Bab 2

Struktur Sistem Operasi

POKOK BAHASAN:

- ✓ Komponen Sistem Operasi
- ✓ Layanan Sistem Operasi
- ✓ Sistem Call
- ✓ Sistem Program
- ✓ Struktur Sistem Operasi
- ✓ Mesin Virtual

TUJUAN BELAJAR:

Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu:

- Memahami komponen pada sistem operasi, manajemen yang diatur system operasi dan layanan pada sistem operasi
- Mengetahui beberapa struktur system operasi

2.1 KOMPONEN SISTEM

Sistem operasi terdiri dari beberapa komponen, antara lain manajemen proses, manajemen memori utama, manajemen file, manajemen sistem I/O, manajemen penyimpan sekunder, system jaringan, system proteksi dan system *command interpreter*.

2.1.1 Manajemen Proses

Proses adalah program yang sedang dieksekusi. Sebuah proses memerlukan sumber daya (*resource*) tertentu seperti waktu CPU, memori, file dan perangkat I/O untuk menyelesaikan tugasnya.

Untuk mengatur proses yang ada, sistem operasi bertanggung jawab pada aktrifitas-aktifitas yang berhubungan denagn manajemen proses berikut :

a. Pembuatan dan penghapusan proses yang dibuat oleh user atau sistem.
b. Menghentikan proses sementara dan melanjutkan proses.
c. Menyediakan kelengkapan mekanisme untuk sinkronisasi proses dan komunikasi proses.

2.1.2 Manajemen Memori Utama

Memori utama atau biasanya disebut dengan memori adalah sebuah array besar berukuran word atau byte, dimana setiap array tersebut mempunyai alamat tertentu. Memori adalah penyimpan yang dapat mengakses data dengan cepat yang digunakan oleh CPU dan perangkat I/O. Memori adalah perangkat penyimpan *volatile*. Isi memori akan hilang apabila terjadi kegagalan system.

Untuk mengatur memori, sistem operasi bertanggung jawab pada aktifitas-aktifitas manajemen memori sebagai berikut :

a. Menjaga dan memelihara bagian-bagian memori yang sedang digunakan dan dari yang menggunakan.
b. Memutuskan proses-proses mana saja yang harus dipanggil ke memori jika tersedia ruang di memori.
c. Mengalokasikan dan mendealokasikan ruang memori jika diperlukan.

2.1.3 Manajemen File

File adalah kumpulan informasi yang saling berhubungan yang sudah didefinisikan oleh pembuatnya (user). Biasanya, file berupa program (baik dalam bentu *source* maupun *object*) dan data.

Untuk mengatur file, sistem operasi bertanggung jawab pada aktifitas-aktifitas yang berhubungan dengan manajemen file sebagai berikut:

a. Pembuatan dan penghapusan file.
b. Pembuatan dan penghapusan direktori.
c. Primitif-primitif yang mendukung untuk manipulasi file dan direktori.
d. Pemetaan file ke memori sekunder.
e. Backup file ke media penyimpanan yang stabil (*nonvolatile*).

2.1.4 Manajemen I/O

Sistem operasi bertanggung-jawab pada aktifitas-aktifitas sistem I/O sebagai berikut:

a. Sistem *buffer-caching*.
b. Antarmuka *device-driver* secara umum.
c. Driver untuk *device hardware-hardware* tertentu.

2.1.5 Manajemen Penyimpan Sekunder

Karena memori utama (*primary storage*) bersifat *volatile* dan terlalu kecil untuk mengakomodase semua data dan program secara permanen, sistem komputer harus menyediakan penyimpan sekunder (*secondary storage*) untuk *back up* memori utama. Beberapa sistem komputer modern menggunakan disk untuk media penyimpan on-lin, baik program maupun data.

Sistem operasi bertanggung jawab pada aktifitas-aktifitas manajemen penyimpan sekunder sebagai berikut:

a. Pengaturan ruang bebas.
b. Alokasi penyimpanan.
c. Penjadwalan disk.

2.1.6 Sistem Jaringan (Sistem Terdistribusi)

Sistem terdistribusi adalah kumpulan prosessor yang tidak menggunakan memori atau clock bersama-sama. Setiap prosessor mempunyai local memori sendiri. Prosessor-prosessor pada sistem dihubungkan melalui jaringan komunikasi. Komunikasi dilakukan dengan menggunakan *protocol*.

Sistem terdistribusi memungkinkan user untuk mengakses sumber daya (*resource*) yang beragam. Dengan mengakses sumber daya yang dapat digunakan bersama-sama tersebut akan memberikan keuntungan dalam :

- Meningkatkan kecepatan komputasi
- Meningkatkan ketersediaan data
- Meningkatkan kehandalan sistem

2.1.7 Sistem Proteksi

Proteksi adalah suatu mekanisme untuk mengontrol akses oleh program, proses atau user pada sistem maupun *resource* dari user.

