MODUL PRAKTIKUM

SISTEM OPERASI



UNIVERSITAS TEKNOLOGI MATARAM

2019

Jl. Kampus Universitas Teknologi Mataram, Kekalik, Mataram

MODUL PERKULIAHAN

Sistem Operasi

1. Sekilas sistem komputer

Abstract

Komponen dasar sistem komputer adalah pemroses, memori utama, perangkat I/O dan interkoneksi. Komponen dan fungsi dan pemroses. Konsep level memori dan hirarkinya.

Kompetensi

Mampu menjelaskan garis besar perangkat komputer

Sekilas sistem komputer

Sebuah komputer akan terdiri dari elemen-elemen dasar seperti prosesor, memori, I/O dan beberapa modul yang melengkapi komponen tersebut. Masing-masing komponen tersebut akan saling terkoneksi sehingga fungsi utama dari komputer dapat berjalan untuk digunakan mengeksekusi program.

Elemen dasar sistem komputer

Untuk menjalankan komputer, minimal komputer mempunyai komponen dasar sistem komputer, sebagi berikut :

a. Prosesor : mempunyai fungsi untuk mengendalikan operasi komputer danmelakukan fungsi memproses data.
b. Memori utama : fungsi utama adalah tempat menyimpan data dan program. Sifat utama dari memori utama adalah volatile (tidak dapat mempertahan data dan program bila sumber daya listrik mati), memori utama juga disebut real memori atau memori primer.
c. Perangkat Masukan dan keluaran : mempunyai fungsi untuk memindahkan data antara komputer dan lingkungan eksternal. Lingkungan eksternal terdiri dari bermacam-macam perangkat eksternal seperti memori sekunder, terminal dan peralatan komunikasi.
d. Interkoneksi antar komponen : struktur dan mekanisme yang berfungsi untuk menghubungkan ketiga komponen di atas sehingga dapat melakukan komunikasi antara prosesor, memori utama dan modul I/O.

a. Presesor

Prosesor umumnya berada di dalam kontrol. Salah satu fungsinya adalah melakukan pertukaran data dengan memori. Bagian utama dari Prosesor adalah :

- ALU (aritmetic Logic Unit) untuk kompilasi (melakukan operasi aritmatika dan logika)
- CU (control unit) untuk pengendali
- Register – register membantu pelaksanaan operasi yang dilakukan pemroses.

Pemroses berfungsi untuk melakukan kegiatan seperti : menghitung, melakukan operasi logik dan mengelola aliran data; dengan cara membaca instruksi dari memori dan mengeksekusinya. Untuk melakukan eksekusi maka pemroses dituntun oleh clock.

Langkah – langkah pemroses dalam melakukan kerja :

- mengambil instruksi yang dikodekan secara biner dari memori utama
- men-dekode instruksi menjadi aksi sederhana
- melakukan aksi – aksi.

Type operasi yang sering dilakukan oleh komputer :

- operasi aritmatika (penambahan, pengurangan, peraklian dsb)
- operasi logika (or, and, Xor, invers dsb)
- operasi pengendalian (pencabangan, loop, lompat dsb)

Register

Dalam system computer terdapat dua kategori register :

a) register yang terlihat pemakai (User-visible register)
b) register untuk kendali status (control and status register)

a) Register yang terlihat pemakai

Untuk jenis register ini maka pemrogram dapat memeriksa isi dari register – register tersebut. Ada 2 jenis yang termasuk register ini, yaitu : register data dan register alamat.

Register data

Register data dapat di-assign ke beragam fungsi oleh pemrogram. Register yang digunakan untuk menyimpan suatu nilai untuk beragam keperluan. Jenis register data adalah : general purpose register dan special purpose register.

(a) General purpose register berfungsi untuk melakukan suatu operasi terhadap data pada beraneka ragam keperluan instruksi mesin.
(b) Special purpose register digunakan untuk keperluan tertentu, misalnya :
menampung operasi floating
menampung limpahan operasi penjumlahan dan perkalian, dsb

Register alamat

Register alamat berisi alamat data dan instruksi yang terdapat dalam memori utama. Register alamat dapat bersifat serbaguna atau untuk tujuan tertentu. Jadi register alamat adalah register yang berhubungan dengan :

alamat data di memori utama
alamat instruksi di memori utama
bagian alamat yang digunakan dalam perhitungan alamat lengkap

Contoh register alamat :

(a) Register indeks. Pengalamatan melibatkan penambahan indeks pada nilai dasar, merupakan mode pengalamatan yang popular.
(b) Register penunjuk segmen. Register penunjuk segmen mencata alamat dasar dari segmen. Memori dibagi menjadi segmen – segmen dengan panjang bervariasi. Sangat penting untuk manajemen memori.
(c) Register penunjuk stack. Stack merupakan mekanisme penting pada sistem komputer, biasanya diimplementasikan pada memori utama. Operasi – operasi terhadap stack, yaitu :
    instruksi push (untuk menyimpan data pada stack)
    instruksi pop (untuk mengambil data dari puncak)
(d) register penanda berisi kondisi – kondisi yang dihasilkan pemroses berkaitan dengan operasi yang baru saja dilakukan.

b) Register untuk kendali dan status.

Digunakan untuk mengendalikan operasi dari pemroses. Sebagian dapat diakses dengan instruksi mesin yang dieksekusi dalam mode kontrol atau kernel sistem operasi. Sedang pada sebagian mesin register tersebut bersifat tidak visible terhadap pengguna. Jenis dari register untuk kendali dan status yaitu :

(a) Register untuk alamat dan buffer terdiri atas :
    - MAR (memory address register) untuk mencatat alamat yang dapat diakses.
    - MBR (memory buffer register) untuk menampung data yang akan dituliskan ke memori yang alamatnya ditunjukan oleh MAR
    - I/O AR (I/O address register) untuk mencatat port I/O yang akan diakses
    - I/O BR (I/O buffer register) untuk menampung data yang akan dituliskan ke port yang alamatnya ditunjukkan oleh I/O AR.

(b) Register untuk eksekusi instruksi, terdiri atas :

- PC (program counter) untuk mencatat alamat memori dimana instruksi didalamnya akan dieksekusi
- IR (instruction register) untuk menampung instruksi yang akan dilaksanakan.

(c) Register untuk informasi status, dapat berupa satu register atau kumpulan register yang disebut PSW (program status word) berisi kode - kode kondisi pemroses ditambah statusnya. PSW biasanya berisi informasi :

- Sign untuk mencatat tanda yang dihasilkan operasi yang sebelumnya dijalankan.
- Zero mencatat apakah operasi sebelumnya menghasilkan nol
- Carry mencatat apakah dihasilkan carry (kondisi dimana operasi penjumlahan atau perkalian menghasilkan bawaan yang tidak dapat ditampung register akumulator)
- Equal mencatat apakah operasi menghasilkan kondisi sama dengan
- Overflow untuk mencatat apakah operasi menghasilkan kondisi overflow
- Interrupt enable/disable mencatat apakah interup sedang aktif atau tidak.
- Supervisor mencatat apakah mode eksekusi yang dilaksanakan mode supervisor atau bukan. Untuk mode supervisor seluruh instruksi dapat dilaksanakan, sedang kalau bukan maka beberapa instruksi kritis tidak dapat diaktifkan.

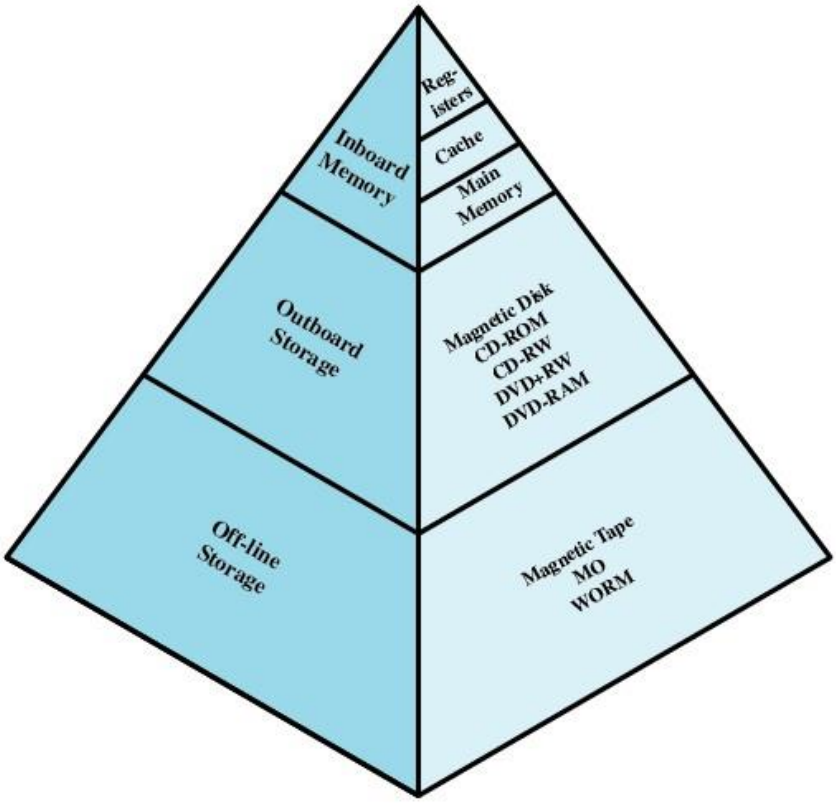
b. Memori utama

Masalah yang timbul dalam rancangan komputer yaitu seberapa besar, seberapa cepat dan seberapa mahal memori tersebut. Sehingga dari ketiga kondisi tersebut maka karakteristik yang penting dalam pemilihan memori adalah harga, kapasitas dan waktu akses. Dalam semua spektrum teknologi terdapat hubungan sebagi berikut :

- Semakin kecil waktu akses maka semakin mahal harga per bit
- Semakin besar kapasitas maka semakin murah harga per bit
- Semakin besar kapasitas maka semakin besar waktu akses

Memori utama berfungsi untuk menyimpan data dan program

Hirarki memori	Kecepatan akses	harga	kapasitas	Frekuensi pengaksesan
Register Chace memory Main memory Disk chace Magnetic disk Magnetic tape atau Optical disk	Tercepat ↓ Terlambat	Mahal ↓ Murah	Banyak ↓ Terbatas	Terbanyak ↓ Paling sedikit



Gambar 1. Hirarki memori

konsep memori dua level, data – data atau instruksi ditampung pada hirarki paling tinggi. Implementasi dari konsep ini yaitu :

a) Chace memory : kapasitas terbatas, kecepatan tinggi, letaknya diantara memori utama dan register pemroses, fungsi agar pemroses tidak langsung mengacu memori utama tetapi cukup di chace memory.
b) Buffering : bagian memori utama untuk menampung data yang akan ditransfer dari/ke perangkat masukan/keluaran dan penyimpan sekunder.

c. Perangkat masukan/keluaran

Perangkat masukan dan keluaran komputer secara garis besar terdiri dari 2 bagian :

a) Komponen mekanis (perangkat)
b) Komponen elektronis (pengendali perangkat berupa chip controller)

Perangkat dikendalikan oleh chip controller di board sistem atau card, yang dihubungkan dengan pemroses atau komponen lainnya melalui bus. Setiap kontroler mempunyai register kendali, register status (kode kesalahan). Setiap controller dapat dialamati oleh pemroses. Antarmuka perangkat pengendali (device adapter) mengikuti standar ANSI, IEEE, ISO, CCITT, EIA atau standar de-facto.

d. Interkoneksi antar komponen

Interkoneksi antar komponen disebut bus, terdiri atas :

a) Bus alamat : yang bisa berisi 16, 20, 24 jalur sinyal paralel atau lebih. Bus alamat merupakan jalur yang digunakan oleh CPU untuk mengirim alamat lokasi ke memori/port. Jumlah memori dan/atau port yang dapat dialamati yaitu 2^N dimana N adalah jalur alamat.
b) Bus data : berisi 8, 16, 32 jalur sinyal paralel atau lebih. bus data merupakan jalur dua arah antara CPU dan memori/port.
c) Bus kendali : berisi 4 – 10 jalur sinyal paralel, merupakan jalur satu arah dari CPU ke memori/port. Bus kendali terdiri atas :
    c) memory read : memerintahkan melakukan pembacaan memori
    d) memory write : memerintahkan melakukan penulisan memori
    e) I/O read : memerintahkan melakukan pembacaan port I/O
    f) I/O write : memerintahkan melakukan penulisan port I/O

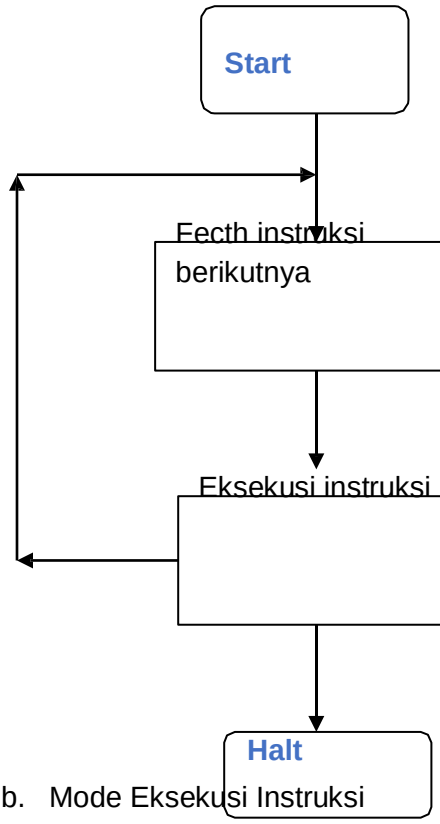
Sistem interkoneksi antar komponen yang popular antara lain : ISA, VESA dan PCI.

1. Eksekusi Instruksi
    a. Mekanisme Eksekusi Instruksi

    Fungsi utama komputer adalah mengeksekusi program. Tahap pemrosesan instruksi pada komputer :

a) Pemroses membaca instruksi dari memori (fetch)
b) Pemroses mengeksekusi instruksi (execute)

Eksekusi program berupa pengulangan fetch dan execute, seperti pada diagram berikut.



Siklus fetch

Siklus eksekusi

b. Mode Eksekusi Instruksi

Mode eksekusi berdasarkan kewenangan :

a) Program bagian dari sistem operasi
b) Program pemakai

Instruksi yang dapat dieksekusi pada mode berkewenangan tinggi :

a) membaca atau memodifikasi register kendali
b) instruksi primitif perangkat masukan/keluaran
c) instruksi untuk manajemen memori
d) dan bagian memori tertentu

Mode pemakai (mode dengan berkewenangan rendah) karena program pemakai dieksekusi pada mode ini. Mode dengan kewenangan tinggi mempunyai beberapa istilah yaitu mode

sistem atau mode kendali atau mode supervisor atau mode kernel. Mode ini berfungsi untuk mengeksekusi rutin sistem atau kendali atau kernel. Tujuan dari pemisahan mode tersebut yang utama yaitu untuk keamanan sistem, mencegah intervensi dari program pemakai ke tabel sistem operasi seperti PCB (tabel proses). Pada mode kernel, maka perangkat lunak mempunyai kendali penuh terhadap pemroses, instruksi, register dan memori. Bila program pemakai meminta sistem operasi dengan memanggil system call akan menyebabkan trap, sehingga sistem operasi mengubah mode eksekusi menjadi mode kernel. Pada mode ini sistem operasi akan memenuhi permintaan dari program pemakai, dan begitu selesai sistem operasi akan segera mengubah menjadi mode pemakai dan mengembalikan kendali ke program pemakai.

Manfaat dari dua mode dan sistem trap :

a) mencegah program pemakai mengacau tabel sistem operasi
b) mencegah program pemakai mengacau mekanisme pengendaliaan sistem operasi

Interkoneksi antar komponen

Karakter bus :

- Jumlah interupsi menentukan banyaknya perangkat independen yang melakukan I/O
- Ukuran bus data eksternal berakibat pada kecepatan operasi I/O
- Ukuran bus alamat menentukan banyak memori yang ditunjuk board ekspansi
- Kecepatan clock maksimum yang dapat diakomodasi bus berakibat pada kinerja

Daftar Pustaka

1. William Stallings, Sistem Operasi, Edisi Keempat, Indeks-Kelompok Gramedia, 2005.
2. Sri Kusumadewi, Sistem Operasi, J&J learning, 2000
3. Bambang Hariyanto, Ir, Sistem Operasi, Informatika Bandung,2000
4. Masyarakat Digital Gotong Royong (MDGR), Pengantar Sistem Operasi Komputer, 2006

MODUL PERKULIAHAN

Sistem Operasi

2. Sekilas Sistem Operasi

Abstract

Sistem Operasi komputer merupakan software pada lapisan pertama memori komputer. Sistem Operasi akan melakukan layanan inti umum untuk software-software. Sistem operasi harus dapat berkembang karena : upgrade

Kompetensi

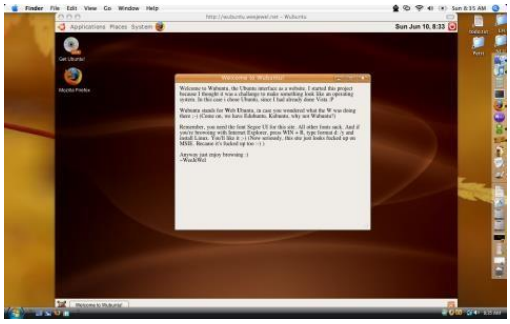
Mahasiswa mampu memahami tentang Sistem Operasi sebagai penghubung antara lapisan hardware dan lapisan software.
Mahasiswa memahami perkembangan sistem operasi.

hardware dan hardware jenis baru, layanan baru dan perbaikan. Evolusi system operasi : pengolahan serial, system batch sederhana, system batch multiprogramming dan system time sharing.

Sekilas sistem operasi

Sistem Operasi komputer merupakan software pada lapisan pertama yang diletakkan pada memori komputer (Hardisk, bukan memory ram) pada saat komputer dinyalakan. Sedangkan software-software lainnya dijalankan setelah sistem operasi komputer berjalan. Sistem Operasi akan melakukan layanan inti umum untuk software-software itu seperti seperti akses ke disk, manajemen memori, skeduling task, dan antar-muka user. Sehingga masing-masing software tidak perlu lagi melakukan tugas-tugas inti umum tersebut, sebab hal tersebut sudah dapat dilayani dan dilakukan oleh Sistem Operasi. Bagian kode yang melakukan tugas-tugas inti dan umum tersebut dinamakan dengan kernel.

Sistem Operasi berfungsi sebagai penghubung antara lapisan hardware dan lapisan software, melakukan semua perintah perintah penting dalam komputer, serta menjamin aplikasi-aplikasi yang berbeda fungsinya dapat berjalan lancar secara bersamaan. *Sistem operasi komputer* menjamin aplikasi perangkat lunak lainnya bisa memakai memori, melakukan input serta output terhadap peralatan lain, dan mempunyai akses kepada sistem file. Jika beberapa aplikasi berjalan secara bersamaan, maka **sistem operasi** akan mengatur penjadwalan yang tepat, sehingga sebisa mungkin semua proses pada komputer yang berjalan mendapatkan waktu yang cukup untuk menggunakan CPU dan tidak saling mengganggu dengan perangkat yang lain.



Antar muka linux Ubuntu



Antar muka sistem operasi Mac OS X



a. Tujuan sistem operasi :

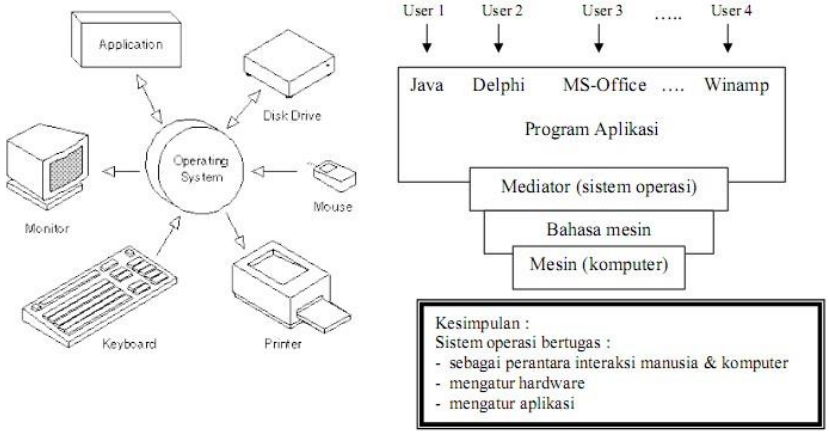
a. Agar dapat merancang sendiri
b. Dapat memodifikasi sistem yang telah ada sehingga sesuai dengan kebutuhan
c. Dapat memilih di antara berbagai alternatif sistem operasi
d. Memaksimalkan penggunaan sistem operasi
e. Konsep dan teknik sistem operasi dapat diterapkan pada aplikasi-aplikasi lain.

