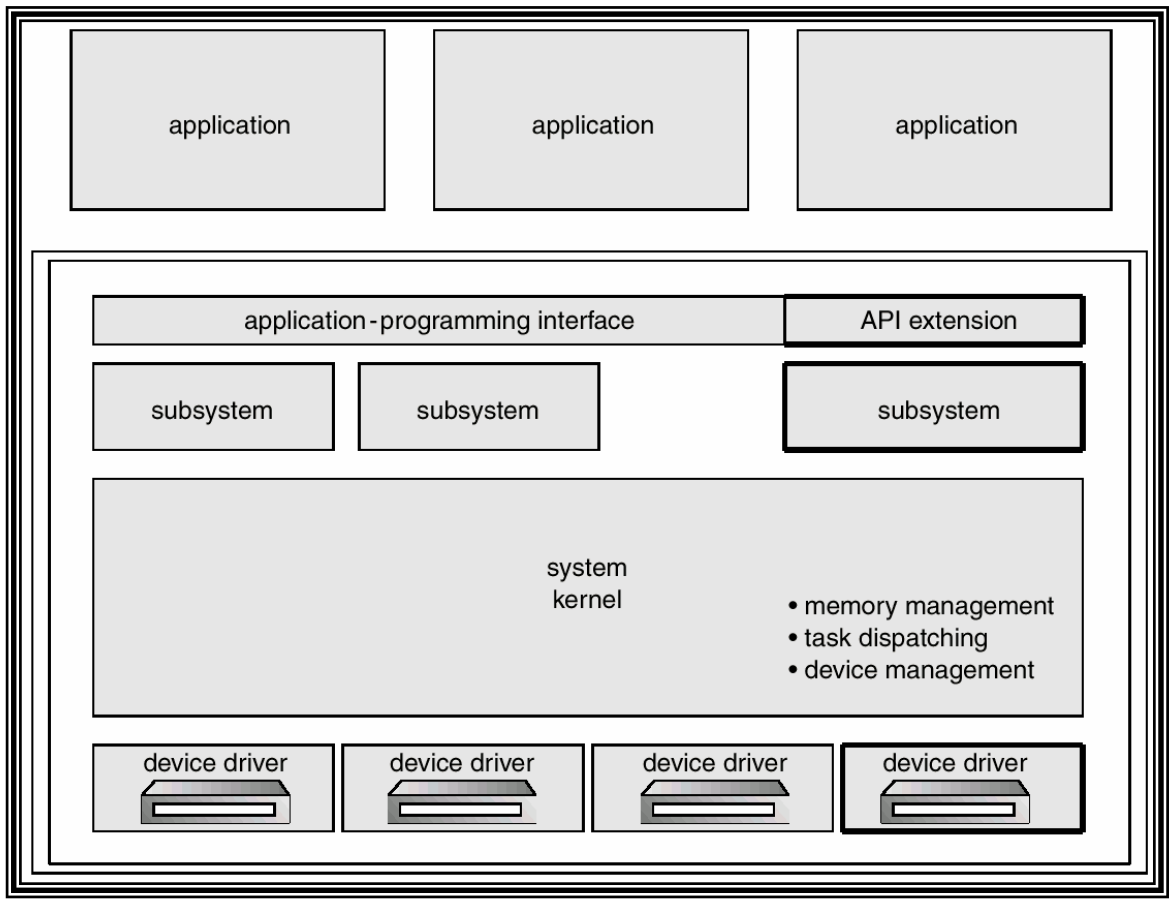
Pendekatan Berlapis (layer approach)

- Susunan berlapis:
  - OS dibagi atas sejumlah (lebih dari 2) layer
  - Setiap layer melingkupi layer di bawahnya (kendali, akses)
  - Layer paling bawah (0) => hardware
  - Layer paling atas (N) => user interface
- Rancangan moduler:
  - Layer disusun sehingga setiap fungsi/operasi layer atas akan menggunakan “services” pada layer bawah.