Mekanisme sistem proteksi yang harus disediakan sistem meliputi :

- Membedakan antara penggunaan yang sah dan yang tidak sah.
- Menentukan kontrol yang terganggu.
- Menetapkan cara pelaksanaan proteksi.

2.1.8 Sistem Command Intepreter

Beberapa perintah yang dimasukkan ke sistem operasi menggunakan pernyataan kontrol yang digunakan untuk

- Manajemen dan pembuatan proses
- Penangananan I/O
- Manajemen penyimpan sekunder
- Manajemen memori utama
- Akses sistem file
- Proteksi
- Jaringan

Program yang membaca dan menterjemakan pernyataan kontrol disebut dengan *command-line intepreter* atau shell pada UNIX. Fungsinya adalah untuk mengambil dan mengeksekusi pernyataan perintah berikutnya.

2.2 LAYANAN SISTEM OPERASI

Sistem operasi menyediakan layanan untuk programmer sehingga dapat melakukan pemrograman dengan mudah.

a. **Eksekusi Program.** Sistem harus dapat memanggil program ke memori dan menjalankannya. Program tersebut harus dapat mengakhiri eksekusinya dalam bentuk normal atau abnormal (indikasi error).
b. **Operasi-operasi I/O.** Pada saat running program kemungkinan dibutuhkan I/O, mungkin berupa file atau peralatan I/O. Agar efisien dan aman, maka user tidak boleh mengontrol I/O secara langsung, pengontrolan dilakukan oleh sistem operasi.
c. **Manipulasi sistem file.** Kapabilitas program untuk membaca, menulis, membuat dan menghapus file.
d. **Komunikasi.** Komunikasi dibutuhkan jika beberapa proses yang sedang dieksekusi saling tukar-menukar informasi. Penukaran informasi dapat dilakukan oleh beberapa proses dalam satu komputer atau dalam komputer yang berbeda melalui sistem jaringan. Komunikasi dilakukan dengan cara berbagi memori (*shared memory*) atau dengan cara pengiriman pesan (*message passing*).
e. **Mendeteksi kesalahan**. Sistem harus menjamin kebenaran dalam komputasi dengan melakukan pendeteksian error pada CPU dan memori, perangkat I/O atau pada user program.

Beberapa fungsi tambahan yang ada tidak digunakan untuk membantu user, tetapi lebih digunakan untuk menjamin operasi sistem yang efisien, yaitu :

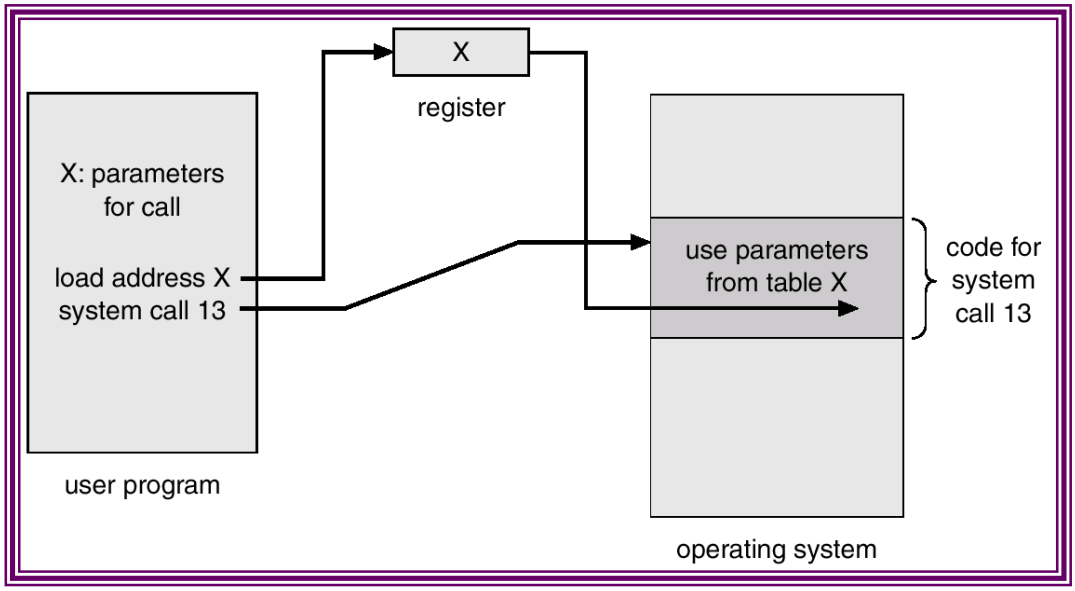
- **Mengalokasikan sumber daya (*resource*).** Sistem harus dapat mengalokasikan *resource* untuk banyak user atau banyak job yang dijalanan dalam waktu yang sama.
- **Akutansi.** Sistem membuat catatan daftar berapa *resource* yang digunakan user dan *resource* apa saja yang digunakan untuk menghitung secara statistik akumulasi penggunaan *resource*.
- **Proteksi.** Sistem operasi harus menjamin bahwa semua akses ke *resource* terkontrol dengan baik.