Sistem operasi sebagai resource manager yaitu pengelola seluruh sumber daya yang terdapat pada sistem komputer dan sebagai extended machine yaitu menyediakan sekumpulan layanan ke pemakai sehingga memudahkan dan menyamankan penggunaan serta pemanfaatan sumber daya sistem komputer.

Sistem operasi merupakan program komputer yang berisi perintah-perintah (command) dan bertugas menjembatani pengertian manusia dengan komputer, sehingga komputer dapat bekerja sesuai keinginan.

b. Definisi sistem operasi :

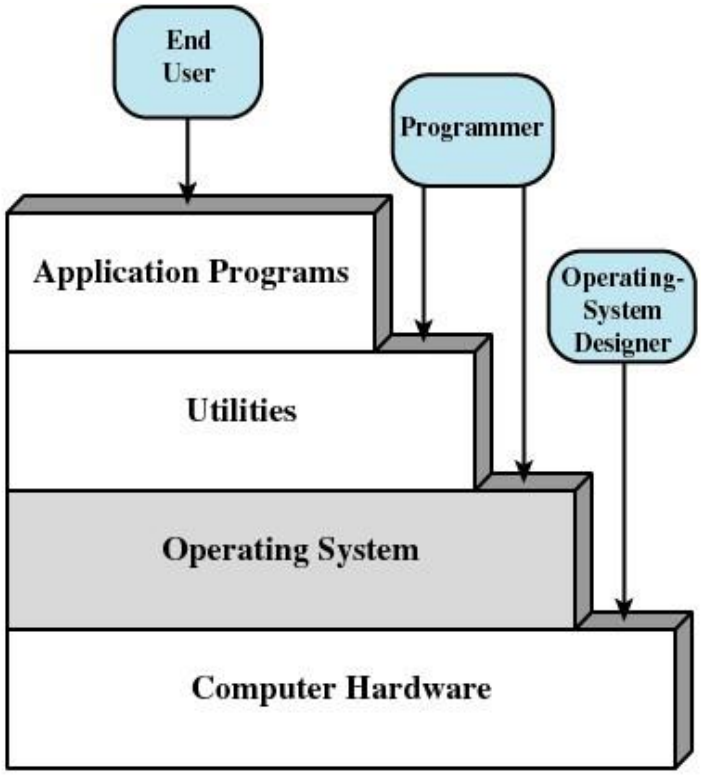
a. Sistem operasi adalah software yang mengontrol hardware.
b. Program yang menjadikan hardware lebih mudah untuk digunakan
c. Kumpulan program yang mengatur kerja hardware sesuai keinginan user
d. Manager sumber daya atau pengalokasian sumber daya komputer, seperti mengatur memori, printer, dll
e. Sebagai program pengendali, yaitu program yang digunakan untuk mengontrol program yang lain
f. Sebagai kernel, yaitu program yang terus-menerus running selama computer dihidupkan
g. Sebagai guardian yang menjaga komputer dari berbagai kejahatan komputer.



c. Sistem operasi sebagai interface antara pengguna dan computer

Sistem operasi sebagai interface antara pengguna dan computer pada umumnya menyediakan layanan berupa :

- Pembuatan program, seperti editor, debugger dan utilitas lainnya
- Eksekusi program, menjalankan instruksi dan memuat data di memori utama, perangkat I/O. System operasi menangani semua task untuk kepentingan pengguna.
- Access ke perangkat I/O. Sistem operasi menjaga detailnya sehingga pembacaan dan penulisan terlihat sederhana.
- Access terkontrol ke file. Sistem operasi menyediakan mekanisme proteksi untuk mengontrol akses ke file.
- Deteksi error dan respons. Sistem operasi membuat respons yang dapat menghilangkan error dengan dampak paling sedikit terhadap aplikasi yang sedang berjalan.
- Accounting. Sistem operasi menyediakan statistic pemakaian sumber daya dan memonitor kinerja sumber daya.



Layers and Views of a Computer System

d. Pengelolaan sumber daya komputer

Sumber daya computer secara umum dibedakan atas dua bagian, yaitu :[3]

a. Sumber daya phisik :
   - keyboard, bar-code reader
   - mouse, light-pen, trackball, joystick, touch-screen, pointer device dsb.
   - hard *disk*, floppy *disk*, tape drive, optical drive, CD rom drive dsb
   - layar monitor (CRT, LCD) dsb
   - printer
   - modem, ethernet card, dsb
   - RAM, chace memory, register dan memory volatile lainnya
   - Perangkat mulimedia, kamera, sound card dsb
   - Perangkat grafis, digitizer, scanner, plotter dsb
   - Sensor dan actuator

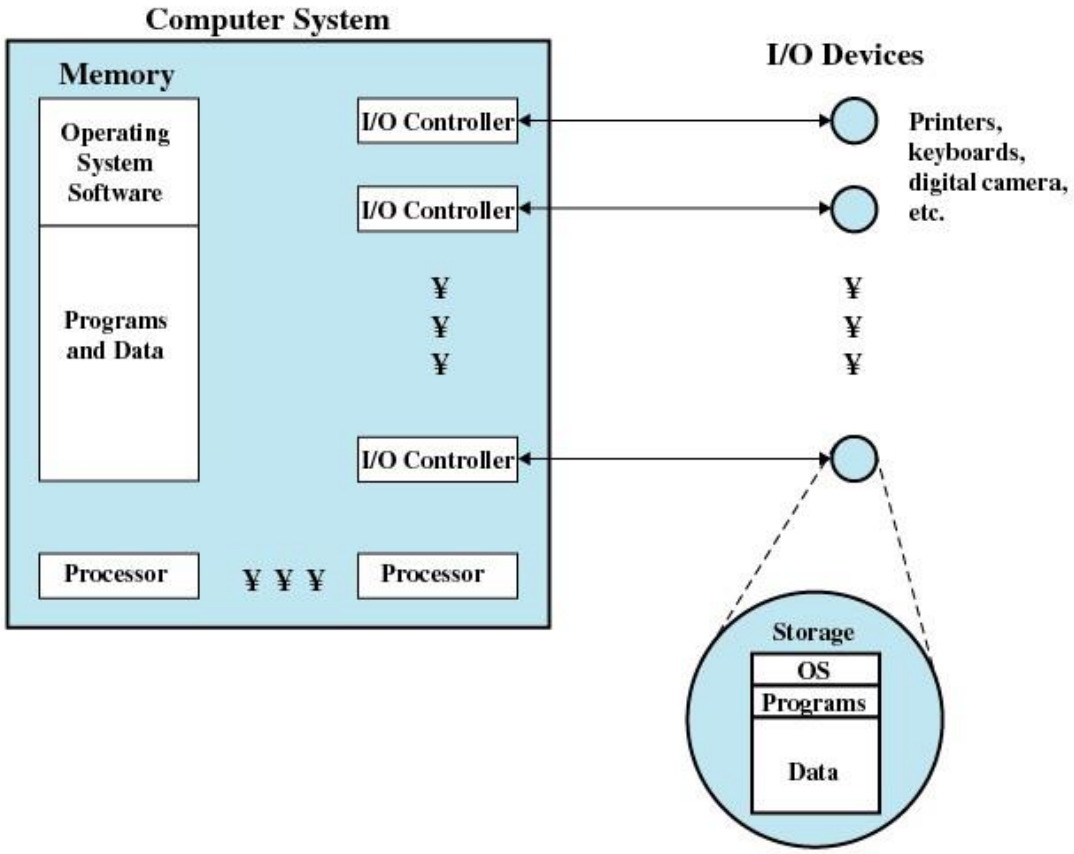
b. Sumber daya abstrak :

a) Data :

- semaphore untuk pengendali sinkronisasi proses
- PCB (process control b*lock*) untuk mencatat dan mengendalikan proses
- Tabel segmen, tabel page, I-node, FAT untuk mengendalikan memori
- Berkas (file) untuk menyimpan data dan program

b) Program : kumpulan instruksi yang dapat dijalankan sistem komputer

- utilitas
- aplikasi



The Operating System as Resource Manager

Penyedia layanan

Sistem operasi sebagai penyedia layanan maka system operasi harus dapat memudahkan dan menyamankan penggunaan sumber daya sistem komputer sehingga pemrogram dapat mengakses sumber daya yang ada tanpa harus memrogram instruksi – instruksi mesin yang rumit.

e. Sistem operasi sebagai extended machine/virtual machine berfungsi :

- memberi abstraksi mesin tingkat tinggi yang lebih sederhana dan menyembunyikan kerumitan perangkat keras, dengan menyediakan system call / API untuk menghindarkan kompleksitas pemrograman dan memberi sekumpulan instruksi yang mudah dan nyaman
- basis bagi program lainnya, memberi layanan pada program lainnya dalam mengendalikan sumber daya bagi aplikasi secara benar dan efisien.

Menurut Stalling, sistem operasi harus menyediakan layanan : [1]

- Pembuatan program : biasanya dalam bentuk utilitas
- Eksekusi program : dapat melakukan eksekusi program
- Pengaksesan perangkat I/O : tiap perangkat I/O memerlukan sejumlah instruksi agar perangkat dapat beroperasi.
- Pengaksesan terkendali terhadap berkas : Sistem operasi menyediakan mekanisme proteksi untuk mengendalikan berkas
- Pengaksesan sistem : dalam bentuk sharing
- Deteksi dan memberi tanggapan terhadap kesalahan : dapat memberi tanggapan dan memberi penjelasan tentang kesalahan dan dampak terhadap aplikasi ayng sedang berjalan.
- Akunting : mengumpulkan data statistik penggunadan memonitor parameter kinerja, seperti waktu tanggap.

Fungsi – fungsi minor sistem operasi :

- implementasi antarmuka pemakai
- sistem sharing pada perangkat keras
- sistem sharing pada data

- mencegah pemakai saling mengganggu
- menjadwal pemakaian sumber daya
- memberi fasilitas I/O
- memulihkan kesalahan
- menghitung penggunaan sumber daya
- pengorganisasian data
- komunikasi jaringan

Untuk memenuhi fungsi diatas sistem operasi mempunyai subsistem :

- Manajemen proses
- Manajemen memori
- Manajemen berkas
- Manajemen perangkat I/O
- Pengamanan sistem
- Sistem komunikasi, dsb

f. Evolusi system operasi

Sistem operasi harus dapat berkembang dengan alasan :

- Upgrade hardware dan hardware jenis baru. Dengan perkembangan hardware yang baru maka system operasi harus dapat mengikuti perkembangan tersebut.
- Layanan baru. Untuk menjawab kebutuhan pengguna atau manajer system maka system operasi harus menambah layanan baru.
- Perbaikan. Setiap system operasi mempunyai fault maka perlu adanya perbaikan dari system operasi.

Perkembangan system operasi

a. Pengolahan serial

Terjadi pada computer lama (akhir 1940 s/d pertengahan 1950). Pengguna langsung berinteraksi dengan hardware tanpa system operasi. Permasalahan utama yaitu :

- Penjadwalan. Umumnya instalasi menggunakan formulir pemesanan waktu mesin yang mempunyai periode tertentu. Jika pengguna menggunakan waktu lebih sedikit

dari waktu yang dipesan maka akan tidak efisien atau jika langsung dihentikan maka akan timbul masalah penjadwalan.

- Set up waktu. Sebuah program tunggal (job) meliputi suatu rangkaian kegiatan yang terdiri dari pemuatan compiler di memori, penyimpanan program, pemuatan dan penggabungan program obyek dan fungsi. Dimana semua kegiatan tersebut memuat pemasangan dan penanggalan pita atau penyusunan kartu, jika terjadi error maka semua kegiatan harus dimulai dari awal.

b. System batch sederhana

Mesin lama sangat mahal dan adanya dua masalah tersebut maka dipikirkan untuk memaksimalkan utilisasi dengan dibuatnya konsep batch. Muncul pertengahan tahun 1950 dengan adanya IBM 704 yang dibuat oleh General Motor. Awal tahun 1960 dikembangkan system operasi batch untuk system computer, yaitu IBSYS untuk system operasi IBM 7090/7094. Inti gagasan system operasi batch adalah penggunaan potongan software yang dikenal sebagai monitor. Secara garis besar monitor menangani masalah penjadwalan. Setiap job diantrikan dan akan dieksekusi secepat mungkin. Instruksi tersebut dimasukan dalam bentuk primitive job control language (JCL). Monitor mengandalkan kemampuan prosesor dalam membaca instruksi dari berbagai memori utama. Feature hardware lain yang dibutuhkan :

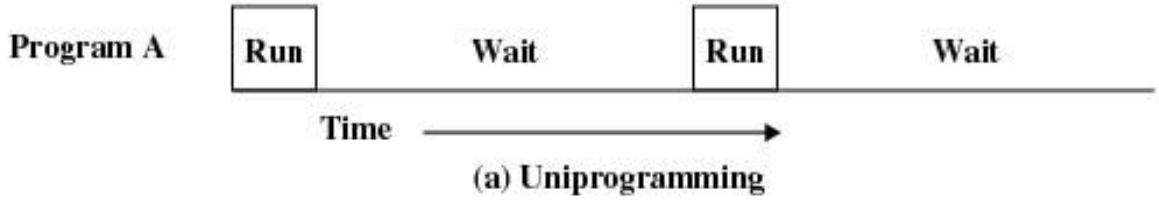
- Proteksi memori. Pada saat program pengguna sedang mengekssekusi maka program tersebut tidak boleh mengubah daerah memori yang lain.
- Timer. Digunakan untuk mencegah suatu job yang akan memonopoli system.
- Privileged instructins. Instruksi tertentu ditandai dengan privileged dan hanya bias dieksekusi oleh monitor.
- Interrupt. Menjadikan system lebih fleksiobel dalam melepaskan kontrolnya ke pengguna dan memperoleh kembali dari pengguna.

c. System batch multiprogramming

Sistem single batch tidak efisien jika terdapat beberapa job yang akan menggunakannya. Berikut gambaran hal tersebut

Sistem batch uniprogramming

Prosesor harus menunggu instruksi I/O lengkap sebelum diproses



(a) Uniprogramming

Efisiensi Prosesor pada uniprogramming

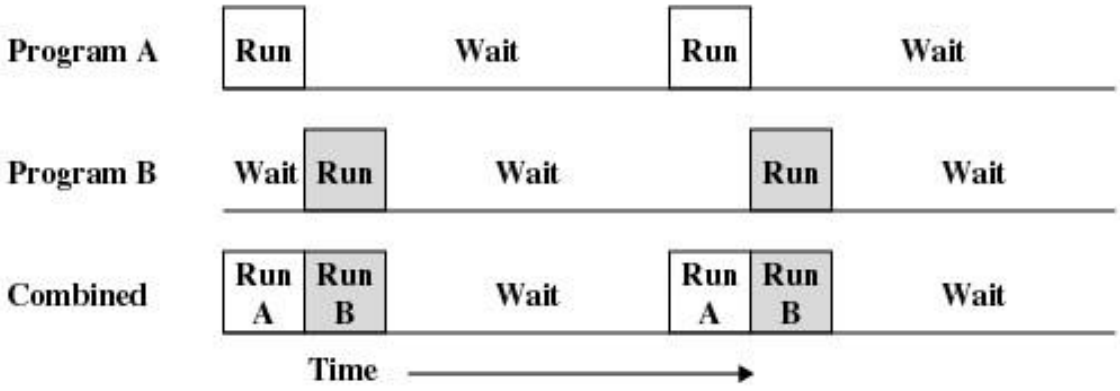
Read one record from file	15 µs
Execute 100 instructions	1 µs
Write one record to file	15 µs
TOTAL	31 µs

Percent CPU Utilization $= \frac{1}{31} = 0.032 = 3.2\%$

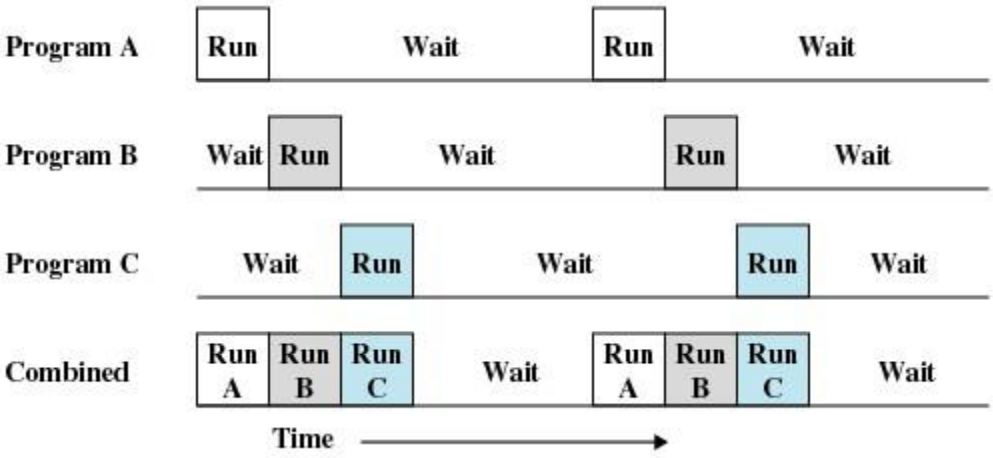
System Utilization Example

Sistem batch multiprogramming

Ketika suatu job masih menunggu I/O maka prosesor dapat mengerjakan job lainnya



Multiprogramming with two programs

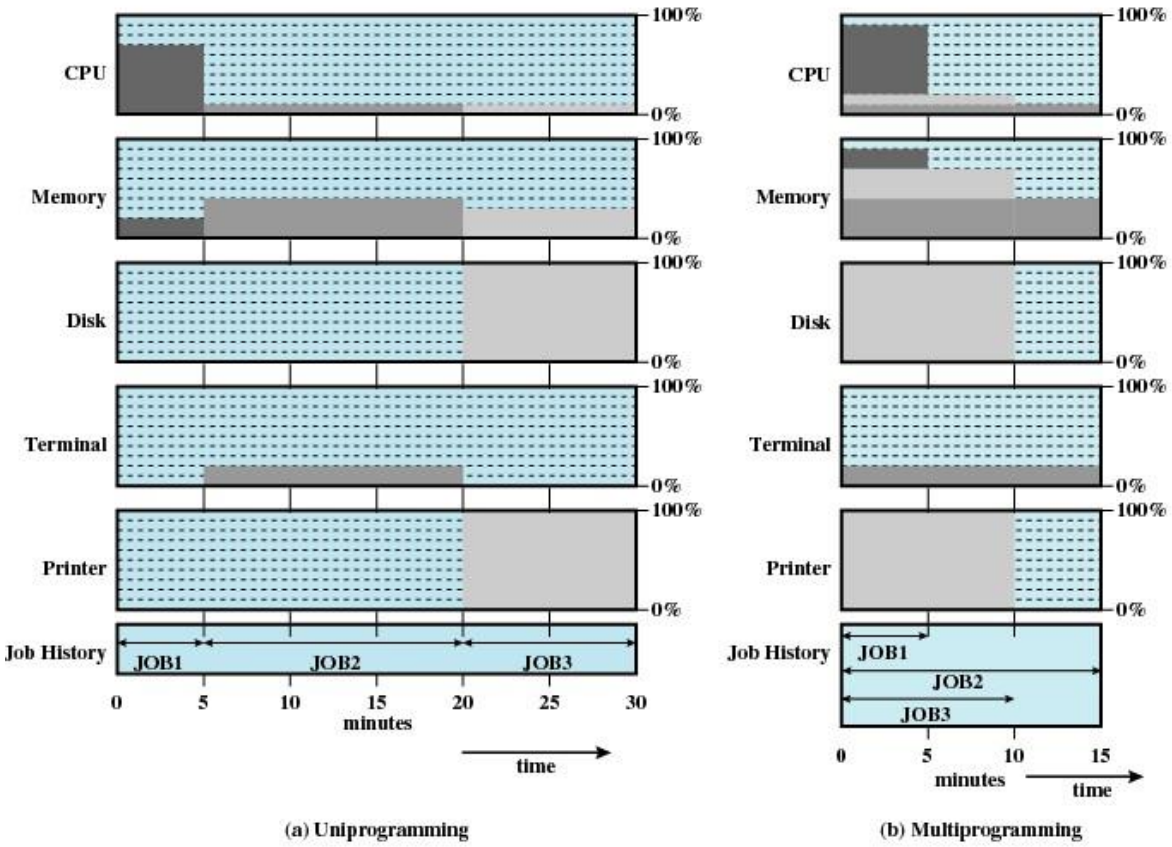


Multiprogramming with three programs

Sample Program Execution Attributes

	JOB1	JOB2	JOB3
Type of job	Heavy compute	Heavy I/O	Heavy I/O
Duration	5 min	15 min	10 min
Memory required	50 M	100 M	75 M
Need disk?	No	No	Yes
Need terminal?	No	Yes	No
Need printer?	No	No	Yes