Sistem Operasi Berlapis



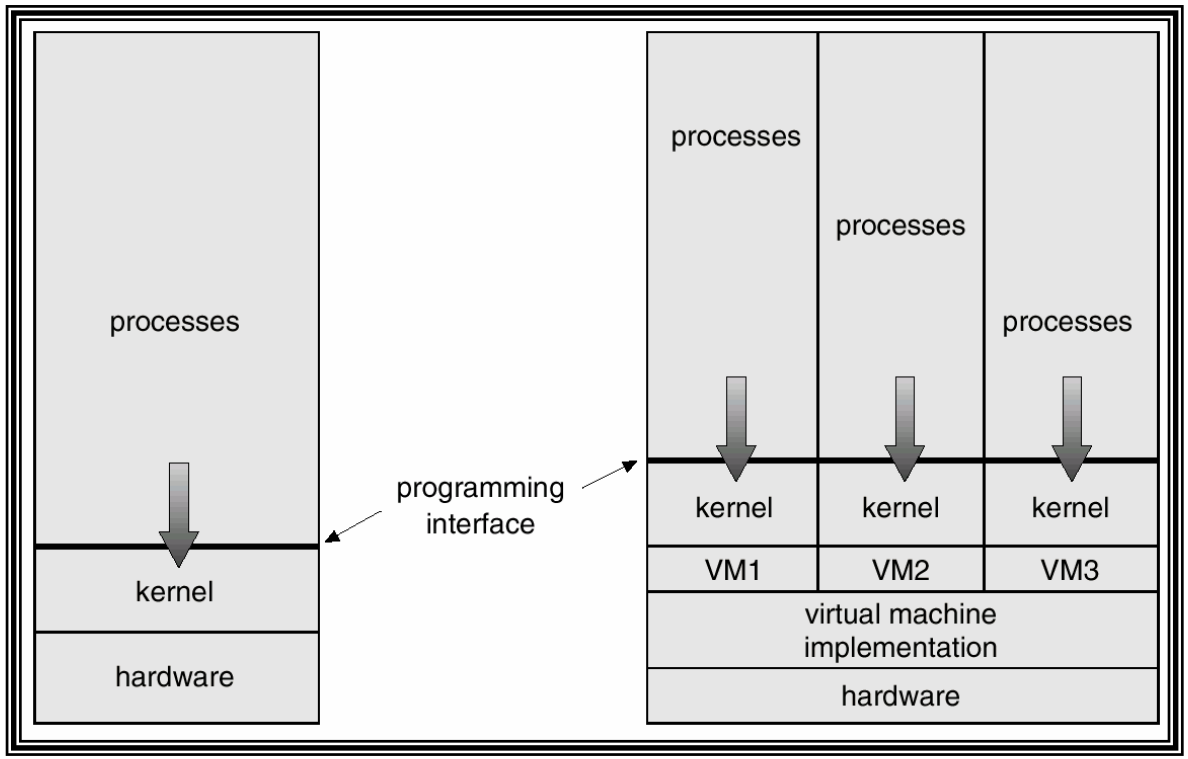
Struktur Berlapis OS/2



Virtual Machines

- Misalkan terdapat system program => control program yang mengatur pemakaian resources hardware.
- Control program = trap system call + hardware acces.
- Control program memberikan fasilitas ke proses user
  - Mendapatkan jatah CPU dan memori.
  - Menyediakan interface “identik” dengan apa yang disediakan oleh hardware => sharing devices untuk berbagai proses.
- Virtual machine => control program yang minimal
  - VM memberikan ilusi multitasking: seolah-olah terdapat prosesor dan memori ekslusif digunakan (virtual machine).
  - VM memilah fungsi multitasking dan implementasi extended machine (tergantung user proses) => flexible dan lebih mudah untuk maintained (proteksi).

Model Sistem VM



Virtual Machines (Cont.)

- Jika setiap user diberikan satu virtual machine =>user bebas untuk menjalankan OS (kernel) yang diinginkan pada VM tersebut.
  - Potensi lebih dari satu OS dalam satu komputer.
  - Contoh:
    - IBM VM370: menyediakan VM untuk berbagai OS: CMS (interaktif), MVS, CICS, dll.
  - Problem:
    - Sharing disk => setiap OS mempunyai file system yang mungkin berbeda.
    - IBM: virtual disk (minidisk) yang dialokasikan untuk user melalui VM.

Java Virtual Machine

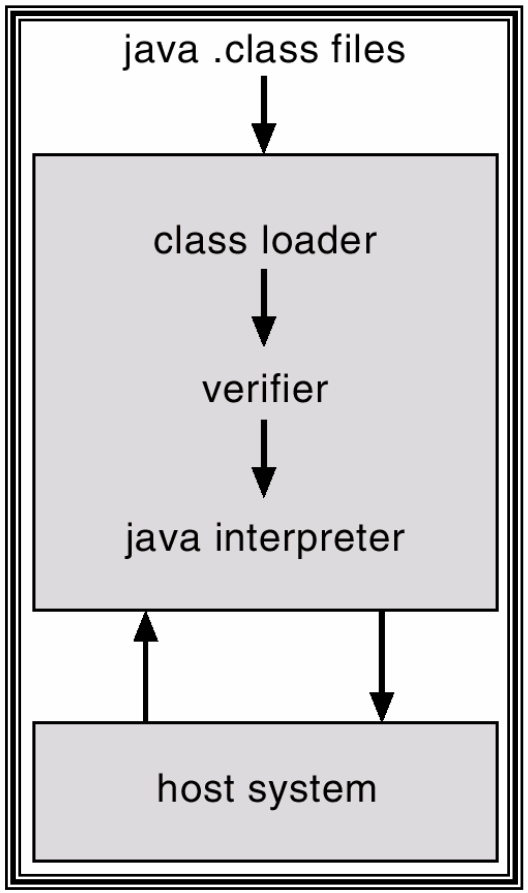
- Program Jaca dikompile pada platform-neutral bytecodes yang dieksekusi oleh Java Virtual Machine (JVM)
- JVM terdiri dari :