2.3 SISTEM CALL

System call menyediakan antar muka antara program yang sedang berjalan dengan sistem operasi. *System call* biasanya tersedia dalam bentuk instruksi bahasa assembly. Pada saat ini banyak bahasa pemrograman yang digunakan untuk menggantikan bahasa assembly sebagai bahasa pemrograman sehingga sistem call dapat langsung dibuat pada bahasa tinggat tinggi seperti bahasa C dan C++.

Terdapat 3 (tiga) metode yang umum digunakan untuk melewatkan parameter antara program yang sedang berjalan dengan sistem opeasi yaitu :

- Melewatkan parameter melalui *register*.
- Menyimpan parameter pada tabel yang disimpan di memori dan alamat tabel tersebut dilewatkan sebagai parameter di register seperti Gambar 2-1.
- *Push* (menyimpan) parameter ke *stack* oleh program dan *pop* (mengambil) isi *stack* yang dilakukan oleh system operasi.



Gambar 2-1 : Melewatkan parameter melalui tabel

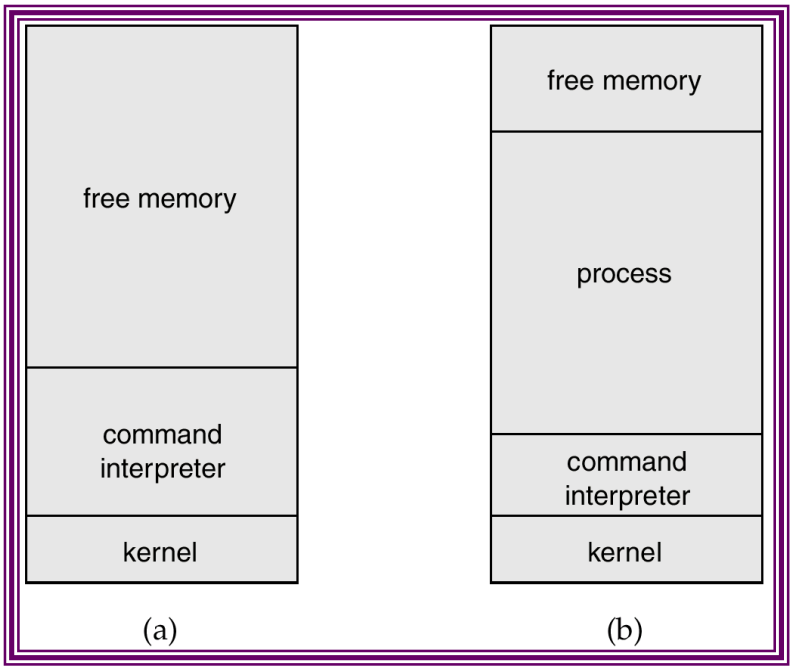
Pada dasarnya *System call* dapat dikelompokkan dalam 5 kategori seperti yang dijelaskan pada sub bab di bawah ini.

2.3.1. Kontrol Proses

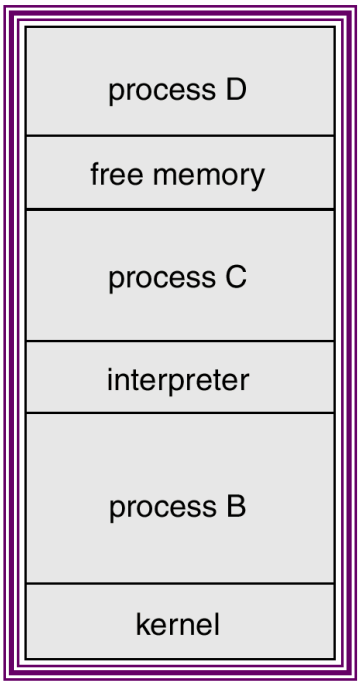
Hal-hal yang dilakukan:

- 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.

Contoh: Sistem operasi pada MS-DOS menggunakan sistem *singletasking* yang memeiliki *command interpreter* yang akan bekerja pada saat *start* (Gambar 2-2). Karena *singletasking*, maka akan menggunakan metode yang sederhana untuk menjalankan program dan tidak akan membuat proses baru. Sistem operasi UNIX dapat menjalankan banyak program (Gambar 2-3).