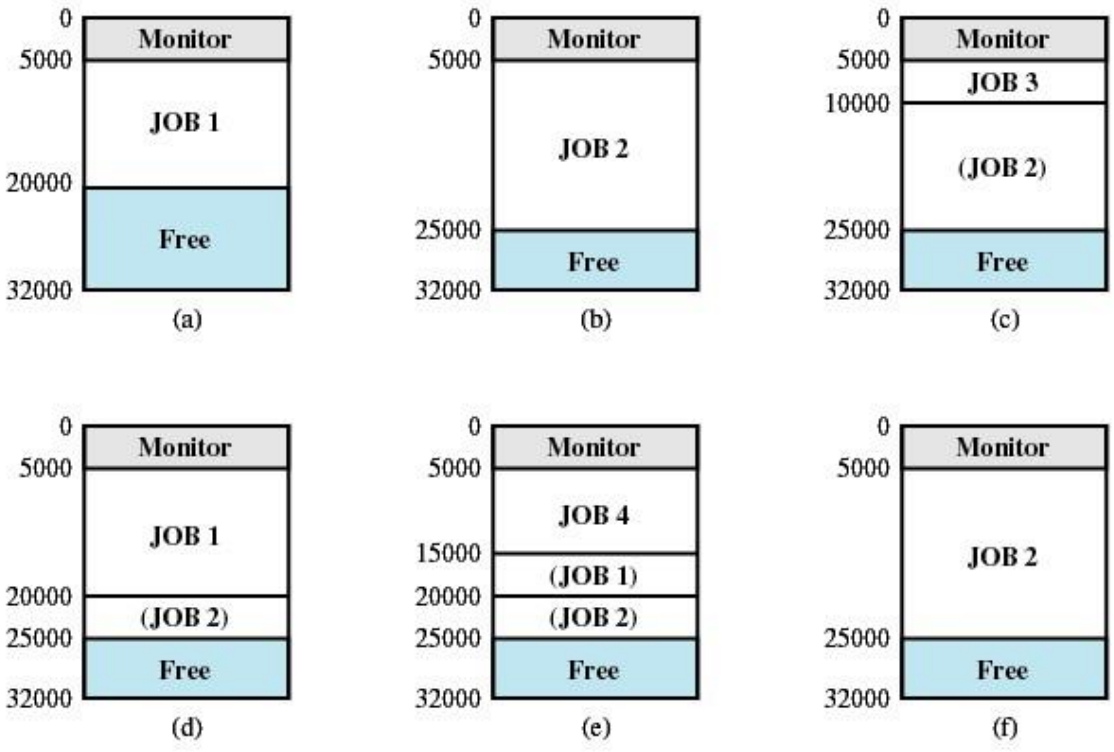
Histogram utilisasi



Utilization Histograms

d. System time sharing

- Penggunaan multiprogramming dapat menghandle beberapa job secara bersamaan
- Beberapa user dapat mengakses secara simultan
- Waktu yang diperlukan prosesor adalah hasil sharing dari beberapa user



CTSS Operation

Daftar Pustaka

5. William Stallings, Sistem Operasi, Edisi Keempat, Indeks-Kelompok Gramedia, 2005.
6. Sri Kusumadewi, Sistem Operasi, J&J learning, 2000
7. Bambang Hariyanto, Ir, Sistem Operasi, Informatika Bandung,2000
8. Masyarakat Digital Gotong Royong (MDGR), Pengantar Sistem Operasi Komputer, 2006

MODUL PERKULIAHAN

Sistem Operasi

3. Proses

Abstract

Proses merupakan unit kerja terkecil yang secara individu memiliki sumber daya dan dijadwalkan sistem operasi. Diagram proses 2 keadaan (running dan not running) serta diagram proses 5 keadaan. PCB (process control block) berisi informasi mengenai proses guna pengelolaan proses, yang diperlukan

Kompetensi

Mahasiswa mampu memahami tentang proses dan operasi yang terjadi pada proses.

oleh system operasi. Operasi-operasi yang terjadi pada proses, yaitu penciptaan, penghancuran, penundaan dsb.

PROSES

Fungsi utama sebuah prosesor adalah mengeksekusi instruksi mesin yang terdapat dalam memori utama. Eksekusi suatu program dikenal dengan nama proses atau task. Proses berisi instruksi dan data, program counter dan semua register pemroses, dan stack berisi data sementara. Proses merupakan unit kerja terkecil yang secara individu memiliki sumber daya dan dijadwalkan sistem operasi. Tanggung jawab utama dari system operasi adalah mengontrol eksekusi proses. Tanggung jawab ini mencakup pola penggiliran eksekusi dan pengalokasian sumber daya. Hal – hal yang berkaitan dengan proses :

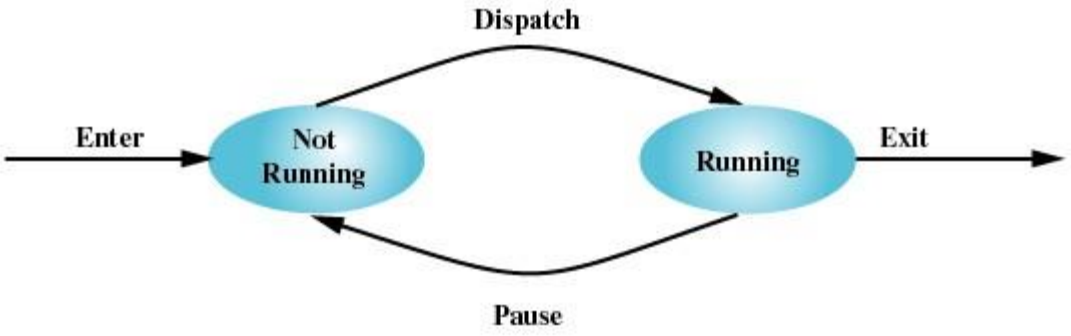
- *multiprogramming* : manajemen banyak proses pada satu pemroses, contoh komputer pribadi bersifat workstation artinya sistem pemroses tunggal yang dapat menjalankan banyak program/task seperti MS-Windows 3.0, MS-Windows NT, OS/2 dan Macintosh system 7. Program yang dijalankan sebenarnya bersifat independent dan one program at any instant (satu program pada satu saat).
- *Multiprocessing* : manajemen banyak proses dalam satu komputer, contoh pada sistem operasi microsoft windows NT, UNIX, Linux telah menyediakan fasilitas multiprocessing.
- *Distributed processing* : manajemen banyak proses yang dieksekusi di banyak sistem komputer yang tersebar.

Kebutuhan utama pengendali proses :

- Saling melanjutkan (interleave) : bertujuan memaksimumkan penggunaan pemroses sambil memberi waktu tanggap.
- Mengikuti kebijaksanaan tertentu : SO harus mengalokasikan sumber daya ke proses mengikuti kebijaksaan yang sudah ditentukansambil menghindari *deadlock*.
- Mendukung komunikasi antar proses dan pencipta proses : SO harus dapat mengetahui state masing – masing proses dan merekam semua perubahan yang terjadi secara dinamis (berguna untuk penjadwalan dan memutuskan alokasi sumber daya).

a. Diagram state proses

a. Diagram state dasar terdiri dari 2 keadaan : proses akan berada pada salah satu dari dua state yaitu running atau tidak running.



(a) State transition diagram

Jika system operasi membuat proses baru maka system operasi akan memasukkan proses tersebut dalam kondisi tidak running. Proses yang keluar akan diketahui oleh system operasi dan menunggu kesempatan untuk dieksekusi. Tetapi ada juga suatu proses yang sedang running akan diinterrupt jika ada proses lain yang mempunyai prioritas lebih bagus. Proses yang diinterrup tersebut akan kembali ke posisi not running.

Pada gambar diatas ditunjukan antrian tunggal dari beberapa proses yang menunggu untuk dieksekusi.

b. Diagram proses 5 keadaan (proses dapat berada dalam salah satu keadaan)

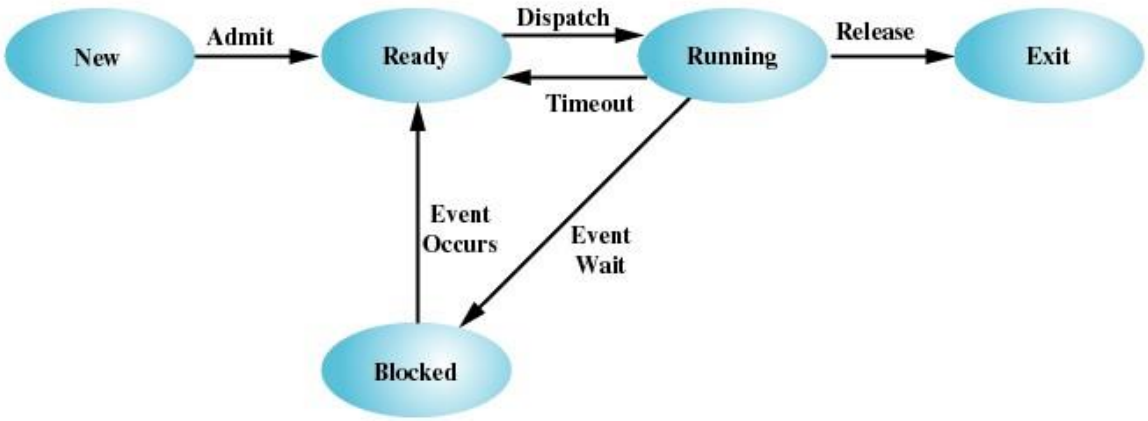


Figure 3.6 Five-State Process Model

Status	**Deskripsi**
Running	Pemroses sedang mengeksekusi intruksi proses
Ready	Proses siap dieksekusi bila ada pemroses yang tersedia
Blocked	Proses menunggu kejadian untuk melengkapi tugasnya seperti : - selesainya operasi perangkat I/O - tersedianya memori - datangnya pesan
New	Proses baru diciptakan, tetapi belum diijinkan masuk ke pool proses-proses yang dapat dieksekusi oleh system operasi
Exit	Suatu proses yang telah dibebaskan dari pool proses yang dapat dieksekusi system operasi.

- Proses yang baru diciptakan akan segera mempunyai state ready
- Proses running menjadi *blocked* karena sumber daya yang diminta belum tersedia, proses menunggu alokasi sumber daya (event wait)
- Proses running menjadi ready karena penjadwalan memutuskan eksekusi proses lain karena jatah waktu telah habis (time out)
- Proses *blocked* menjadi ready saat sumber daya yang diminta tersedia
- Proses ready menjadi running karena penjadwal memutuskan penggunaan pemroses untuk proses itu.

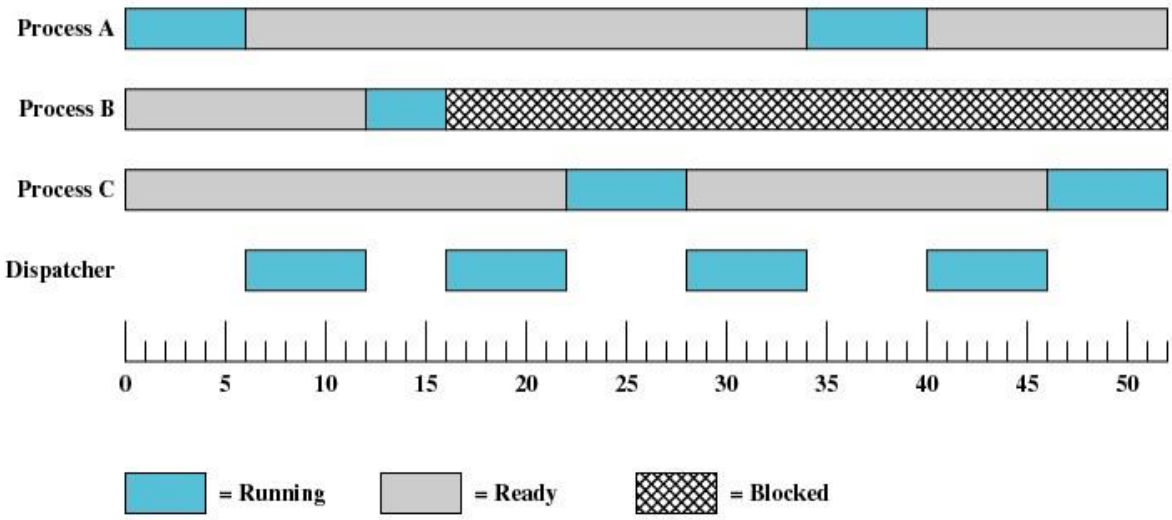
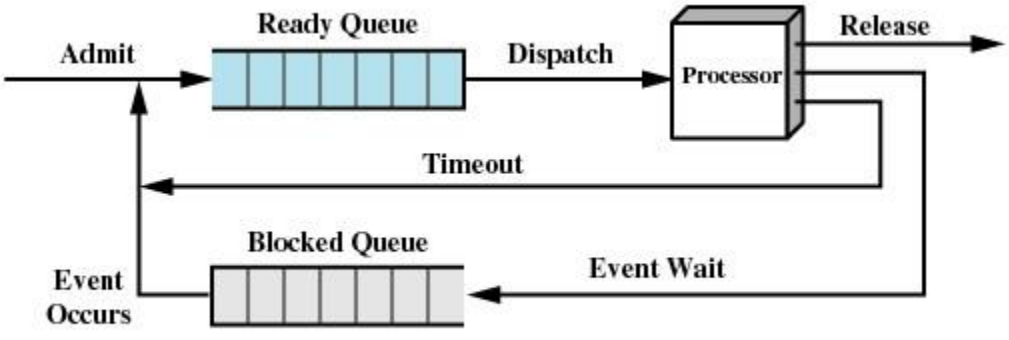


Figure 3.7 Process States for Trace of Figure 3.4

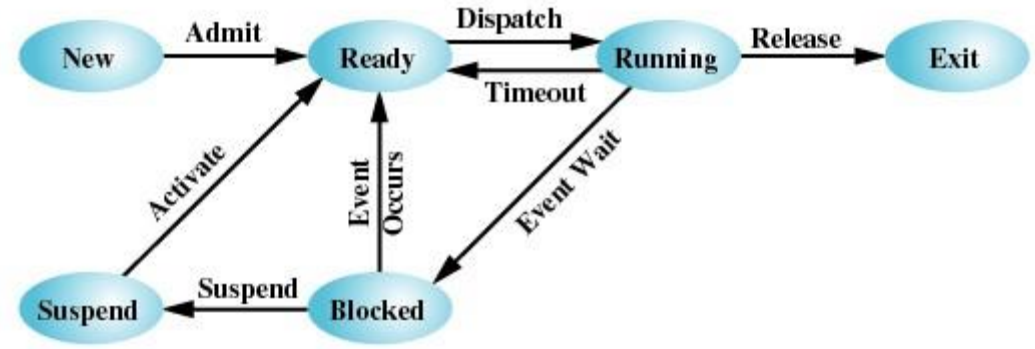


(a) Single blocked queue

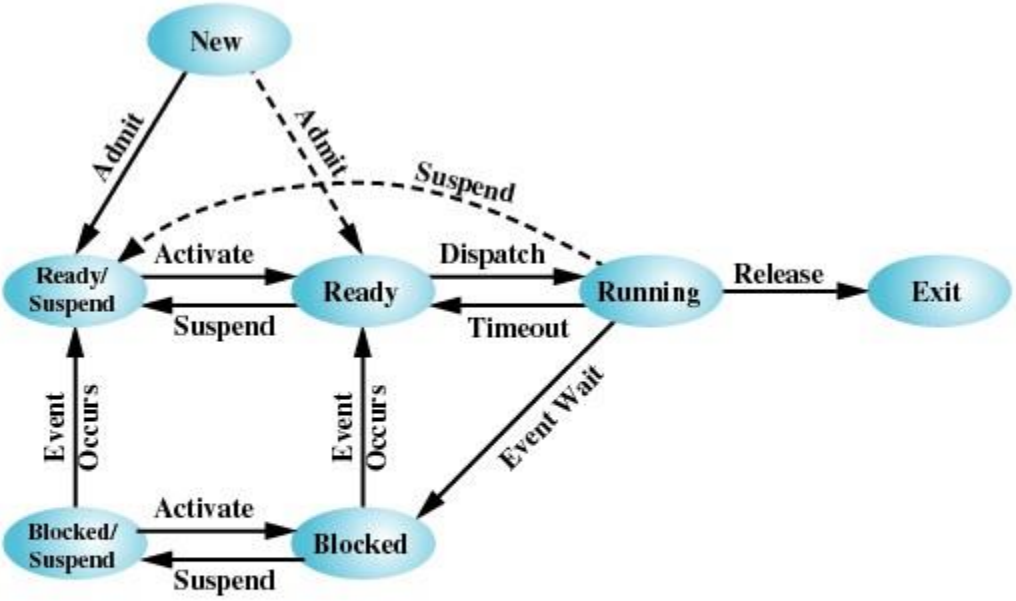
b. Penundaan proses

Penundaan proses (suspended process) yaitu suatu kondisi dimana proses mengalami penundaan pada saat running, kondisi ini disebabkan oleh beberapa hal :

- Prosesor lebih cepat daripada I/O sehingga banyak proses yang menunggu I/O
- Swap these processes to disk to free up more memory
- Blocked state becomes suspend state when swapped to disk



(a) With One Suspend State



(b) With Two Suspend States

Figure 3.9 Process State Transition Diagram with Suspend States

Alasan – alasan penundaan proses

Kodisi	**Diskripsi**
swapping	System operasi perlu membebaskan memori utama untuk dapat membawa proses yang siap untuk dieksekusi ke dalamnya
OS lainnya	System operasi dapat menunda proses yang dianggap bermasalah
Permintaan pengguna	User dapat menunda eksekusi program untuk tujuan debugging atau yang berhubungan dengan sumber daya
Pewaktuan	Proses dapat ditunda untuk interval waktu tertentu
Permintaan induk proses	Penundaan dari induk proses kepada turunannya untuk sinkronisasi

c. PCB (Program Control B*lock*)

Sistem operasi memerlukan banyak informasi mengenai proses guna pengelolaan proses, informasi ini berada di PCB. Dalam PCB informasi dibagi atas 3 kelompok :

a. Informasi identifikasi proses : identifikasi berkaitan dengan proses yang unik.
   Identifier numerik yang meliputi : prose situ sendiri, siapa proses yang menciptakan dan yang pemakai

b. Informasi status pemroses : berisi register – register pemroses. Jumlah dan jenis register tergantung arsitek komputer. Register tersebut terdiri :
   - register yang terlihat pemakai : dapat ditunjuk instruksi bahasa assembly untuk diproses pemroses
   - register kendali dan status : untuk mengendalikan operasi pemroses seperti: program counter, PSW dsb.
   - Pointer stack : untuk parameter atau alamat prosedur pemanggil dan system call, pointer stack menujuk posisi paling atas dari stack.

c. Informasi Kendali proses yang terdiri atas :

- informasi penjadwalan dan status : untuk menjalankan fungsi penjadwalan : status proses (running, ready, *blocked* dsb), prioritas (lama menunggu, lama proses dsb) dan kejadian (identitas kejadian yang ditunggu proses)
- penstruktur data : satu proses dapat dikaitkan dengan proses lain dalam satu antrian atau ring atau struktur lainnya.
- Komunikasi antar proses : beragam *flag*, sinyal dan pesan dapat diasosiasikan dengan komunikasi antara dua proses yang terpisah.
- Kewenangan proses : kewenangan berkaitan dengan memori dan tipe instruksi yang dapat dijalankan.
- Manajemen memori : berisi pointer ke tabel segmen atau page yang menyatakan memori maya proses.
- Kepemilikan dan utilisasi sumber daya : sumber daya yang dikendalikan proses harus diberi tanda : berkas dibuka, pemakaian pemroses dan pemakaian sumber daya lainnya.

d. Operasi – operasi pada proses

Operasi – operasi terhadap proses antara lain :

- penciptaan proses (create a process)
- penghancuran proses (destroy a process)
- penundaan proses (suspend a process)
- pelanjutan kembali proses (resume a process)
- pengubahan prioritas process
- memblok proses
- membangunkan proses
- menjadwal proses
- memungkinkan proses berkomunikasi dengan proses lain

Penciptaan proses melibatkan banyak aktivitas seperti : menamai, menyisipkan, menentukan prioritas, menciptakan PCB dan mangalokasikan sumber daya proses.Kejadian yang dapat menciptakan proses :

- pada lingkungan batch sebagai tanggapan atas job

- pada lingkungan interaktif, terjadi saat pemakai berusaha log on
- sebagai tanggapan suatu aplikasi
- adanya penciptaan proses lain (anak proses)

Pada UNIX : proses diciptakan dengan *system call fork* : menciptakan kopian proses pemanggil (induk) yang identik.