  - class loader

  - class verifier

  - runtime interpreter
- Kompiler Just-In-Time (JIT) meningkatkan kinerja

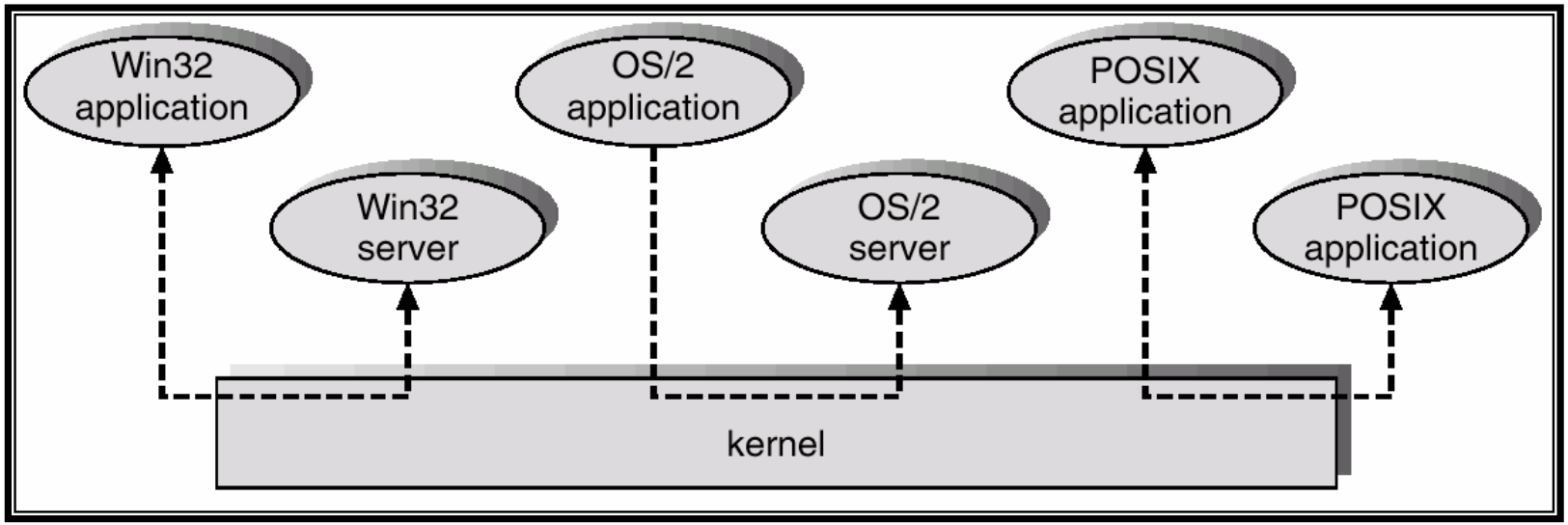
Java Virtual Machine



Model Client-Server

- Implementasi fungsi OS (extended machine) dapat menjadi bagian dari user proses (client)
  - Request service ke server proses (kernel).
  - Kernel: terdiri dari server (file, memory, I/O dll) yang melayani request dari client proses.
  - Akses ke hardware harus melalui server proses dari kernel yang mengontrol hardware tersebut.
- Proses : partisi dan struktur yang sangat jelas (interface dan fungsional).
- Konsekuensi : lambat (komunikasi antar client dan server), tidak efisien dalam menggunakan resources.

Windows NT Client-Server Structure



❃ Perancangan Sistem

Tujuan Perancangan Sistem

- Tujuan User – sistem operasi mudah digunakan, mudah dipelajari, handal, aman dan cepat
- Tujuan Sistem – sistem operasi mudah didisain, diimplementasikan dan dikelola, memiliki fleksibilitas, handal, bebas kesalahan dan efisien

❃ Implementasi Sistem

- Secara tradisional ditulis dalam bahasa assemby, saat ini sistem operasi dapat ditulis dalam bahasa tingkat tinggi.
- Kode yang ditulis dalam bahasa tingkat tinggi :
- Code written in a high-level language:
  - Dapat ditulis secara cepat.
  - Lebih compact.
  - Lebih mudah dipahami dan debug.
- Sistem operasi jauh lebih mudah untuk di port (dipindahkan ke hardware lain) jika ditulis dalam bahasa tingkat tinggi)

❃ System Generation (SYSGEN)

- Sistem operasi didisain untuk dijalankan pada berbagai kelas mesin. Sistem harus dikonfirgurasi untuk setiap komputer secara spesifik
- Program SYSGEN memiliki informasi dalam mengkonfigurasi sistem hardware secara spesifik
- *Booting* –awal komputer diaktifkan dengan melakukan loading kernel.
- *Bootstrap program* – kode yang disimpan di ROM yang dapat ditempatkan pada kernel, di load ke memori dan memulai eksekusi.