Gambar 2-2 : Sistem MSDOS : (a) pada saat startup (b) pada saat running



Gambar 2-3 : UNIX menjalankan lebih dari satu proses

2.3.2. Manipulasi File

Hal-hal yang dilakukan:

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

2.3.3. Manipulasi Device

Hal-hal yang dilakukan:

- Meminta dan mmebebaskan *device;*
- Membaca, menulis, dan mereposisi file;
- Menentukan dan mengeset atribut *device;*

2.3.4. Informasi Lingkungan

Hal-hal yang dilakukan:

- Mengambil atau mengeset waktu atau tanggal;
- Mengambil atau mengeset sistem data;

- Mengambil atau mengeset proses, file atau atribut-atribut *device;*

2.3.5. Komunikasi

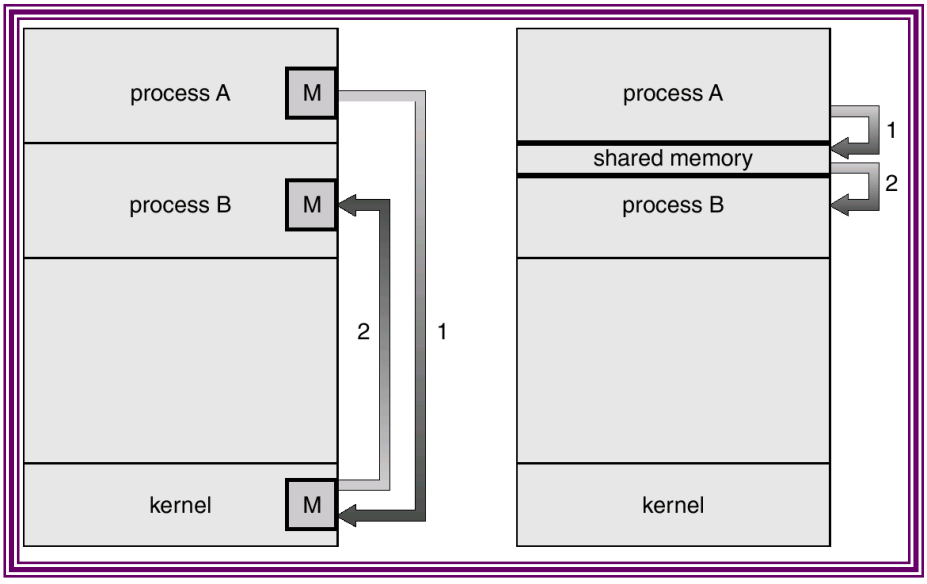
Hal-hal yang dilakukan:

- Membuat dan menghapus sambungan komunikasi;
- Mengirim dan menerima pesan;
- Mentransfer satus informasi;

Ada 2 model komunikasi:

a *Message-passing model*. Informasi saling ditukarkan melalui fasilitas yang telah ditentukan oleh sistem operasi (Gambar 2-4a).

b. *Shared-memory Model*. Proses-proses menggunakan map memory untuk mengakses daerah-daerah di memori dengan proses-proses yang lain (Gambar 2-4b).



Gambar 2-4 : Model komunikasi : (a) Message Passing; (b) Shared Memory

2.4 SISTEM PROGRAM

System program menyediakan lingkungan yang nyaman untuk pengembangan dan eksekusi program. Kebanyakan user melihat system operasi yang didefinisikan oleh *system program* dan bukan *system call* sebenarnya. System program adalah masalah yang relatif kompleks, namun dapat dibagi menajdi beberapa kategori, antara lain:

a. *Manipulasi File*. Meliputi: membuat, menghapus, mengcopy, rename, print, dump, list pada file dan direktori.
b. *Status Informasi*. Meliputi: tanggal, waktu (jam, menit, detik), penggunaan memori atau disk space, banyaknya user.
c. *Modifikasi File*. Ada beberapa editor yang sanggup digunakan sebagai sarana untuk menulis atau memodifikasi file yang tersimpan dalam disk atau tape.
d. *Bahasa Pemrograman yang mendukung*. Meliputi: Compiler, assambler, dan interpreter untuk beberapa bahasa pemrograman (seperti: Fortran, Cobol, Pascal, Basic, C, dan LISP).
e. *Pemanggilan dan Eksekusi Program*. Pada saat program dicompile, maka harus dipanggil ke memori untuk dieksekusi. Suatu sistem biasanya memiliki absolute *loader*, melokasikan *loader, linkage editor*, dan *overlay loader*. Juga dibutuhkan debugging sistem untuk bahasa tingkat tinggi.
f. *Komunikasi*. Sebagai mekanisme untuk membuat hubungan virtual antar proses, *user*, dan sistem komputer yang berbeda.
g. *Program-program aplikasi*. Sistem operasi harus menyokong program-program yang berguna untuk menyelesaikan permasalahan secara umum, atau membentuk operasi-operasi secara umum, seperti kompiler, pemformat teks, paket plot, sistem basis data, spreadsheet, paket analisis statistik, dan *games*.