Pada MS-DOS : *system call* adalah memuatkan file biner ke memori dan mengeksekusi sebagai anak proses. Pada sistem ini proses induk dan proses anak tidak berjalan paralel.

Penghancuran proses : pada penghancuran proses maka akan melibatkan kegiatan pembebasan proses dari system, antara lain :

- sumber daya yang dipakai dikembalikan
- proses dihancurkan dari tabel sistem
- PCB dihapus

Penghancuran akan lebih rumit bila telah menciptakan proses. Untuk kondisi tersebut maka akan dilakukan pendekatan, antara lain :

- pada beberapa sistem proses-proses turunan dihancurkan saat proses induk dihancurkan secara otomatis
- menganggap proses anak independen terhadap proses induk

Alasan – alasan penghancuran proses :

Penyebab terminasi	**Deskripsi**
Selesainya proses secara normal	Proses telah berjalan secara lengkap
Batas waktu telah terlewati	Proses telah melewati batas waktu total yang telah dispesifikasikan
Memori tidak tersedia	Keperluan memori lebih banyak daripada yang diesiakan sistem
Pelanggaran terhadap batas memori	Mencoba mengakses lokasi memori yang tidak diijinkan diakses

Terjadi kesalahan karena pelanggaran proteksi	Mencoba menggunakan sumber daya yang tidak diijinkan
Terjadi kesalahan aritmatika	Mencoba perhitungan terlarang, seperti pembagian dengan nol
Waktu telah kadaluwarsa	Telah lama menunggu daripada waktu maksimum yang ditentukan
Instruksi yang tak benar	Berusaha mengeksekusikan instruksi yang tak ada seperti pencabangan
Terjadi usaha memakai instruksi yang tak diijinkan	Menggunakan instruksi yang disimpan untuk sistem operasi
Terjadi kegagalan I/O	Kesalahan muncul pada masukan atau keluaran
Kesalahan pengguna data	Ada data yang tipenya salah atau tidak terinisialisasi
Diintervensi oleh sistem operasi atau operator	Karena alasan tertentu operator mengakhiri proses
Berakhirnya proses induk	Ketika parent berakhir, sistem akan mengakhiri semua anak proses
Atas permintaan dari proses induk	Parent proses biasanya mempunyai otoritas mengakhiri anak proses

Pada UNIX penghancuran proses dengan memanggil system call exit( ). Pada MS-DOS terdapat terminate program.

e. Implementasi Proses

Setiap proses mempunyai state yang perlu diperhatikan sistem operasi yang dicatat dalam tabel :

a. Tabel informasi manajemen memori untuk menjaga keutuhan memori utama dan memori skunder, yang memuat : alokasi memori utama yang dipakai pemroses, alokasi memori skunder yang terpakai, atribut segmen utama dan sekunder dan informasi lain yang digunakan untuk pengelolaan memori.
b. Tabel informasi manajemen masukan /keluaran : mengelola perangkat I/O dan mengetahui statusnya dan lokasi memori utama yang digunakan transfer data
c. Tabel sistem informasi : berisi informasi eksetensi file, lokasi pada memori sekunder, statusnya dan informasi lainnya.

d. Tabel proses mengelola informasi proses di sistem operasi, lokasinya memori, status dan atribut lainnya.

Elemen citra proses	**Deskripsi**
Data pemakai	Bagian dapat dimodifikasi seperti data program, daerah stack pemakai
Program pemakai	Program biner yang dieksekusi
Stack sistem	Menyimpan parameter dan alamat pemanggilan untk prosedur dan system call
PCB	Informasi yang diperlukan sistem operasi dalam mengendalikan proses

Struktur umum tabel kendali pada system operasi

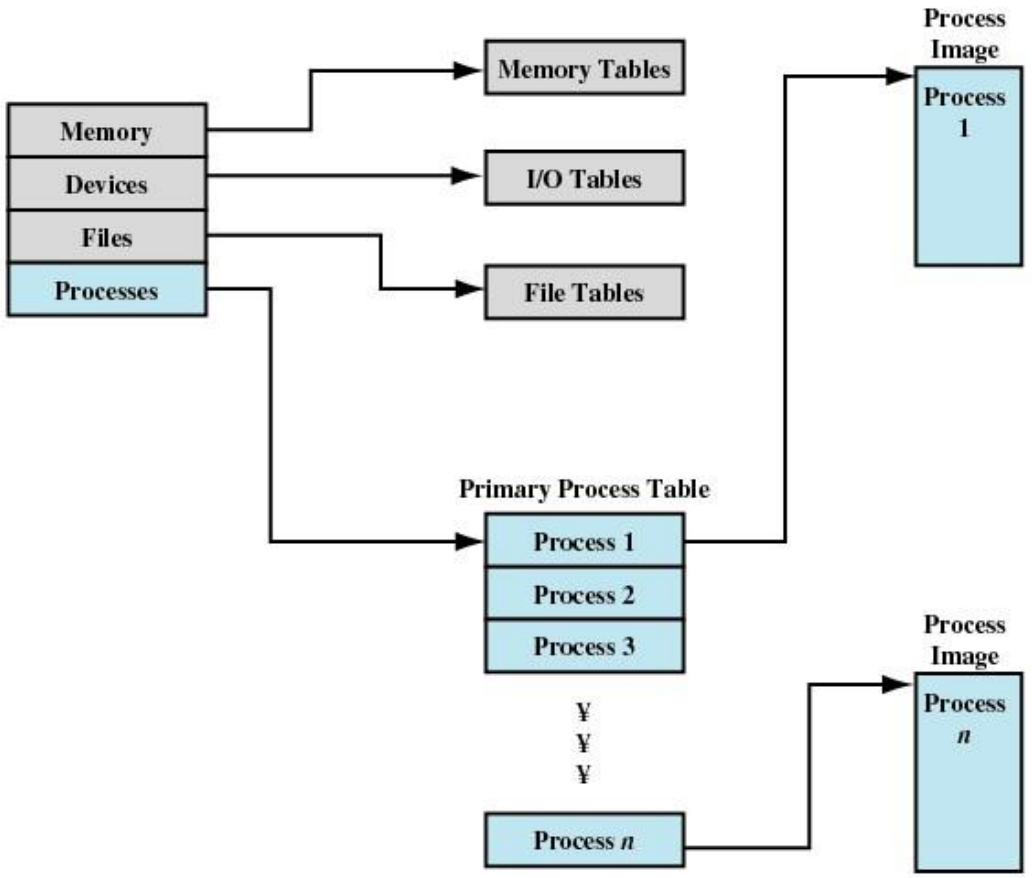
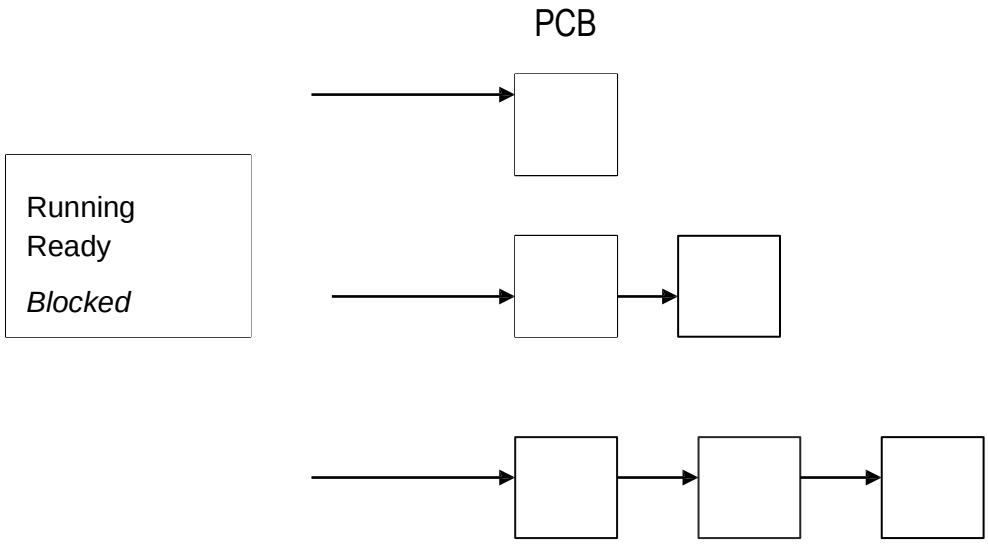


Figure 3.11 General Structure of Operating System Control Tables

PCB dan senarai proses

PCB dapat dibaca dan atau dimodifikasi rutin sistem operasi seperti penjadwalan, alokasi sumber daya, pemroses interupsi, monitoring dan analisis kinerja.

Untuk menyatakan senarai proses di sistem operasi dibuat senarai PCB



Pengaksesan informasi di PCB

Setiap proses dilengkapi denga ID yang unik digunakan sebagai indeks ke tabel untuk mengambil PCB, kesulitan yang terjadi disebabkan oleh proteksi terhadap PCB, dua masalah utama proteksi PCB :

a. Bug (kesalahan) pada rutin tunggal misalnya interrupt handler dapat merusak PCB
b. Perubahan rancangan struktur dan semantiks PCB

Kedua masalah tersebut diatasi dengan rutin penanganan PCB dalam pengaksesan. Tugas rutin ini adalah memproteksi PCB dan menjadi perantara pembacaan dan penulisan PCB. Sehingga masalah diatas dapat diatasi karena :

a. Kesalahan dapat dicegah sehingga PCB tidak rusak
b. Antar muka terhadap rutin – rutin lain masih dapat dijaga meskipun rincian PCB diubah.

Daftar Pustaka

9. William Stallings, Sistem Operasi, Edisi Keempat, Indeks-Kelompok Gramedia, 2005.
10. Sri Kusumadewi, Sistem Operasi, J&J learning, 2000
11. Bambang Hariyanto, Ir, Sistem Operasi, Informatika Bandung,2000
12. Masyarakat Digital Gotong Royong (MDGR), Pengantar Sistem Operasi Komputer, 2006

MODUL PERKULIAHAN

Sistem Operasi dan Sistem Komputer

Sistem Operasi komputer merupakan software pada lapisan pertama yang diletakkan pada memori komputer, berfungsi sebagai penghubung antara lapisan hardware dan lapisan software.

Abstract

Sistem Komputer adalah elemen-elemen yang terkait untuk menjalankan suatu aktifitas dengan menggunakan komputer, terdiri dari brainware, software, instruction

Kompetensi

Memahami konsep Sistem Komputer. Memahami tentang Arsitektur Von Neumann. Memahami tentang Sistem Operasi sebagai penghubung antara lapisan hardware dan lapisan software.

set, dan hardware.

Sistem Operasi dan Sistem Komputer

SKEMA DASAR SISTEM KOMPUTER

A. Perangakat Keras

Adalah komponen fisik komputer yang terdiri dari rangkaian elektronika dan peralatan mekanis lainnya. Pada abtraksi tingkat atas terdiri dari empat komponen, yaitu :

1. Pemroses (processor)

   Berfungsi mengendalikan operasi komputer & melakukan fungsi pemrosesan data.

2. Memori utama

   - Berfungsi menyimpan data & program

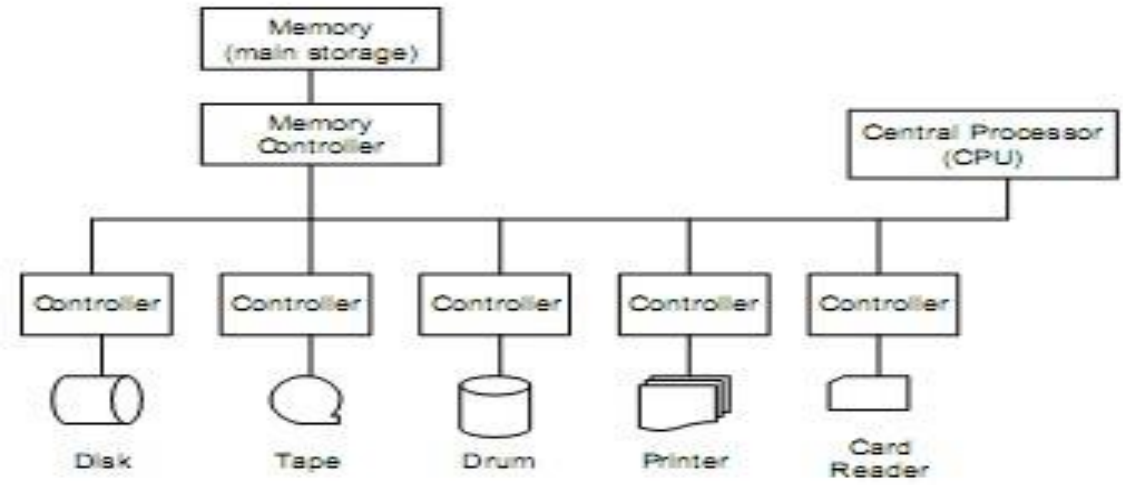
   - Biasanya volatile : tidak dapat mempertahankan data & program yang disimpan bila sumber daya energi (listrik) dihentikan.

3. Perangkat masukan dan keluaran

   Berfungsi memindahkan data antara komputer & lingkungan eksternal yaitu : perangkat penyimpan sekunder, perangkat komunikasi, terminal, dsb

4. Interkoneksi antarkomponen (bus)

   Adalah struktur & mekanisme untuk menghubungkan pemroses, memori utama, & perangkat masukan/keluaran.



Proses

Pemroses disebut CPU, berfungsi mengendalikan operasi komputer dan melakukan pengolahan data.

Pemroses melakukan kerja dengan langkah sbb:

1. Mengambil instruksi yang dikodekan secara biner dari memori utama
2. Men-dekode instruksi menjadi proses-proses sederhana
3. Melaksanakan proses-proses tersebut

Operasi-operasi pada pemroses dikategorikan menjadi:

1. Operasi aritmetika
   Penambahan, pengurangan, perkalian, pembagian dsb
2. Operasi logika
   OR, AND, X-OR, inversi dsb
3. Operasi pengendalian
   Operasi percabangan, lompat dsb

Pemroses terdiri dari tiga komponen, yaitu:

1. CU (Control Unit)
   Berfungsi mengendalikan operasi yang dilaksanakan sistem komputer

2. ALU (Aritmetic Logic Unit)
   Berfungsi melakukan operasi aritmatika dan logika

3. Register

Merupakan memori yang sangat cepat yang berfungsi sebagai tempat operan-operan dari operasi yang akan dilakukan oleh pemroses.

Register

Register atau yang disebut dengan memori adalah suatu rangkaian logika yang mampu menyimpan data dalam bentuk bilangan biner. Fungsi dar i register ini selain sebagai penyimpanan data juga untuk menghindari berkedipnya angka yang ditunjukkan oleh display (seven segment) pada saat menerima pulsa-pulsa yang diberikan oleh decoder.

Sebuah register geser dapat memindahkan bit-bit yang tersimpan ke kiri atau ke kanan. Register geser dikelompokkan sebagai urutan rangkaian logika, oleh karena itu register geser disusun dari rangkain Flip-Flop. Selain untuk pergeseran data, register geser juga dapat digunakan untuk mengubah data seri ke paralel atau dari data parallel ke seri.

Skema blok Pemroses



Register dapat dikategorikan menjadi 2 :

1. Register yg terlihat pemakai (pemrogram)

Pemrogram dapat memeriksa isi dari register-register tipe ini. Beberapa instruksi disediakan untuk mengisi (memodifikasi) register tipe ini. Terdiri dari 2 jenis :

1.1 Register Data : menyimpan suatu nilai untuk beragam keperluan

1.1.1 General purpose register

Digunakan untuk beraneka ragam keperluan pada suatu instruksi mesin yang melakukan suatu operasi terhadap data.

1.1.2 Special purpose register

Digunakan untuk menampung operasi floating point, menampung limpahan operasi penjumlahan atau perkalian.

1.2 Register Alamat : berisi alamat data di memori utama, alamat instruksi di memori utama,bagian alamat yang digunakan dalam penghitungan alamat lengkap

1.2.1 Register Indeks (index register)

Pengalamatan berindeks merupakan salah satu mode pengalamatan popular. Pengalamatan melibatkan penambahan indeks ke nilai dasar untuk memperoleh alamat efektif

1.2.2 Register penunjuk segmen (segment pointer register)

Pada pengalamatan bersegmen, memori dibagi menjadi segmen-segmen. Segmen berisi satu blok memori yang panjangnya dapat bervariasi.Untuk mengacu memori bersegmen digunakan pengacuan terhadap segmen dan offset di segmen itu. Register penunjuk segmen mencatat alamat dasar (lokasi awal) dari segmen. Mode pengalamatan bersegmen sangat penting dalam manajemen memori.

1.2.3 Register penunjuk stack (stack pointer register)

Instruksi yang tak memerlukan alamat karena alamat operan ditunjuk register penunjuk stack. Operasi-operasi terhadap stack :

- instruksi push : menyimpan data pada stack, dengan meletakkan data di puncak stack
- instruksi pop : mengambil data dari puncak stack.

1.2.4 Register penanda (flag register)

Isi register merupakan hasil operasi dari pemroses. Register berisi kondisi-kondisi yang dihasilkan pemroses berkaitan dengan operasi yang baru saja dilaksanakan. Register ini terlihat oleh pemakai tapi hanya dapat diperbaharui oleh pemroses sebagai dampak (efek) operasi yang dijalankannya.

2. Register untuk kendali & status

Digunakan untuk mengendalikan operasi pemroses, kebanyakan tidak terlihat oleh pemakai. Sebagian dapat diakses dengan instruksi mesin yang dieksekusi dalam mode kontrol atau kernel sistem operasi.

2.1 Register untuk alamat dan buffer, terdiri dari :

2.1.1 MAR (Memory Address Register)

Untuk mencatat alamat memori yang akan diakses (baik yang akan ditulisi maupun dibaca)

2.1.2 MBR (Memory Buffer Register)

Untuk menampung data yang akan ditulis ke memori yang alamatnya ditunjuk MAR atau untuk menampung data dari memori (yang alamatnya ditunjuk oleh MAR) yang akan dibaca.

2.1.3 I/O AR (I/O Address Register)

Untuk mencatat alamat port I/O yang akan diakses(baik akan ditulisi / dibaca).

2.1.4 I/O BR (I/O Buffer Register)

Untuk menampung data yang akan dituliskan ke port yang alamatnya ditunjuk I/O AR atau untuk menampung data dari port (yang alamatnya ditunjuk oleh I/O AR) yang akan dibaca.