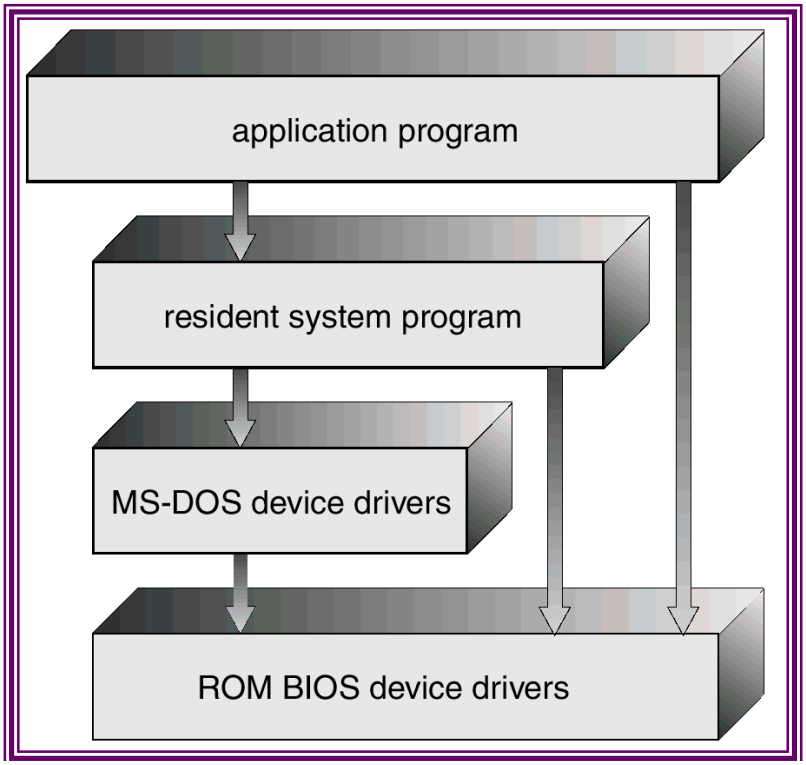
2.5 STRUKTUR SISTEM OPERASI

Sistem komputer modern yang semakin komplek dan rumit memerlukan sistem operasi yang dirancang dengan sangat hati-hati agar dapat berfungsi secara optimum dan mudah untuk dimodifikasi.

2.5.1 Struktur Sistem MS-DOS

Ada sejumlah sistem komersial yang tidak memiliki struktur yang cukup baik. Sistem operasi tersebut sangat kecil, sederhana dan memiliki banyak keterbatasan. Salah satu contoh sistem tersebut adalah MS-DOS. MS-DOS dirancang oleh orang-orang yang tidak memikirkan akan kepopuleran *software* tersebut. Sistem operasi tersebut terbatas pada perangkat keras sehingga tidak terbagi menjadi modul-modul. Meskipun MS-DOS

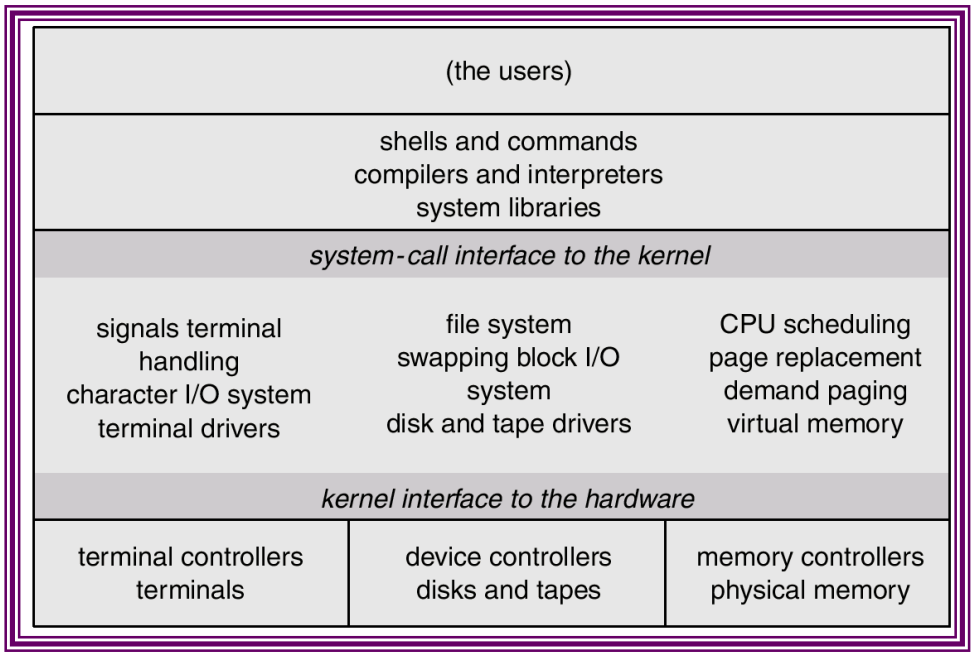
mempunyai beberapa struktur, antar muka dan tingkatan fungsionalitas tidak terpisah secara baik seperti pada Gambar 2-5. Karena Intel 8088 tidak menggunakan dual-mode sehingga tidak ada proteksi hardware. Oleh karena itu orang mulai enggan menggunakannya.



Gambar 2-5 : Struktur Layer pada MS-DOS

2.5.2 Struktur Sistem UNIX

Sistem operasi UNIX (Original UNIX) juga terbatas pada fungsi perangkat keras dan struktur yang terbatas. UNIX hanya terdiri atas 2 bagian, yaitu Kernel dan program sistem. Kernel berada di bawah tingkat antarmuka *system call* dan diatas perangkat lunak secara fisik. Kernel ini berisi sistem file, penjadwalan CPU, menejemen memori, dan fungsi sistem operasi lainnya yang ada pada sistem call berupa sejumlah fungsi yang besar pada satu level. Program sistem meminta bantuan kernel untuk memanggil fungsi-fungsi dalam kompilasi dan manipulasi file. Struktur system UNIX dapat dilihat pada Gambar 2-6.



Gambar 2-6 : Struktur sistem UNIX

2.5.3 Pendekatan Terlapis (*Layered Approach*)

Teknik pendekatan terlapis pada dasarnya dibuat dengan menggunakan pendekatan *top-down*, semua fungsi ditentukan dan dibagi menjadi komponen-komponen. Modularisasi sistem dilakukan dengan cara memecah sistem operasi menajdi beberapa lapis (tingkat). Lapisan terendah (layer 0) adalah perangkat keras dan lapisan teratas (layer N) adalah *user interface*. Dengan system modularisasi, setiap lapisan mempunyai fungsi (operasi) tertentu dan melayani lapisan yang lebih rendah. Gambar 2-7 menunjukkan system pendekatan terlapis tersebut.

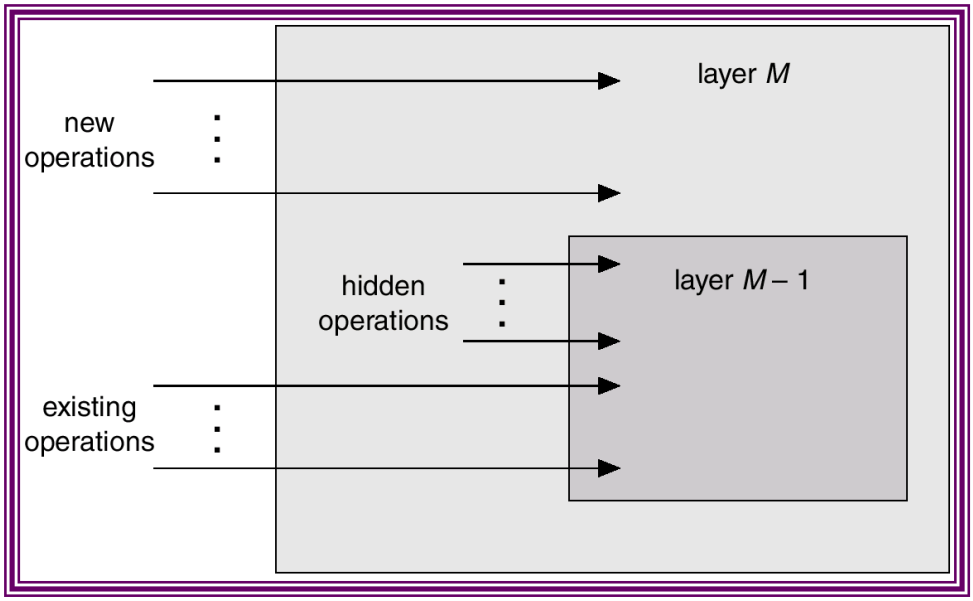
Contoh sistem operasi yang menggunakan sistem ini adalah: UNIX termodifikasi, THE, Venus dan OS/2 (Gambar 2-8). Lapisan pada struktur THE adalah:

Lapis-5 : user program
Lapis-4 : buffering untuk I/O device
Lapis-3 : operator-console *device driver*
Lapis-2 : menejemen memori
Lapis-1 : penjadwalan CPU
Lapis-0 : hardware