2.2 Register untuk eksekusi instruksi

2.2.1 PC (Program Counter) : mencatat alamat memori dimana instruksi di dalamnya akan dieksekusi

2.2.2 IR (Instruction Register) : menampung instruksi yang akan dilaksanakan

2.3 Register untuk informasi status

Register ini berupa satu register / kumpulan register. Kumpulan register ini disebut PSW (Program Status Word). PSW berisi kode-kode kondisi pemroses ditambah informasi-informasi status lain, yaitu :

♦ Sign

Flag ini mencatat tanda yang dihasilkan operasi yang sebelumnya dijalankan

♦ Zero

Flag ini mencatat apakah operasi sebelumnya menghasilkan nilai nol

♦ Carry

Flag ini mencatat apakah dihasilkan carry (kondisi dimana operasi penjumlahan/

perkalian menghasilkan bawaan yang tidak dapat ditampung register akumulator)

♦ Equal

Flag ini mencatat apakah operasi menghasilkan kondisi sama dengan

♦ Interupt enable/disable

Flag ini mencatat apakah interrupt sedang dapat diaktifkan atau tidak

♦ Supervisor

Flag ini mencatat mode eksekusi yang dilaksanakan, yaitu mode supervisor atau bukan. Pada mode supervisor maka seluruh instruksi dapat dilaksanakan sedang untuk mode bukan mode supervisor(mode user) maka beberapa instruksi kritis tidak dapat diaktifkan.

A. Memory

Memori berfungsi untuk menyimpan data dan program. Hirarki memori berdasarkan kecepatan akses :

Register
Memori case (Chace Memory)
Memori kerja (Main Memory)
Disk Magnetik (Magnetic Disk)
Disk Optik (Optical Disk)
Tape Magnetik (Magnetic Tape

Menurut urutan dari atas ke bawah dapat diukur hirarki dalam hal :

Harga : semakin ke bawah, harga semakin murah, harga dihitung dari rasio rupiah per bit data disimpan
Kapasitas : semakin ke bawah, kapasitas makin terbatas
Kecepatan akses : semakin ke bawah, semakin lambat
Frekuensi pengaksesan : semakin ke bawah, semakin rendah frekuensi pengaksesan

Setiap kali pemroses melakukan eksekusi, pemroses harus membaca instruksi dari memori utama. Agar eksekusi dilakukan secara cepat maka harus diusahakan instruksi tersedia di

memori pada lapisan berkecepatan akses lebih tinggi. Kecepatan eksekusi ini akan meningkatkan kinerja sistem.

Konsep ini diimplementasikan antara lain berupa :

Chace memory

Merupakan memori berkapasitas terbatas, berkecepatan tinggi yang lebih mahal dibanding memori utama. Chace memory adalah di antara memori utama dan register pemroses yang berfungsi agar pemroses tidak langsung mengacu memori utama tetapi di chace memory yang kecepatan akses lebih tinggi. Metode ini akan meningkatkan kinerja sistem.

Buffering

Bagian memori utama untuk menampung data yang akan ditransfer dari / ke perangkat masukan / keluaran dan penyimpan sekunder. Buffering dapat mengurangi frekuensi pengaksesan dari/ke perangkat masukan/keluaran dan penyimpan sekunder sehingga meningkatkan kinerja sistem.

B. Perangakat Masukan Dan Keluaran

C.1. Macam- Macam Perangakat masukan Dan Keluaran

Perangkat masukan/keluaran terdiri dari 2 bagian :

1. Komponen mekanis : perangkat itu sendiri
2. Komponen elektronis : pengendali perangkat berupa chip controller

Pengendali perangakat (Device Adapter) terdapat dua macam alat pengendali yaitu:

Perangkat adalah perangkat nyata yang dikendalikan chip controller di board system atau card.

Controller dihubungkan dengan pemroses dan komponen-komponen lain lewat bus. Controller berbeda-beda, tapi biasanya mempunyai register-register untuk mengendalikannya.

C.2 Sruktur I/O

1. I/O interrupt yaitu I/O kecapatan rendah
2. Struktur DMA yaitu I/O kecapatan tinggi

DMA dibagi menjadi : Third Party DMA dan frist party DMA

D. Interkoneksi Antar Komponen

Disebut BUS dan interkoneksi ini berkaitan dengan tatacara hubungan antarkomponen-komponen sistem komputer.

Bus terdiri dari tiga macam, yaitu:

1. Bus alamat (addres bus) : Untuk memberikan alamat dari memori atau port yang hendak diakses. Bus alamat berisi 16, 20, 24 jalur sinyal paralel atau lebih.
2. Bus data (data bus): Untuk membaca dan mengirim data dari/ke memori atau port. Bus data berisi 8,16, 32 jalur sinyal paralel atau lebih.
3. Bus kendali (control bus)
   Sinyal bus kendali antara lain:
   - Memory Read
   - Memory Write
   - I/O read
   - I/O Write

Mekanisme pembacaan :

Untuk membaca data suatu alokasi memori, CPUmengirim alamat memori yang dikehendaki melalui busalamat kemudian mengirim sinyal memory read pada bus kendali. Sinyal memory read memerintahkan keperangkat memori untuk mengeluarkan data padalokasi tersebut ke bus data agar dibaca CPU.

Interkoneksi antar komponen membentuk jenis koneksitas yang populer antara lain : ISA, EISA, MCA, VESA, PCI dan AGP.

Tingkatan Konsep Komputer

a. Tingkat Konsep Elektronika

Bentuk komputer terdiri atas sejumlah rangkaian komponen elektronika ditambah dengan komponen mekanika, magnetika dan optika.

b. Tingkat Konsep Rangkaian Saklar

Sudah dapat terlihat rangkaian elektronika yangsesungguhnya, yang membentuk banyak saklar yangtersusun secara paralel dan membentuk sekelompoksaklar. (terhubung dan terputus).

c. Tingkat Konsep Transfer Register

Berbagai kelompok sakelar di dalam computer membentuk sejumlah register (Logika, aritmatika, akumulator, indeks, adress register dll)

d. Tingkat Konsep Arsitektur

Sejumlah register tersusun dalam suatu arsitektur tertentu. Prosesor, memory dan satuan komponen lainnya terhubung melalui galur (bus) penghubung.

e. Tingkat Konsep Diagram Blok Arsitektur komputer atau sistem komputer dapat dipetak-petakan ke dalam sejumlah blok (masukan,blok satuan, prosesor pusat, memori, dll.)

Kerja komputer

Kerja komputer pada tingkat konsep, antara lain :

Tingkat konsep diagram blok, berlangsung sebagailalu lintas informasi di dalam dan diantara blokpada sistem komputer

Tingkat transfer register, kerja komputerberlangsung melalui pemindahan rincian informasidi antara register.

Tingkat konsep saklar, kerja komputer berlangsungdalam bentuk terputus dan terhubungnya berbagaisaklar eletronika di dalam sistem komputer.

Kerja komputer pada fungsi komputer, terdiri atas :kegiatan masukan, catatan, pengolahan dan keluaran

Kerja komputer pada rekaman

Sekelompok satuan data direkam ke dalam alatperekaman dalam bentuk berkas data.

Tataolah direkam ke dalam alat perekam danmembentuk berkas tataolah

Eksekusi instruksi

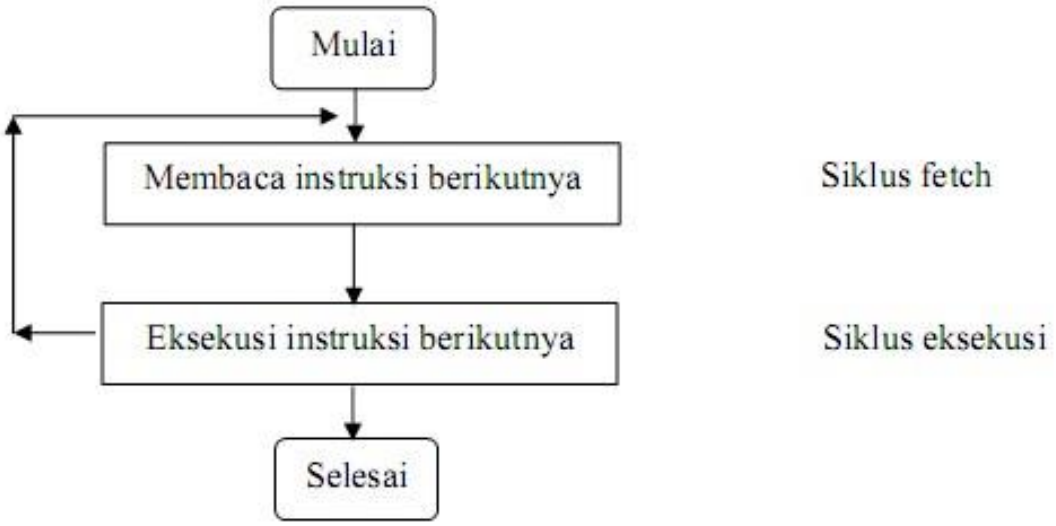
Tahap pemrosesan instruksi :

1. Pemroses membaca instruksi dari memori (fetch)

2. Pemroses mengeksekusi instruksi (execute)

Eksekusi program berisi pengulangan fetch dan execute. Pemrosesan 1 instruksi disebut satu siklus

instruksi. Siklus eksekusi instruksi :



Mode eksekusi instruksi

1. Mode pemakai (user mode)

Mode dengan kewenangan rendah, program pemakai (aplikasi) biasa dieksekusi dalam mode ini.

2. Mode sistem (system mode)

Mode dengan kewenangan tinggi. Biasanya rutin sistem atau kendali atau kernel dieksekusi dengan mode ini.

E. Perangkat Lunak (Software)

Merupakan komponen non fisik berupa kumpulanprogram beserta struktur datanya.Program adalah Sekumpulan instruksi yang disusun sedemikian rupa untuk dapat menyelesaikan masalah-masalah tertentu sesuai dengan kebutuhan.Siklus Instruksi, Untuk memproses instruksi dilakukan melalui 2 tahap :

mengambil instruksi (instruction fetch)
eksekusi instruksi (instruction execution)
Interrupt
Trap

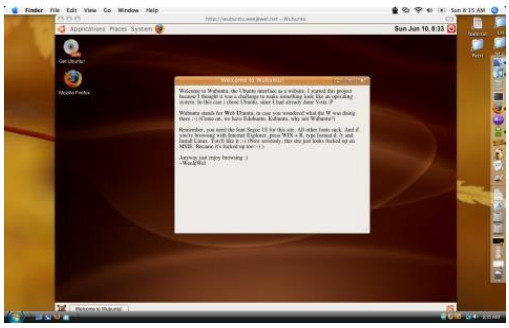
Gambar. Susunan Hirarki Perangkat Lunak



Sistem Operasi / OS.

Sistem Operasi komputer merupakan software pada lapisan pertama yang diletakkan pada memori komputer, (memori komputer dalam hal ini ada Hardisk, bukan memory ram) pada saat komputer dinyalakan. Sedangkan software-software lainnya dijalankan setelah Sistem Operasi Komputer berjalan, dan Sistem Operasi akan melakukan layanan inti umum untuk software-software itu. Layanan inti umum tersebut seperti akses ke disk, manajemen memori, skeduling task, dan antar-muka user. Sehingga masing-masing software tidak perlu lagi melakukan tugas-tugas inti umum tersebut, karena dapat dilayani dan dilakukan oleh Sistem Operasi. Bagian kode yang melakukan tugas-tugas inti dan umum tersebut dinamakan dengan kernel suatu Sistem Operasi.

Sistem Operasi berfungsi sebagai penghubung antara lapisan hardware dan lapisan software. selain itu, melakukan semua perintah perintah penting dalam komputer, serta menjamin aplikasi-aplikasi yang berbeda fungsinya dapat berjalan lancar secara bersamaan tanpa hambatan. *Sistem Operasi Komputer* menjamin aplikasi perangkat lunak lainnya bisa memakai memori, melakukan input serta output terhadap peralatan lain, dan mempunya akses kepada sistem file. Jika beberapa aplikasi berjalan secara bersamaan, maka Sistem Operasi Komputer akan mengatur jadwal yang tepat, sehingga sebisa mungkin semua proses pada komputer yang berjalan mendapatkan waktu yang cukup untuk menggunakan CPU dan tidak saling mengganggu dengan perangkat yang lain.



Antar muka linux Ubuntu



Antar muka sistem operasi Mac OS X



Tujuan sistem operasi:

1. Agar dapat merancang sendiri
2. Dapat memodifikasi sistem yang telah ada sehingga sesuai dengan kebutuhan
3. Dapat memilih di antara berbagai alternatif sistem operasi
4. Memaksimalkan penggunaan sistem operasi
5. Konsep dan teknik sistem operasi dapat diterapkan pada aplikasi-aplikasi lain.

Sistem operasi sebagai resource manager yaitu pengelola seluruh sumber daya yang terdapat pada sistem komputer dan sebagai extended machine yaitu menyediakan sekumpulan layanan ke pemakai sehingga memudahkan dan menyamankan penggunaan serta pemanfaatan sumber daya sistem komputer.

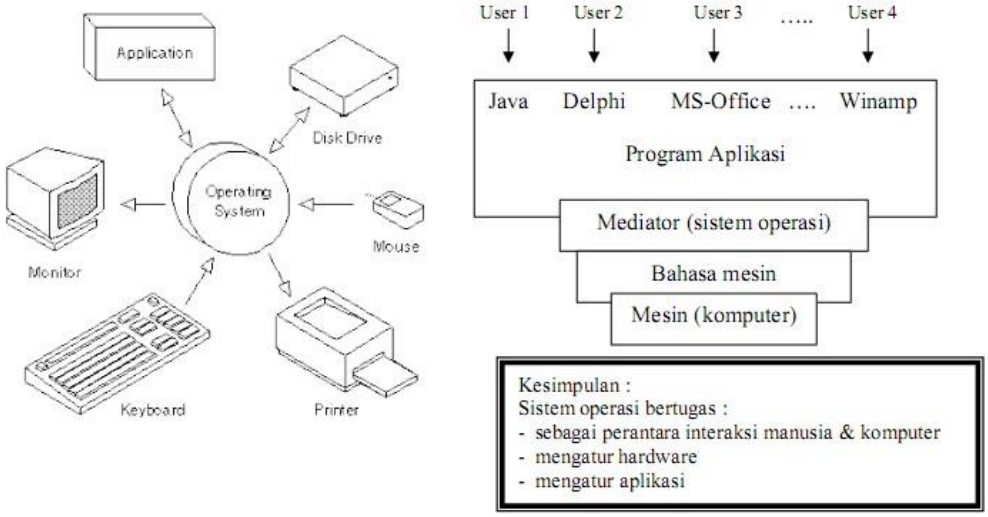
Sistem operasi merupakan program komputer yang berisi perintah-perintah (command) dan bertugas menjembatani pengertian manusia dengan komputer, sehingga komputer dapat bekerja sesuai keinginan.

Definisi sistem operasi :

1. Sistem operasi adalah software yang mengontrol hardware.
2. Program yang menjadikan hardware lebih mudah untuk digunakan
3. Kumpulan program yang mengatur kerja hardware sesuai keinginan user

4. Manager sumber daya atau pengalokasian sumber daya komputer, seperti mengatur memori, printer, dll
5. Sebagai program pengendali, yaitu program yang digunakan untuk mengontrol program yang lain
6. Sebagai kernel, yaitu program yang terus-menerus running selama computer dihidupkan
7. Sebagai guardian yang menjaga komputer dari berbagai kejahatan komputer.

3.



Brainware

Brainware adalah orang yang mengoperasikan sebuah komputer, karena jika tidak ada orang yang mengoperasikan maka tidak akan dapat digunakan.

Daftar Pustaka

1. Ibam, Sistem Operasi, Gabungan Kelompok Kerja IKI Semester Genap 2002/2003, www.Ilmukomputer.com, 2003.
2. Sri Kusuma Dewi, Sistem Operasi, Graha Ilmu, 2002.
3. William Stallings, Sistem Operasi Edisi Empat, PT. Indeks, Kelompok Gramedia, 2005.

MODUL PERKULIAHAN

PROSES dan THREAD

Proses merupakan unit kerja terkecil yang secara individu memiliki sumber daya dan dijadwalkan sistem operasi.

Abstract

Proses berisi instruksi, data, program counter, semua register pemroses, dan stack bersis data sementara.

Kompetensi

Memahami hal – hal yang berkaitan dengan proses: Memahami tentang Operasi – operasi terhadap proses Memahami tentang Implementasi Proses

.

PROSES dan THREAD

Proses - Proses

Proses berisi instruksi dan data, program counter dan semua register pemroses, dan stack bersis data sementara. Proses merupakan unit kerja terkecil yang secara individu memiliki sumber daya dan dijadwalkan sistem operasi. Hal – hal yang berkaitan dengan proses:

- *multiprogramming*:

  Manajemen banyak proses pada satu pemroses, contoh komputer pribadi bersifat workstation artinya sistem pemroses tunggal yang dapat menjalankan banyak program/task seperti MS-Windows 3.0, MS-Windows NT, OS/2 dan Macintosh system 7. Program yang dijalankan sebenarnya bersifat independent dan one program at any instant (satu program pada satu saat).

- *Multiprocessing:*

  Manajemen banyak proses dalam satu komputer, contoh pada sistem operasi microsoft windows NT, UNIX, Linux telah menyediakan fasilitas multiprocessing.

- *Distributed processing*:

  Manajemen banyak proses yang dieksekusi di banyak sistem komputer yang tersebar.

Kebutuhan utama pengendali proses:

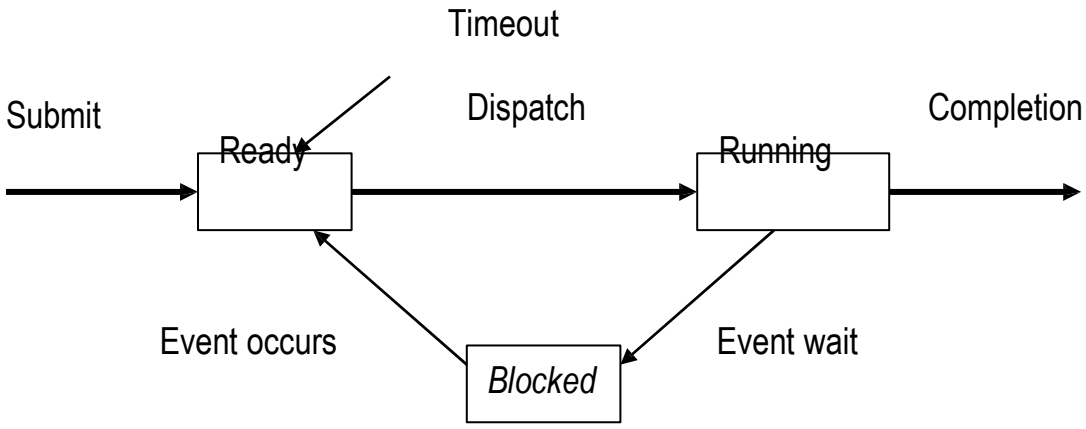
- Saling melanjutkan (interleave): bertujuan memaksimumkan penggunaan pemroses sambil memberi waktu tanggap.
- Mengikuti kebijaksanaan tertentu: SO harus mengalokasikan sumber daya ke proses mengikuti kebijaksaan yang sudah ditentukansambil menghindari *deadlock*.
- Mendukung komunikasi antar proses dan pencipta proses: SO harus dapat mengetahui state masing – masing proses dan merekam semua perubahan yang terjadi secara dinamis (berguna untuk penjadwalan dan memutuskan alokasi sumber daya).

6. Diagram state proses

c. Diagram state dasar terdiri dari 3 keadaan : proses akan berada pada salah satu dari tiga state :

Status	**Deskripsi**
Running	Pemroses sedang mengeksekusi intruksi proses
Ready	Proses siap dieksekusi tapi pemroses tidak tersedia untuk eksekusi
Blocked	Proses menunggu kejadian untuk melengkapi tugasnya seperti : - selesainya operasi perangkat I/O - tersedianya memori - datangnya pesan

Hubungan ketiga state tersebbut dapat digambarkan:



Keterangan:

- Proses yang baru diciptakan akan segera mempunyai state ready
- Proses running menjadi *blocked* karena sumber daya yang diminta belum tersedia, proses menunggu alokasi sumber daya (event wait)
- Proses running menjadi ready karena penjadwalan memutuskan eksekusi proses lain karena jatah waktu telah habis (time out)
- Proses *blocked* menjadi ready saat sumber daya yang diminta tersedia

- Proses ready menjadi running karena penjadwal memutuskan penggunaan pemroses untuk proses itu.