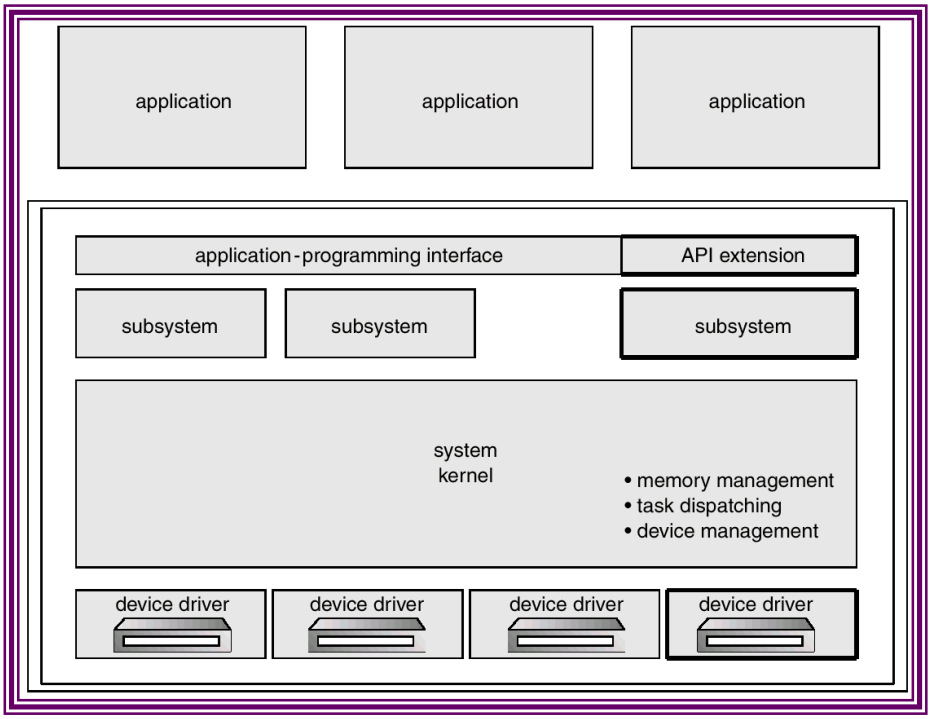
Sedangkan lapisan pada struktur Venus adalah :

Lapis-6 : user program
Lapis-5 : *device driver* dan sceduler

Lapis-4	:	virtual memory
Lapis-3	:	I/O channel
Lapis-2	:	penjadwalan CPU
Lapis-1	:	instruksi interpreter
Lapis-0	:	hardware



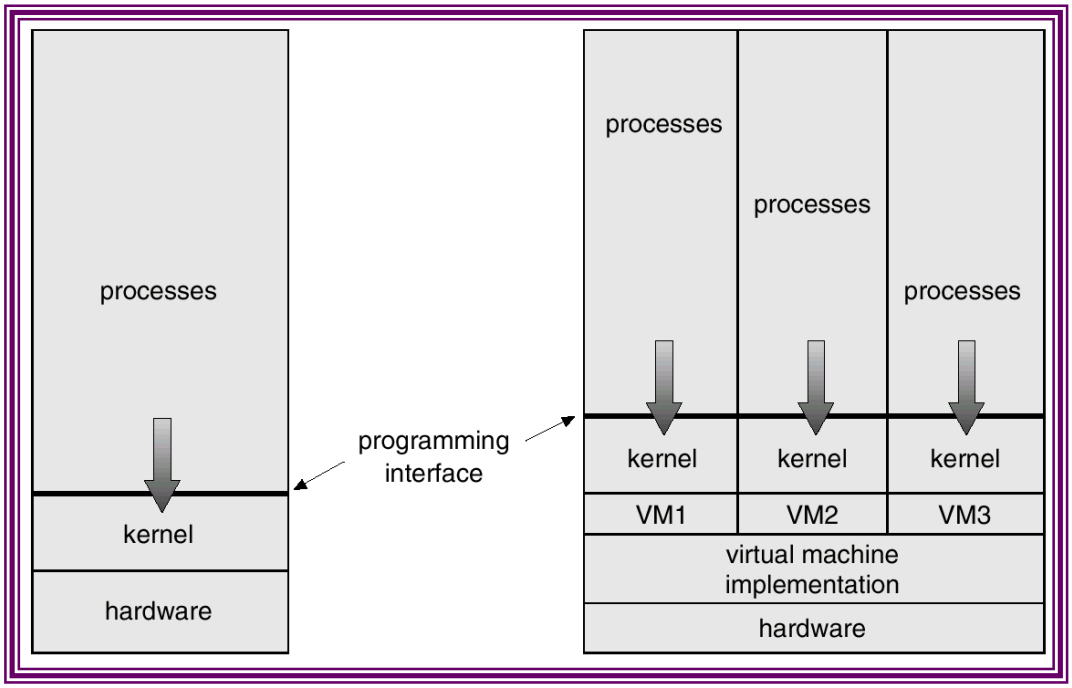
Gambar 2-7 : Struktur sistem terlapis



Gambar 2-8 : Struktur sistem OS/2

2.6 MESIN VIRTUAL

Konsep dasar dari mesin virtual ini tidak jauh berbeda dengan pendekatan terlapis, hanya saja konsep ini memberikan sedikit tambahan berupa antarmuka yang menghubungkan perangkat keras dengan kernel untuk tiap-tiap proses, Gambar 2-9 menunjukkan konsep tersebut. Mesin virtual menyediakan antar muka yang identik untuk perangkat keras yang ada. Sistem operasi membuat ilusi untuk beberapa proses, masing-masing mengeksekusi prosessor masing-masing untuk memori (virtual) masing-masing.



Gambar 2-9: Struktur mesin virtual

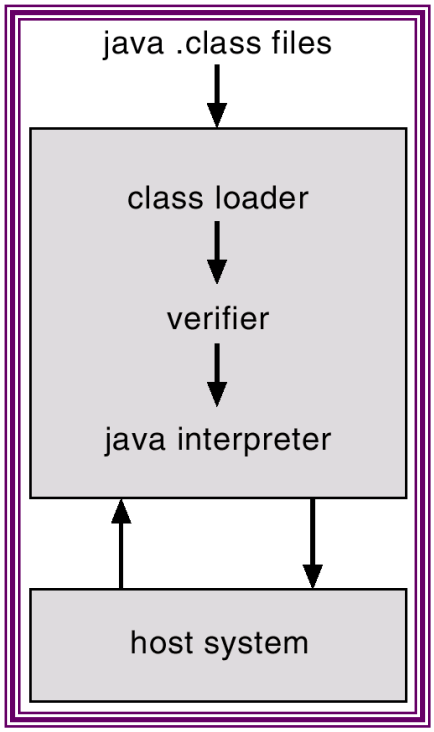
Meskipun konsep ini cukup baik, namun sulit untuk diimplementasikan, ingat bahwa sistem menggunakan metode *dual-mode*. Mesin virtual hanya dapat berjalan pada *monitor-mode* jika berupa sistem operasi, sedangkan mesin virtual itu sendiri berjalan dalam bentuk *user-mode*. Konsekuensinya, baik virtual *monitor-mode* maupun virtual *user-mode* harus dijalankan melalaui *physical user mode*. Hal ini menyebabkan adanya transfer dari *user-mode* ke *monitor-mode* pada mesin nyata, yang juga akan

menyebabkan adanya transfer dari virtual *user-mode* ke virtual *monitor-mode* pada mesin virtual.

Sumber daya (*resource*) dari computer fisik dibagi untuk membuat mesin virtual. Penjadwalan CPU dapat membuat penampilan bahwa user mempunyai prosessor sendiri. *Spooling* dan system file dapat menyediakan *card reader* virtual dan *line printer* virtual. Terminal time sharing pada user melayani sebagai *console* operator mesin virtual. Contoh sistem operasi yang memakai mesin virtual adalah IBM VM system.

Keuntungan dan kerugian konsep mesin virtual adalah sebagai berikut :

- Konsep mesin virtual menyediakan proteksi yang lengkap untuk sumber daya system sehingga masing-masing mesin virtual dipisahkan mesin virtual yang lain. Isolasi ini tidak memperbolehkan pembagian sumber daya secara langsung
- Sistem mesin virtual adalah mesin yang sempurna untuk riset dan pengembangan system operasi. Pengembangan system dikerjakan pada mesin virtual, termasuk di dalamnya mesin fisik dan tidak mengganggu operasi system yang normal.
- Konsep mesin virtual sangat sulit untuk mengimplementasikan kebutuhan dan duplikasi yang tepat pada mesin yang sebenarnya.



Gambar 2-10 : Java Virtual Machine

Java merupakan system yang menggunakan implementasi mesin virtual. Untuk mengkompilasi program Java maka digunakan kode bit yang disebut *platform-neutral bytecode* yang dieksekusi oleh Java *Virtual Machine* (JVM). JVM terdiri dari class loader, class verifier dan runtime interpreter seperti pada Gambar 2-10.

LATIHAN SOAL :

1. Aktifitas apa yang dilakukan sistem operasi berhubungan dengan :
   a. Manajemen proses
   b. Manajemen memory utama
   c. Manajemen file
2. Apa kegunaan sistem command interpreter ?
3. Apa yang dimaksud dengan system calls ? Sebutkan contohnya.
4. Apa yang dimaksud sistem program ?
5. Apa keuntungan dan kelemahan sistem layer ?
6. Apa keuntungan dan kerugian sistem virtual memory ?