7. PCB (*Program Control Block*)

Sistem operasi memerlukan banyak informasi mengenai proses guna pengelolaan proses, informasi ini berada di PCB. Dalam PCB informasi dibagi 3 kelompok:

d. Informasi identifikasi proses:
   Identifikasi berkaitan dengan proses yang unik. Identifier numerik yang meliputi : prose situ sendiri, siapa proses yang menciptakan dan yang pemakai

e. Informasi status pemroses:
   Berisi register–register pemroses. Jumlah dan jenis register tergantung arsitek komputer. Register tersebut terdiri:

   - register yang terlihat pemakai : dapat ditunjuk instruksi bahasa assembly untuk diproses pemroses
   - register kendali dan status : untuk mengendalikan operasi pemroses seperti: program counter, PSW dsb.
   - Pointer stack: untuk parameter atau alamat prosedur pemanggil dan system call, pointer stack menujuk posisi paling atas dari stack.

f. Informasi Kendali proses yang terdiri atas :
   - informasi penjadwalan dan status : untuk menjalankan fungsi penjadwalan : status proses (running, ready, *blocked* dsb), prioritas (lama menunggu, lama proses dsb) dan kejadian (identitas kejadian yang ditunggu proses)
   - penstruktur data : satu proses dapat dikaitkan dengan proses lain dalam satu antrian atau ring atau struktur lainnya.
   - Komunikasi antar proses: beragam *flag*, sinyal dan pesan dapat diasosiasikan dengan komunikasi antara dua proses yang terpisah.
   - Kewenangan proses : kewenangan berkaitan dengan memori dan tipe instruksi yang dapat dijalankan.
   - Manajemen memori : berisi pointer ke tabel segmen atau page yang menyatakan memori maya proses.

- Kepemilikan dan utilisasi sumber daya : sumber daya yang dikendalikan proses harus diberi tanda : berkas dibuka, pemakaian pemroses dan pemakaian sumber daya lainnya.

8. Operasi – operasi pada proses

Operasi – operasi terhadap proses antara lain:

- penciptaan proses (create a process)
- penghancuran proses (destroy a process)
- penundaan proses (suspend a process)
- pelanjutan kembali proses (resume a process)
- pengubahan prioritas process
- memblok proses
- membangunkan proses
- menjadwal proses
- memungkinkan proses berkomunikasi dengan proses lain

Penciptaan proses melibatkan banyak aktivitas seperti : menamai, menyisipkan, menentukan prioritas, menciptakan PCB dan mangalokasikan sumber daya proses.Kejadian yang dapat menciptakan proses :

- pada lingkungan batch sebagai tanggapan atas job
- pada lingkungan interaktif, terjadi saat pemakai berusaha log on
- sebagai tanggapan suatu aplikasi
- adanya penciptaan proses lain (anak proses)

Pada UNIX : proses diciptakan dengan *system call fork* : menciptakan kopian proses pemanggil (induk) yang identik.

Pada MS-DOS: *system call* adalah memuatkan file biner ke memori dan mengeksekusi sebagai anak proses. Pada sistem ini proses induk dan proses anak tidak berjalan paralel.

Penghancuran proses: pada penghancuran proses maka akan melibatkan kegiatan pembebasan proses dari system, antara lain :

- sumber daya yang dipakai dikembalikan
- proses dihancurkan dari tabel sistem
- PCB dihapus

Penghancuran akan lebih rumit bila telah menciptakan proses. Untuk kondisi tersebut maka akan dilakukan pendekatan, antara lain:

- pada beberapa sistem proses-proses turunan dihancurkan saat proses induk dihancurkan secara otomatis
- menganggap proses anak independen terhadap proses induk

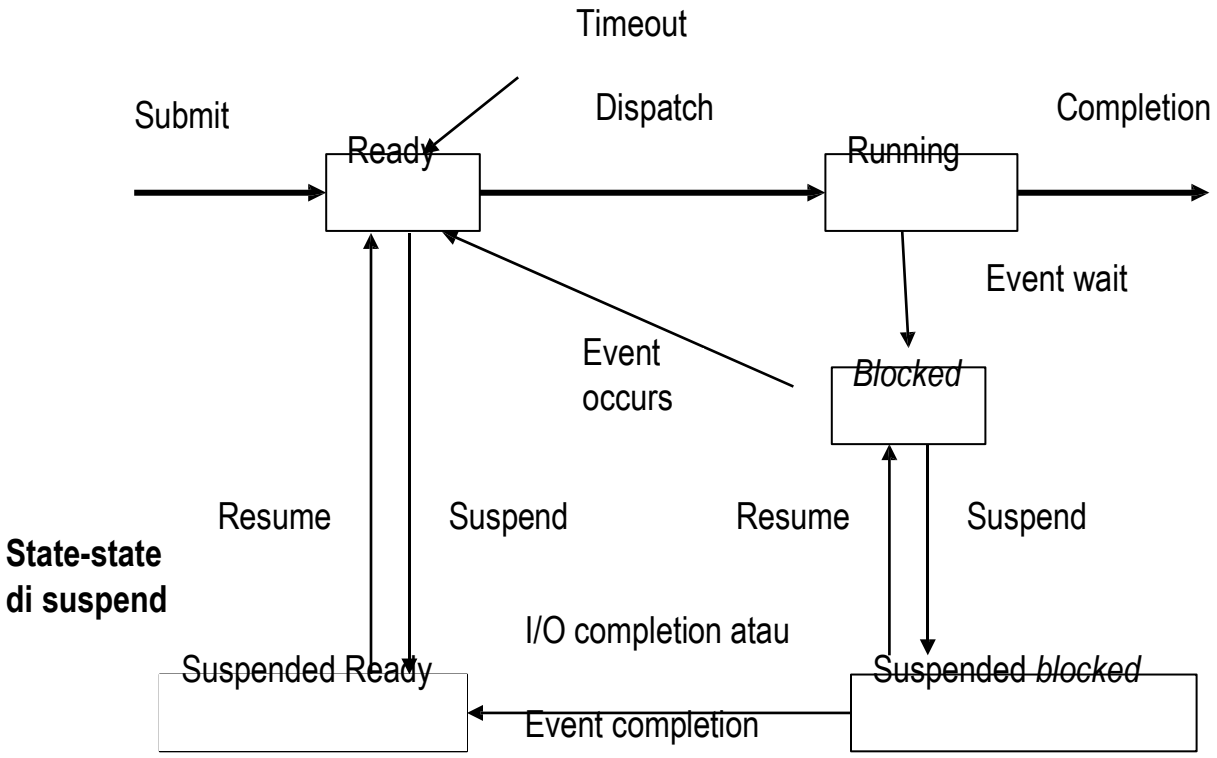
Alasan – alasan penghancuran proses:

Penyebab Terminasi	Deskripsi
Selesainya proses secara normal	Proses telah berjalan secara lengkap
Batas waktu telah terlewati	Proses telah melewati batas waktu total yang telah dispesifikasikan
Memori tidak tersedia	Keperluan memori lebih banyak daripada yang diesiakan sistem
Pelanggaran terhadap batas memori	Mencoba mengakses lokasi memori yang tidak diijinkan diakses
Terjadi kesalahan karena pelanggaran proteksi	Mencoba menggunakan sumber daya yang tidak diijinkan
Terjadi kesalahan aritmatika	Mencoba perhitungan terlarang, seperti pembagian dengan nol
Waktu telah kadaluwarsa	Telah lama menunggu daripada waktu maksimum yang ditentukan
Instruksi yang tak benar	Berusaha mengeksekusikan instruksi yang tak ada seperti pencabangan

Terjadi usaha memakai instruksi yang tak diijinkan	Menggunakan instruksi yang disimpan untuk sistem operasi
Terjadi kegagalan I/O	Kesalahan muncul pada masukan atau keluaran
Kesalahan pengguna data	Ada data yang tipenya tidak terinisialisasi
Diintervensi oleh sistem operasi atau operator	Karena alasan tertentu operator mengakhiri proses
Berakhirnya proses induk	Ketika parent berakhir, sistem akan mengakhiri semua anak proses
Atas permintaan dari proses induk	Parent proses biasanya mempunyai otoritas mengakhiri anak proses

Pada UNIX penghancuran proses dengan memanggil system call exit( ). Pada MS-DOS terdapat terminate program.

9. Diagram state lanjut (lima keadaan)



10. Implementasi Proses

Setiap proses mempunyai state yang perlu diperhatikan sistem operasi yang dicatat dalam tabel :

e. Tabel informasi manajemen memori untuk menjaga keutuhan memori utama dan memori skunder, yang memuat : alokasi memori utama yang dipakai pemroses, alokasi memori skunder yang terpakai, atribut segmen utama dan sekunder dan informasi lain yang digunakan untuk pengelolaan memori.

f. Tabel informasi manajemen masukan /keluaran : mengelola perangkat I/O dan mengetahui statusnya dan lokasi memori utama yang digunakan transfer data

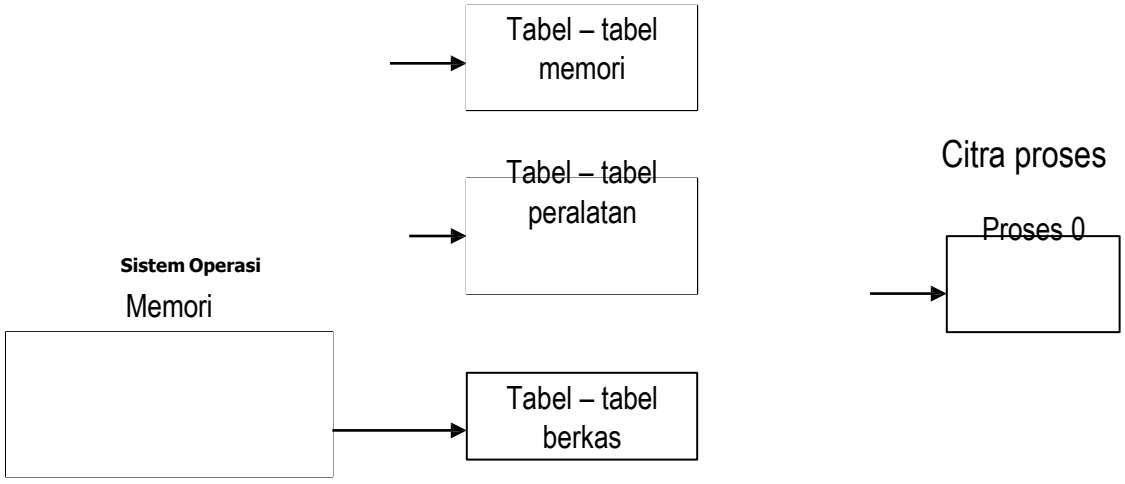
g. Tabel sistem informasi : berisi informasi eksetensi file, lokasi pada memori sekunder, statusnya dan informasi lainnya.

h. Tabel proses mengelola informasi proses di sistem operasi, lokasinya memori, status dan atribut lainnya.

Implementasi Proses

Elemen citra proses	Deskripsi
Data pemakai	Bagian dapat dimodifikasi seperti data program, daerah stack pemakai
Program pemakai	Program biner yang dieksekusi
Stack sistem	Menyimpan parameter dan alamat pemanggilan untk prosedur dan system call
PCB	Informasi yang diperlukan sistem operasi dalam mengendalikan proses

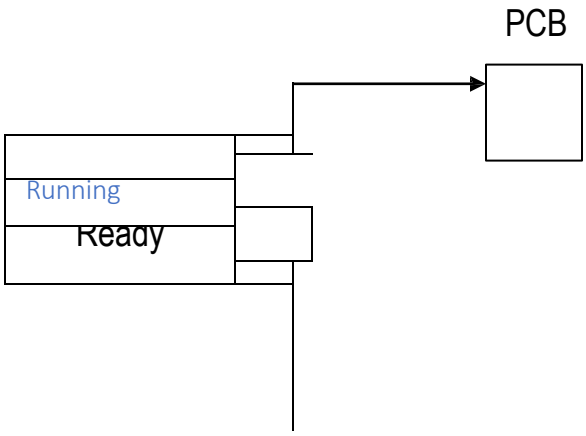
Struktur umum tabel kendali pada sistem operasi.



PCB dan senarai proses.

PCB dapat dibaca dan atau dimodifikasi rutin sistem operasi seperti penjadwalan, alokasi sumber daya, pemroses interupsi, monitoring dan analisis kinerja.

Untuk menyatakan senarai proses di sistem operasi dibuat senarai PCB



Pengaksesan informasi di PCB

Setiap proses dilengkapi denga iD yang unik digunakan sebagai indeks ke tabel untuk mengambil PCB, kesulitan yang terjadi disebabkan oleh proteksi terhadap PCB, dua masalah utama proteksi PCB :

c. Bug (kesalahan) pada rutin tunggal misalnya interrupt handler dapat merusak PCB
d. Perubahan rancangan struktur dan semantiks PCB

Kedua masalah tersebut diatasi dengan rutin penanganan PCB dalam pengaksesan. Tugas rutin ini adalah memproteksi PCB dan menjadi perantara pembacaan dan penulisan PCB. Sehingga masalah diatas dapat diatasi karena :

c. Kesalahan dapat dicegah sehingga PCB tidak rusak
d. Antar muka terhadap rutin – rutin lain masih dapat dijaga meskipun rincian PCB diubah.

Daftar Pustaka

4. Ibam, Sistem Operasi, Gabungan Kelompok Kerja IKI Semester Genap 2002/2003, www.Ilmukomputer.com, 2003.
5. Sri Kusuma Dewi, Sistem Operasi, Graha Ilmu, 2002.
6. William Stallings, Sistem Operasi Edisi Empat, PT. Indeks, Kelompok Gramedia, 2005.

MODUL PERKULIAHAN

PROSES PENJADUALAN DI PROSESOR

Tipe penjadualan di prosesor Non Preemptive.

Abstract

Penjadwalan untuk memutuskan proses mana yang harus berjalan dan kapan serta berapa lama waktu yang diperlukan di sistem operasi.

.

Kompetensi

Mampu memahami hal – hal yang berkaitan dengan proses penjadualan di prosesor. Mampu memahami Algoritma pada trategi *non-preemptive.*

.

PROSES PENJADWALAN di PROSESOR

1. Deskripsi penjadwalan proses.

Penjadwalan merupakan kumpulan kebijaksanaan dan mekanisme di sistem operasi yang berkaitan dengan urutan kerja yang sering dilakukan di sistem komputer. Tujuan untuk memutuskan proses mana yang harus berjalan dan kapan serta berapa lama waktu yang diperlukan.

Kriteria yang digunakan untuk optimasi kinerja penjadwalan proses :

a. Adil (fairness): setiap proses mendapat jatah waktu yang sama.
b. Efiesiensi: menjaga agar pemroses dalam keadaan sibuk sehingga efisiensinya maksimum.
c. Waktu tanggap (response time): meminimalkan waktu tanggap. Waktu tanggap ada 2 yaitu waktu tanggap interaktif (terminal response time) dan waktu tanggap pada sistem nyata (event response time). Waktu tanggap interaktif adalah waktu yang dihabiskan dari saat karakter perintah terakhir berinteraksi sampai instruksi pertama rutin layanan dieksekusi. Waktu tanggap pada sistem nyata adalah waktu saat kejadian sampai instruksi pertama rutin layanan dieksekusi.
d. *Turn* around time: meminimalkan *turn* around time. *Turn* around time adalah waktu yang dihabiskan dari saat program masuk ke sistem sampai proses diselesaikan sistem. Dengan kata lain *turn* around time = waktu eksekusi + waktu tunggu.
e. Throughput: memaksimalkan jumlah job yang diproses per satuan waktu. Throughput adalah jumlah kerja yang dapat diselesaikan persatuan waktu.

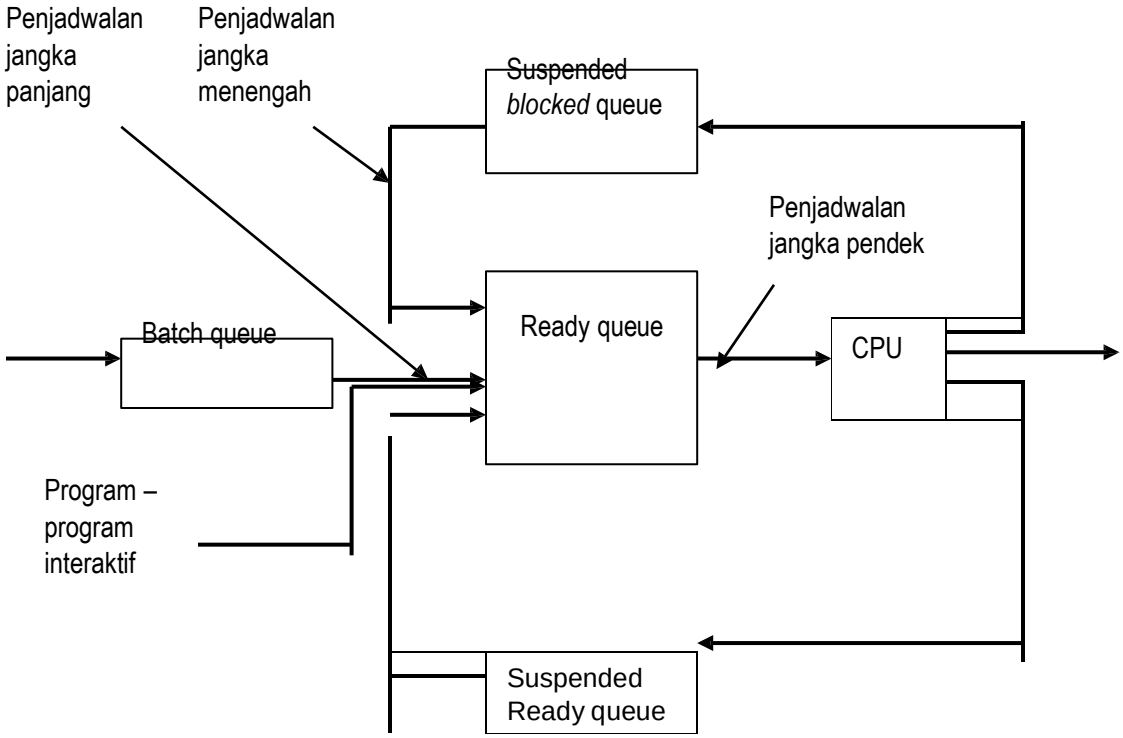
2. Type Penjadwalan

Terdapat 3 tipe penjadwalan pada sistem operasi yang kompleks :

a. Penjadwalan jangka pendek (short term scheduler) bertugas menjadwalkan alokasi pemroses diantara proses – proses ready di memori utama. Sasarannya memaksimumkan kinerja untuk memenuhi satu kumpulan kriteria.

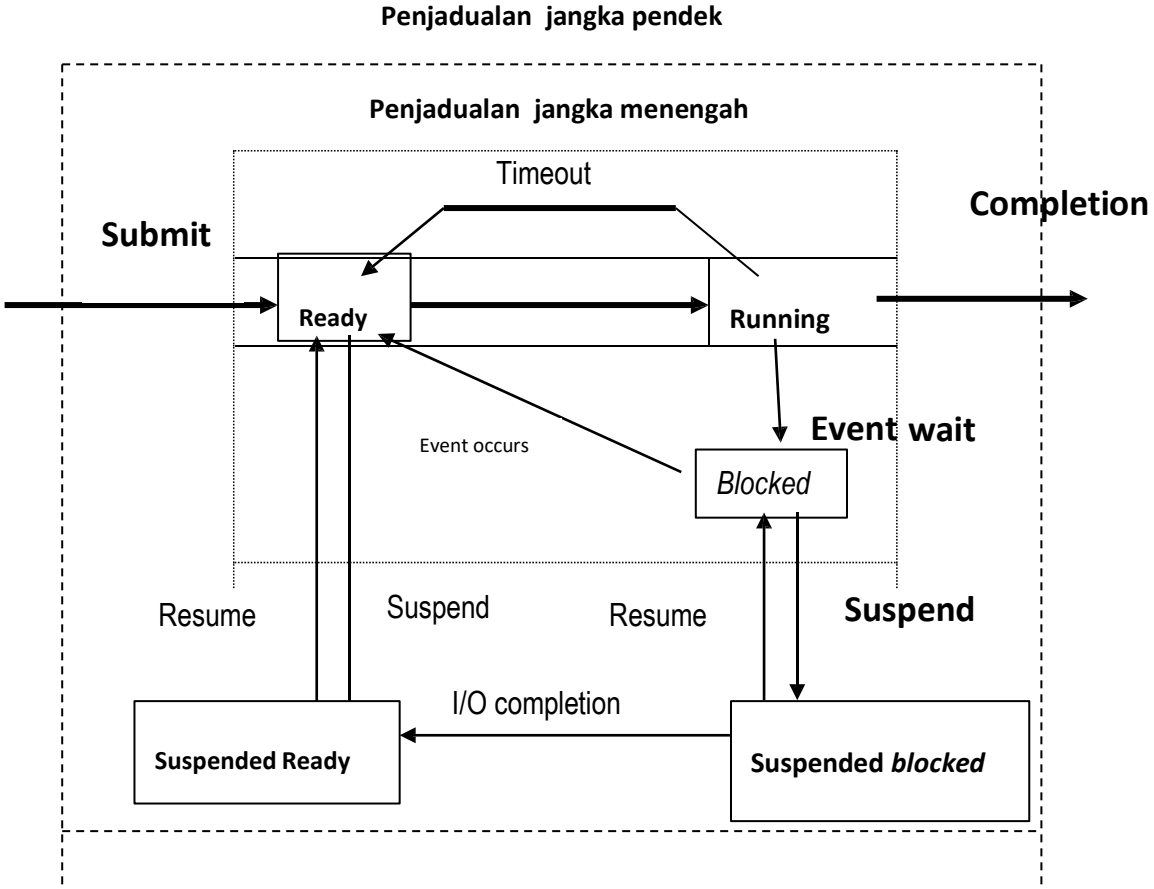
b. Penjadwalan jangka menengah (medium term scheduler) bertugas menangani proses – proses swapping, mengendalikan transisi dari suspended to ready proses-proses swapping. Swapping adalah pemindahan proses yang tertunda dari memori utama ke memori sekunder.

c. Penjadwalan jangka panjang (long term scheduler) bertugas terhadap antrian batch dengan sasaran memberi keseimbangan job – job campuran.



Tipe – tipe penjadwalan

Hubungan antara penjadwalan dengan state – state dalam proses dapat dilihat pada diagram berikut:



3. Strategi penjadwalan

Terdapat 2 strategi penjadwalan :

a. Penjadwalan *nonpreemptive (run-to-completion)* : apabila suatu proses telah diberi jatah maka tidak dapat diambilalih proses lain sampai proses tersebut selesai.

b. Penjadwalan *preemptive* : proses lain dapat mengambilalih jatah waktu proses lain. Ini sangat berguna pada sistem dimana proses memerlukan tanggapan dari pemroses lebih cepat. Misalnya pada sistem waktu nyata, kehilangan interupsi dapat berakibat fatal atau pada sistem interaktif time sharing, penjadwalan preemptive untuk menjamin waktu tanggap yang memadai.

4. Algoritma penjadwalan

Algoritma yang sering digunakan pada setrategi *non-preemptive* :

- FIFO (First-in, first out) atau FCFS (first come, first serve)

- SJF (Shortest Job First)
- HRN (Higest-ratio next)
- MFQ (Multiple Feedback Queues)

Algoritma yang sering digunakan dalam strategi *preemptive*:

- RR (Round robin)
- SRF (Shortest remaining first)
- PS (priority schedulling)
- GS (Guaranteed Schedulling)

Penjadwalan FIFO (First In First Out)

Penjadwalan paling sederhana, tidak berprioritas, proses diberi jatah berdasarkan waktu datangnya dan dijalankan sampai proses selesai. FIFO biasanya dikombinasikan dengan skema lainnya.

Berdasarkan kriteria :

Kriteria	**Penilaian**
Fairness	Adil dari semantik antrian
Efisiensi	Sangat efisiensi
Waktu tanggap	Sangat jelek, terutama untuk waktu nyata
Turnaround time	jelek
Throughput	jelek

Beberapa teknik penjadwalan sbb :

- Menggunakan algoritma (First Come First Served, FCFS atau FIFO).
- FIFO merupakan penjadwalan tanpa prioritas dan tanpa preemptive.
- Pada FIFO, proses yang tiba lebih dahulu akan dilayani lebih dahulu.

Perintah :

- Buatlah Gant Card
- Hitung waktu tanggap rata-rata (AWT = Average Waiting Time)

Untuk Penyelesaian :

- Buat diagram proses versus waktu
- Buat tabel untuk menghitung waktu tanggap

Contoh Kasus :

Dianggap semua proses sebagai satu kumpulan proses serentak dan hanya dilayani oleh satu prosesor. Table berikut menggambarkan dari beberapa proses yang akan dilayani oleh satu prosesor.

Nama Proses	Saat Tiba	Lama Proses
A	0	9
B	0	30
C	0	4
D	0	8
E	0	12

Penyelesaian buat Gant Card lalu hitung hitung rata-rata lama waktu tanggap untuk semua proses itu.

Nama proses	Saat Tiba	Lama Proses	Saat Mulai	Saat Rampung	Lama Tanggap
A	0	9	0	9	9
B	0	30	9	39	39
C	0	4	39	43	43
D	0	8	43	51	51
E	0	12	51	63	63
				Jumlah	205
				Rerata	41

Rerata lama tanggap = 41 satuan waktu

Untuk proses yang datangnya tidak bersamaan atau berbeda, contoh Yang Mengandung Saat Tiba Proses Beda.

Nama Proses	Saat tiba	Lama Proses
A	0	4
B	1	7
C	3	3
D	7	8

Di bawah ini dibuat tabel kerja prosesor.

Nama Proses	Saat Tiba	Lama Proses	Saat Mulai	Saat Rampung	Lama Tanggap
A	0	4	0	4	4
B	1	7	4	11	10
C	3	3	11	14	11
D	7	8	14	22	15
				Jumlah	40
				Rerata	10

Hasil perhitungan rerata lama tanggap = 10 satuan waktu.

Penjadwalan Waktu Proses Terpendek *(SJF= Shortest Job First).*

Tak berprioritas, jarang digunakan, menggunakan asumsi waktu jalan proses sudah diketahui, menjadwalkan proses dengan waktu terpendek dijalankan lebih dahulu, efisiensi tinggi dan time around time rendah. Masalah yang terjadi : tidak dapat mengetahui ukuran job saat job masuk dan untuk proses yang datang tidak bersamaan menggunakan penetapan dinamis.

Contoh : terdapat 4 proses yaitu proses A dengan waktu jalan 10 kwanta, B = 8, C = 6 dan D = 4. Dapat dibuat *turn* around time sebagai berikut :

Proses	*Turn around time*	
	Cara I	**Cara II**
A	10	28
B	18	18
C	24	10
D	28	4
Rata – rata	20	15

Disini ada 2 langkah , yaitu:

1. Penentuan prioritas berdasarkan pendeknya proses yang dilayani.
2. Penentuan pada saat tertentu, proses mana yang perlu dilayani procesor.

Contoh kasus

vvvv1. Proses tiba secara bersamaan dan dilayani oleh satu prosesor.

Nama Proses	Saat Tiba	Lama Proses
A	0	11
B	0	8
C	0	10
D	0	3
E	0	5

Gambar dibawah ini menunjukkan sekumpulan proses A ; B; C; D; E dengan saat tiba sama yaitu : 0 dengan lama proses berbeda.

Berdasarkan perbedaan tersebut maka langkah pertama adalah dilakukan penyusun prioritas.

Urutannya adalah sbb :

Nama Proses	Saat Tiba	Lama Proses	Saat Mulai	Saat Rampung	Saat Tanggap
D	0	3	0	3	3
E	0	5	3	8	8
B	0	8	8	16	16
C	0	10	16	26	26
A	0	11	26	37	37
				Jumlah	90
				Rerata	18

Tampak disini bahwa SJF mengakibatkan lama tanggap semua proses = 18 satuan waktu

Hasil ini lebih singkat dari pada rerata lama tanggap untuk FIFO.

2. Contoh Kumpulan proses yang mempunyai waktu tiba secara tidak bersamaan.

Nama Proses	Saat Tiba	Lama Proses
A	0	5
B	3	7
C	5	2
D	6	4

Proses berdasarkan prioritas dilihat dari lama proses yaitu : C; D; A; dan B.

Pelaksanaannya :

- Dimulai dari saat 0.
- Proses terpendek " C " belum tiba,
- Yang sudah tiba adalah " A ", inilah yang dikerjakan.
- Proses " A " rampung pada saat " 5 ".
- Proses terpendek sudah tiba, maka yang dikerjakan adl proses " C ", dimulai pada saat " 5 " dan rampung pada saat " 7 ".
- Setelah proses "C" rampung, proses "D" yang urutan kedua sudah tiba sehingga dpt dikerjakan oleh processor.mulai saat " 7 " dan rampung saat " 11".
- Proses B urutan akhir sudah tiba dan dikerjakan, mulai saat " 11 " dan rampung saat " 18 ". Di bawah ini dibuat barisan saat dan daftar proses.

Saat	0	1	2	3	4	5	6	7	8	9	10	11
proses	A	A	A	A	A	A	C	C	D	D	D	D

Saat	11	12	13	14	15	16	17	18
Proses		B	B	B	B	B	B	B

Nama Proses	Saat Tiba	Lama Proses	Saat Mulai	Saat Rampung	Lama Tanggap
A	0	5	0	5	5
B	3	7	11	18	15
C	5	2	5	7	2
D	6	4	7	11	5
				Jumlah	27
				Rerata	6,75

Keunggulan Penjadwalan SJF dibandingkan dengan FIFO:

- Kemampuannya memperkecil rerata lama tanggap.

Kelemahannya:

- Jika proses pendek tiba terus-menerus, maka layanan terhadap proses panjang tertunda.
- Sulit implementasi karena sulit untuk memprediksi waktu proses.

Daftar Pustaka

7. Ibam, Sistem Operasi, Gabungan Kelompok Kerja IKI Semester Genap 2002/2003, www.Ilmukomputer.com, 2003.
8. Sri Kusuma Dewi, Sistem Operasi, Graha Ilmu, 2002.
9. William Stallings, Sistem Operasi Edisi Empat, PT. Indeks, Kelompok Gramedia, 2005.

MODUL PERKULIAHAN

Tipe penjadualan di prosesor Preemptive

Penjadwalan berkaitan dengan permasalahan memutuskan proses mana yang akan dilaksanakan dalam suatu sistem. Proses yang belum mendapat jatah alokasi dari CPU akan mengantri di *ready queue*.

Abstract

suatu proses akan menggunakan CPU sampai proses tersebut dalam status *wait* (misalnya meminta I/O) atau selesai. Pada saat *wait* , maka CPU akan nganggur (*idle*). Untuk mengatasi hal ini, maka CPU dialihkan ke proses lain pada saat suatu proses sedang dalam *wait*, demikian seterusnya.

Kompetensi

Memahami konsep dasar dari multiprogramming.
Memahami tentang Penjadwalan prosesor.
Memahami tentang Algoritma Penjadualan.

Tipe penjadualan di prosesor Preepmtive.

Seperti halnya pada prosesor tunggal, prosesor jamak juga membutuhkan penjadwalan. Namun pada prosesor jamak, penjadwalannya jauh lebih kompleks daripada prosesor tunggal karena pada prosesor jamak memungkinkan adanya *load sharing* antar prosesor yang menyebabkan penjadwalan menjadi lebih kompleks namun kemampuan sistem tersebut menjadi lebih baik. Oleh karena itu, kita perlu mempelajari penjadwalan pada prosesor jamak berhubung sistem dengan prosesor jamak akan semakin banyak digunakan karena kemampuannya yang lebih baik dari sistem dengan prosesor tunggal. Ada beberapa jenis dari sistem prosesor jamak, namun yang akan dibahas dalam bab ini adalah penjadwalan pada sistem prosesor jamak yang memiliki fungsi-fungsi prosesor yang identik (*homogenous*).

PROSES TERPENDEK DIPERTAMAKAN (TPD)

1. Shortes Job First (SJF).
2. Shortes Job Next (SJN).
3. Shortes Process Next (SPN).

Disini ada 2 langkah , yaitu :

3. Penjadwalan PTD dengan penentuan prioritas berdasarkan pendeknya proses yang dilayani.
4. Penentuan pada saat tertentu, proses mana yang perlu dilayani procesor.

Contoh : PROSES SAAT TIBA SAMA.

Tabel di bawah ini menunjukkan sekumpulan proses A ; B; C; D; E dengan saat tiba sama yaitu : 0 . Namun lama proses berbeda.

Nama Proses	Saat Tiba	Lama Proses
A	0	11
B	0	8
C	0	10
D	0	3
E	0	5

Berdasarkan perbedaan ini,langkah pertama dilakukan penyusun prioritas.

Urutannya adl sbb :

Nama Proses	Saat Tiba	Lama Proses	Saat Mulai	Saat Rampung	Saat Tanggap
D	0	3	0	3	3
E	0	5	3	8	8
B	0	8	8	16	16
C	0	10	16	26	26
A	0	11	26	37	37
				Jumlah	90
				Rerata	18

Tampak disini bahwa PTD mengakibatkan lama tanggap semua proses = 18 satuan wkt

Hasil ini lebih singkat dgn rerata lama tanggap untuk PTPD.

Cth : Sekumpulan proses dgn saat tiba tidak sama (PTD).

Nama Proses	Saat Tiba	Lama Proses
A	0	5
B	3	7
C	5	2
D	6	4

Proses berdasarkan prioritas dilihat dari lama proses yaitu : C; D; A; dan B.

Pelaksanaannya :

1. Dimulai dari saat 0.
2. Proses terpendek " C " belum tiba,
3. Yang sudah tiba adalah " A ", inilah yang dikerjakan.
4. Proses " A " rampung pada saat " 5 ".
5. Proses terpendek sudah tiba, maka yang dikerjakan adl proses " C ", dimulai pada saat " 5 " dan rampung pada saat " 7 ".
6. Setelah proses " C " rampung, proses " D " yang urutan kedua sudah tiba sehingga dpt dikerjakan oleh processor.mulai saat " 7 " dan rampung saat " 11 ".
7. Proses B urutan akhir sudah tiba dan dikerjakan, mulai saat " 11 " dan rampung saat " 18 ".

Di bawah ini dibuat barisan saat dan daftar proses.

Saat	0	1	2	3	4	5	6	7	8	9	10	11
proses	A	A	A	A	A	C	C	D	D	D	D	

Saat	11	12	13	14	15	16	17	18
Proses	B	B	B	B	B	B	B	

Nama Proses	Saat Tiba	Lama Proses	Saat Mulai	Saat Rampung	Lama Tanggap
A B C D	0 3 5 6	5 7 2 4	0 11 5 7	5 18 7 11	5 15 2 5
				Jumlah Rerata	27 6,75

Keunggulan Penjadwalan PTD :

- Kemampuannya memperkecil rerata lama tanggap.

Kelemahannya :

- Layanan thd proses panjang.
- Jika proses pendek tiba terus-menerus, maka layanan proses panjang tertunda.
- Implementasi yaitu, bgmn menentukan proses terpendek kalau proses terus tiba

PROSES TERPENDEK DIPERTAMAKAN PRE-EMPSI (PTDP)

1. Pre-emptive Shortes Process Next (PSPN).
2. Shortest Remaining Time First (SRTF).

- PTDP merupakan penjadwalan dengan prioritas dan dengan pre-empsi.
- Prioritas didasarkan pada pendeknya prioritas.
- Makin pendek sisa proses makin tinggi prioritasnya.

Digunakan 2 langkah melihat pelaksanaan penjadwalan ini :

- Setiap kali, diperhatikan saat proses tiba atau saat proses rampung.
- Menghitung lama proses dari semua proses yang ada pada saat itu.

Cth : P T D P

Nama Proses	Saat Tiba	Lama Proses
A	0	7
B	2	3
C	4	9
D	5	4

Dibawah ini dibuat barisan saat.

Saat	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Proses	A	A	B	B	B	D	D	D	D	A	A	A	A	A	

Saat	14	15	16	17	18	19	20	21	22	23
Proses	C	C	C	C	C	C	C	C	C	

D a f t a r P r o s e s.

Nama Proses	Saat Tiba	Lama Proses	Saat Mulai	Saat Rampung	Lama Tanggap
A	0	7	0	14	14
B	2	3	2	5	3
C	4	9	14	23	19
D	5	4	5	9	4
				Jumlah	40
				Rerata	10

Keterangan:

- Pada saat 0, hanya proses A yang tiba. Maka mulai saat 0, proses A yang dikerjakan prosesor.
- Sampai saat 2, proses B tiba.
- Sisa proses dari proses A adl 5, sedangkan lama dari proses B adl 3, karena proses B lebih dekat ke rampung,
- Maka melalui pre-empsi, mulai saat 2, proses A dikeluarkan dari prosesor untuk digantikan proses B.
- Pada saat 4, proses C tiba.
- Pada saat ini sisa proses dari A = 5, sisa proses B = 1 lama proses dari C = 9, yang paling dekat ke rampung adl proses B, shg pd saat 4 itu, pelaksanaan proses B diteruskan.
- Pada saat 5, proses D tiba, pada saat 5 ini, sisa proses dari A adl 5, proses B rampung,
- Lama proses dari C = 9, lama proses D = 4 yang paling dekat ke rampung adl proses D, sehingga mulai pd saat 5, proses D dikerjakan.
- Pada saat 9, proses D rampung.
- Pada saat itu, sisa proses dari proses A adl 5 serta lama proses dr C adl 9.
- Yg paling dekat ke rampung adl proses A, sehingga mulai pd saat 9, sisa proses A dikerjakan akan rampung pd saat 14.
- Dan mulai saat 14,proses C dikerjakan sampai rampung pd saat 23.

Proses A ,dari saat 0 sampai saat 14.
Proses B, dari saat 2 sampai saat 5.
Proses C, dari saat 14 sampai saat 23.
Proses D, dari saat 5 sampai saat 9.

Maka dari daftar proses di dpt Rerata Lama Tanggap = 10 satuan wkt.

RATIO PENALTI TERTINGGI DIPERTAMAKAN (RPTD)

1. Highest Penalty Ratio Next (HPRN).
2. Highest Response Ratio Next (HRN).

Penjadwalan ini terletak diantara PTDP dengan PTD.

- Pada penjadwalan PTPD, proses pendek pada bagian belakang antrian akan mengalami banyak penundaan.
- Penjadwalan PTD,proses panjang akan banyak mengalami byk penundaan.
- Penjadwalan RPTD tetap mendahulukan proses pendek, namun prioritas proses panjang akan turut meningkatkan melalui peningkatan *rasio penaltynya*. Pada suatu sat, prioritas proses panjang yang lama tertunda akan dpt menyusul prioritas proses pendek.

Rumus Ratio Pinalti adl : $Rp = \frac{T}{t} = \frac{(s + t)}{t}$

dimana t = Lama Proses.

s = Lama Tunggu (antri, wkt sia-sia)

contoh : Penjadwalan RPTD

Nama Proses	Saat Tiba	Lama Proses
A	0	4
B	1	2
C	2	5
D	3	8
E	4	4

B a r i s a n S a a t

Saat	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Proses	A	A	A	A	B	B	C	C	C	C	C	E	E	E	E	

Saat	15	16	17	18	19	20	21	22	23

proses	D	D	D	D	D	D	D	D

Ket:

- Pada saat 0, hanya proses A yg tiba,
- Pd saat 0 proses A dikerjakan.
- Pd saat 4, proses A rampung.
- Dan pd saat itu dihitung nilai ratio pinalti dari setiap proses lain yg sdh tiba.
- Pd saat itu, semua proses sdh tiba, shg nilai ratio pinalti mereka dpt dihitung.

D a f t a r P r o s e s

Nama Proses	Saat Tiba	Lama Proses	Saat Mulai	Saat Rampung	Lama Tanggap
A	0	4	0	4	4
B	1	2	4	6	5
C	2	5	6	11	9
D	3	8	15	23	20
E	4	4	11	15	11
				Jumlah	49
				Rerata	9,8

Pada saat 4, ditemukan ratio penalti untuk proses B;C;D; E ; sbb:

Proses	Tiba Selama	Rasio Pinalti
B	3	(3 + 2) / 2 = 2,5
C	2	(2 + 5) / 5 = 1,2
D	1	(1 + 8) / 8 = 1,125
E	0	(0 + 4) / 4 = 1

Tampak disini proses B mempunyai nilai ratio pinalti tertinggi yaitu pd saat 4, Proses B dikerjakan.

Proses B rampung pd saat 6. pd saat itu ratio pinalti proses adl :

Proses	Tiba Selama	Ratio Penalti
C	4	(4 + 5) / 5 = 1,8
D	3	(3 + 8) / 8 = 1,375
E	2	(2 + 4) / 4 = 1,5

Pd saat 6, Proses C, memiliki ratio pinalti tertinggi.

Proses C dikerjakan sampai rampung pd saat 11.

Proses	Tiba Selama	Ratio Pinalti
D	8	(8 + 8) / 8 = 2
E	7	(7 + 4) / 4 = 2,75

Pd saat ini,proses E memiliki ratio penalti tertinggi.

Proses E dikerjakan , pd saat E, proses E rampung.

Proses D dikerjakan sampai rampung pd saat 23.

Jadi Proses A : 0 - 4

B : 4 - 6 D : 15 - 23

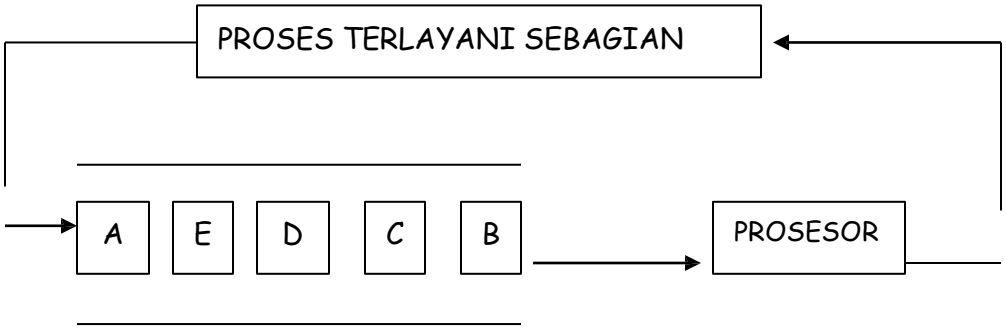
C : 11 - 15 E : 11 - 15

Penjadwalan Putar Gelang.

- Penjadwalan ROUND ROBIN (RR).
- RIME SLICE.
- Penjadwalan ini berlangsung tanpa prioritas, tetapi dengan pre-empsi.

- Pd penjadwalan ini, secara bergiliran berdasarkan antrian (tanpa prioritas), prosesor melayani sejenak setiap proses.

- Secara berturut-turut, proses yg telah dilayani prosesor dan belum rampung akan kembali ke akhir atrian yg ada pd saat itu, shg penggiliran ini berputar seperti gelang.
- Hanya proses yg rampung terlayani yg meninggalkan prosesor dan antrian itu.



Pelayanan prosesor dgn penjadwalan Putar Gelang (PG).

Wkt sejenak yg digunakan oleh prosesor untuk melayani setiap proses adl

" KUANTUM WAKTU ".

Cth : penjadwalan PG = RR

Nama Proses	Saat Tiba	Lama Proses
A	0	7
B	0	5
C D	0	8
E	0	2
	0	6
KUANTUM WAKTU : 3		

Pelaksanaan Proses adl sbb :

saat	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**

Proses	**A**	**A**	**A**	**B**	**B**	**B**	**C**	**C**	**C**	**D**	**D**	**E**	**E**	**E**	**A**	**A**	**A**
Saat	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**					
Proses	**B**	**B**	**C**	**C**	**C**	**E**	**E**	**E**	**A**	**C**	**C**						

Ket:

- Pd saat 0, proses A dikerjakan,pd saat 3 atau pd saat akhir kuantum, pelaksanaan proses A dihentikan. Melalui pre-empsi proses A dikeluarkan dr prosesor.
- Pd saat itu proses B dikerjakan. Pd saat 6 (pd akhir kuantum) wkt proses B dikeluarkan dr prosesor.
- Pd saat itu juga proses C dikerjakan. Pd saat 9 wkt proses C dikeluarkan dr prosesor.
- Pd saat itu proses D dikerjakan, krn lama proses D = 2 satuan wkt (kurang dr kuantum wkt), maka proses D rampung pd saat 11.
- Pd saat 11, proses E dikerjakan. Pd saat 14 (sisa akhir kuantum wkt), proses E dikeluarkan.
- Pd saat itu juga, sisa proses A dikerjakan lagi, namun pd saat 17 proses A dikeluarkan lagi.
- Pd saat 17 ini, sisa proses B dikerjakan. Dan pd saat 19, proses B rampung.
- Pd saat 19, sisa proses C dikerjakan.
- Pd saaat 22, wkt proses C dikeluarkan lagi. Krn rampung, mk pd saat 22 sisa E dikerjakan. Pd saat 25 proses E rampung.
- Pd saat 25, sisa proses A dikerjakan, pd saat 26 proses A rampung.
- Pd saat 26 itu, sisa proses E dikerjakan dan rampung saat 28.

Nama Proses	Saat Tiba	Lama Proses	Saat Mulai	Saat Rampung	Lama Tanggap
A	0	7	0	26	26
B	0	5	3	19	19
C	0	8	6	28	28
D	0	2	9	11	11

E	0	6	11	25	25
				Jumlah Rerata	109 21,8

Jadi Proses A berlangsung dari 0 - 26.

B 3 - 19.

C 6 - 28

D 9 - 11. E 11 - 25.

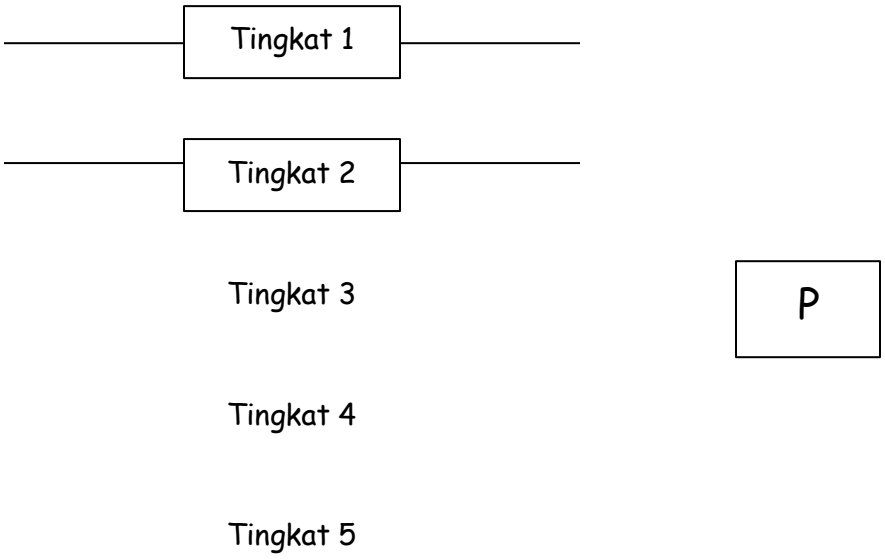
Lama tanggap Rerata = 21,8 satuan wkt.

PENJADWALAN MULTITINGKAT

Ada 2 penjadwalan multitingkat.

1. Antrian Multi tingkat.
   Sesuai dgn perbedaan tingkat kepentingan dpt disusun beberapa tingkat proses. Stp tingkat proses mengandung sejumlah proses dgn tingkat kepentingan yg sama atau bersamaan.

Cth :



Penjadwalan antrian multi tingkat. Tingkat 1

mencakup proses terpenting. Tingkat 2

mencakup proses kedua terpenting. Tingkat 3

mencakup proses ketiga terpenting.

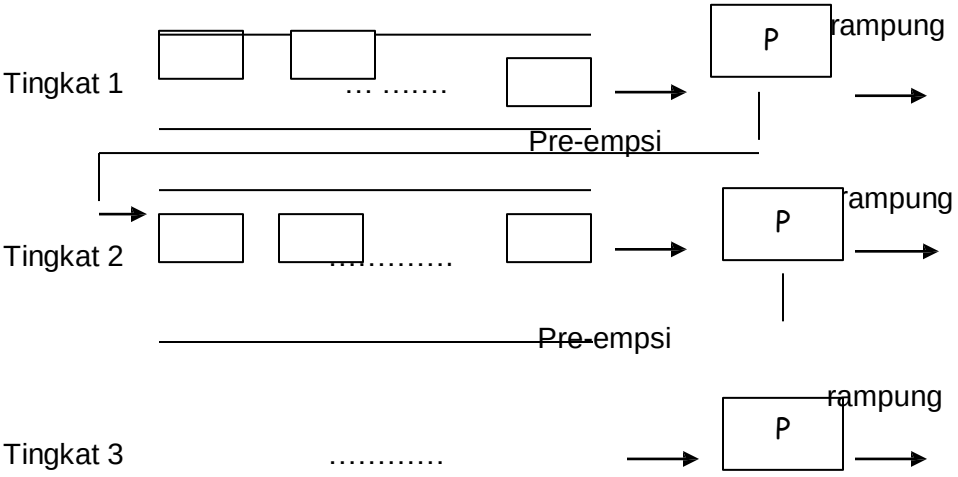
…….d……s……t……

2. Antrian Multi Tingkat Berbalikan.

   Gbr dibawah ini adl penjadwalan antrian multitingkat.

   Karena kuantum wkt pd tingkat setelah tingkat 1 makin panjang, mk pd tingkat selanjutnya itu, kesempatan rampung bagi proses makin besar, sekalipun wkt tunggu antrian juga makin panjang.

   Proses pendek akan rampung pd tingkat atas. Dan hanya proses yg sgt panjang yg mencapai tingkat akhir.



METODE EVALUASI PENJADWALAN.

Penjadwalan mempunyai hubungan dgn jenis pekerjaan.

Untuk menemukan jenis penjadwalan yg cocok untuk jenis pekerjaan tertentu,perlu dilakukan evaluasi thd sejumlah penjadwalan prosesor dlm kerangka jenis pekerjaan itu.

Ada beberapa metode evaluasi yaitu :

1. Metode Evaluasi analitik, dibagi :
    a. Pemodelan determistik.
    b. Analisis model antrian.
2. Metode simulasi.
3. Metode implementasi.

- Ad.a. Metode Pemodelan Deterministik.

  Pd metode ini sengaja ditetapkan sejumlah pekerjaan tertentu. Setelah itu, diterapkan berbagai macam penjadwalan thd pekerjaan itu.

  Pd penerapan itu, dicatat hasil yg dicapai oleh stp macam penjadwalan. Terutama mencatat Lama Tanggap atau Rerata wkt sia-sia (wkt tunggu).

- Keunggulan metode ini terletak pada ketepatannya untuk menunjukkan rerata lama tanggap atau rerata wkt sia-sia.
- Kelemahan metode ini terletak pd hasil evaluasi yg hanya berlaku untuk proses yg sudah ditetapkan.
- Cara mengatasi kelemahan itu adl : melalui percobaan berulang-ulang thd beberapa cth perangkat proses.

- Ad.b. Metode Analisis Model Antrian.

Metode ini menganggap sistem komputer sebagi suatu jaringan alat layan.

Setiap pelayan memiliki suatu antrian proses yg menunggu giliran untuk dilayani. Sebagai pelayan, proses mempunyai antrian siap, serta alat periferal mempunyai antrian alat.

- Dlm hal ini di gunakan rumus distribusi probabilitas untuk diperkirakan bentuk antrian serta bentuk proses pd sistem komputer.
- Rumus itu adl, sbb:

n = rerata panjang antrian
s = rerata wkt sia-sia

u = rerata kecep. Tiba proses baru

- Ad.2. Metode Simulasi.

  Metode ini membuat sejumlah variabel yg mensimulasi sistem komputer.

  Ada variabel yg mensimulasi kunci wkt, saat tiba, dst…

Ad.3.. Metode Implementasi.

Metode ini menerapkan berbagai penjadwalan pd pekerjaan yg sesungguhnya.

- Metode ini cukup cermat.
- Kelemahannya : mengganggu para pemakai komputer.

Daftar Pustaka

1. Stallings (2005). *Operating Systems, Internals and Design Principles*. Pearson: Prentice Hall.
2. Dhotre, I.A. (2009). *Operating Systems*. Technical Publications.

MODUL PERKULIAHAN

Sinkronisasi

Monitor termasuk ke dalam level tertinggi mekanisme sinkronisasi yang berguna untuk mengkoordinir aktivitas dari banyak thread ketika mengakses data melalui pernyataan yang telah disinkronisasi.

Abstract

Akses-akses yang dilakukan secara ersama-sama ke data yang sama, dapat menyebabkan data menjadi tidak konsisten, maka dibutuhkan mekanisme -mekanisme untuk memastikan pemintaan ekseskusi dari proses yang bekerja.

Kompetensi

Memahami konsep dasar sinkronisasi dalam kaitannya dengan kerjasama diantara proses-proses yang ada.
Memahami bagaimana proses penanganan masalah pada proses yang secara bersamaan menggunakan memori yang sama.

Sinkronisasi

Dua proses berbagi sebuah buffer dengan ukuran yang tetap. Salah satunya produser, meletakkan informasi ke buffer yang lainnya. Konsumen mengambil informasi dari buffer. Ini juga dapat digeneralisasi untuk masalah yang memiliki m buah produsen dan n buah konsumen, tetapi kita hanya akan memfokuskan kasus dengan satu produsen dan satu konsumen karena diasumsikan dapat menyederhanakan solusi.

Masalah akan timbul ketika produsen ingin menaruh barang yang baru tetapi buffer sudah penuh. Solusi untuk produsen adalah istirahat (*sleep*) dan akan dibangunkan ketika konsumen telah mengambil satu atau lebih barang dari buffer. Biasanya jika konsumen ingin mengambil barang dari buffer dan melihat bahwa buffer sedang kosong, maka konsumen istirahat (*sleep*) sampai produsen meletakkan barang pada buffer dan membangunkan (*wake up*) consumer.

Pendekatan seperti ini terdengar cukup sederhana, tetapi hal ini dapat menggiring kita ke jenis masalah yang sama seperti *race condition* dengan spooler direktori.

Untuk mengetahui jumlah barang di buffer, kita membutuhkan sebuah variabel kita namakan count. Jika jumlah maksimum dairi barang yang dapat ditampung buffer adalah N, kode produser pertama kali akan mencoba untuk mengetahui apakah nilai count sama dengan nilai N. Jika itu terjadi maka produsen akan istirahat (*sleep*), tetapi jika nilai count tidak sama dengan N, produsen akan terus menambahkan barang dan menaikkan nilai count.

Sekarang mari kita kembali ke permasalahan race condition. Ini dapat terjadi karena akses ke count tidak dipaksakan. Situasi seperti itu mungkin dapat terjadi. Buffer sedang kosong dan konsumen baru saja membaca count untuk melihat apakah count bernilai 0. Pada saat itu, penjadual memutuskan untuk mengentikan proses konsumen sementara dan menjalakan produsen. Produsen memasukkan barang ke buffer, menaikkan nilai count, dan memberitahukan bahwa count sekarang bernilai 1. Pemikiran bahwa count baru saja bernilai 0 sehingga konsumen harus istirahat (*sleep*). Produsen memanggil fungsi *wake up* untuk membangkitkan konsumen.

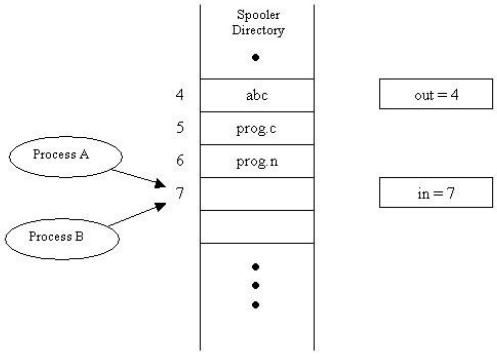
Sayangnya, konsumen secara logika belum istirahat. Jadi sinyal untuk membangkitkan konsumen, tidak dapat ditangkap oleh konsumen. Ketika konsumen bekerja berikutnya, konsumen akan memeriksa nilai count yang dibaca sebelumnya, dan mendapatkan nilai 0, kemudian konsumen istirahat (*sleep*) lagi. Cepat atau lambat produsen akan mengisi buffer dan juga pergi istirahat (*sleep*). Keduanya akan istirahat selamanya.

Inti permasalahannya disini adalah pesan untuk membangkitkan sebuah proses tidak tersampaikan. Jika pesan/ sinyal ini tersampaikan dengan baik, segalanya akan berjalan lancar.

Race Condition

Race Condition adalah situasi di mana beberapa proses mengakses dan memanipulasi data bersama pada saat besamaan. Nilai akhir dari data bersama tersebut tergantung pada proses yang terakhir selesai. Unutk mencegah *race condition*, proses-proses yang berjalan besamaan harus di disinkronisasi. Dalam beberapa sistem operasi, proses-proses yang berjalan bersamaan mungkin untuk membagi beberapa penyimpanan umum, masing-masing dapat melakukan proses baca (*read*) dan proses tulis (*write*). Penyimpanan bersama (*shared storage*) mungkin berada di memori utama atau berupa sebuah berkas bersama, lokasi dari memori bersama tidak merubah kealamian dari komunikasi atau masalah yang muncul. Untuk mengetahui bagaimana komunikasi antar proses bekerja, mari kita simak sebuah contoh sederhana, sebuah print spooler. Ketika sebuah proses ingin mencetak sebuah berkas, proses tersebut memasukkan nama berkas ke dalam sebuah spooler direktori yang khusus. Proses yang lain, printer daemon, secara periodik memeriksa untuk mengetahui jika ada banyak berkas yang akan dicetak, dan jika ada berkas yang sudah dicetak dihilangkan nama berkasnya dari direktori.

Bayangkan bahwa spooler direktori memiliki slot dengan jumlah yang sangat besar, diberi nomor 0, 1, 2, 3, 4,... masing-masing dapat memuat sebuah nama berkas. Juga bayangkan bahwa ada dua variabel bersama, out, penunjuk berkas berikutnya untuk dicetak, dan in, menunjuk slot kosong di direktori. Dua vaiabel tersebut dapat menangani sebuah two-word berkas untuk semua proses. Dengan segera, slot 0, 1, 2, 3 kosong (berkas telah selesai dicetak), dan slot 4, 5, 6 sedang terisi (berisi nama dari berkas yang antre untuk dicetak). Lebih atau kurang secara besamaan, proses A dan B, mereka memutuskan untuk antre untuk sebuah berkas untuk dicetak.



Race Condition.

Dalam *Murphy's Law* kasus tesebut dapat terjadi. Proses A membaca in dan menyimpan nilai "7" di sebuah variabel lokal yang disebut next_free_slot. Sebuah clock interrupt terjadi dan CPU memutuskan bahwa proses A berjalan cukup lama, sehingga digantika oleh proses B. Proses B juga membaca in, dan juga mengambil nilai 7, sehingga menyimpan nama berkas di slot nomor 7 dan memperbaharui nilai in menjadi 8. Maka proses mati dan melakukan hal lain.

Akhirnya proses A berjalan lagi, dimulai dari tempat di mana proses tersebut mati. Hal ini terlihat dalam next_free_slot, ditemukan nilai 7 di sana, dan menulis nama berkas di slot nomor 7, menghapus nama berkas yang bau saja diletakkan oleh proses B. Kemudian proses A menghitung next_free_slot + 1, yang nilainya 8 dan memperbaharui nilai in menjadi 8. Direktori spooler sekarang secara internal konsisten, sehingga printer daemon tidak akan memberitahukan apa pun yang terjadi, tetapi poses B tidak akan mengambil output apa pun. Situasi seperti ini, dimana dua atau lebih proses melakukan proses reading atau writing beberapa shared data dan hasilnya bergantung pada ketepatan berjalan disebut *race condition*.

Critical Section

Bagaimana menghindari *race conditions*? Kunci untuk mencegah masalah ini dan di situasi yang lain yang melibatkan shared memori, shared berkas, and shared sumber daya yang lain adalah menemukan beberapa jalan untuk mencegah lebih dari satu proses untuk melakukan proses writing dan reading kepada shared data pada saat yang sama. Dengan kata lain kita membutuhkan *mutual exclusion*, sebuah jalan yang menjamin jika sebuah proses sedang menggunakan shared berkas, proses lain dikeluarkan dari pekerjaan yang sama. Kesulitan yang terjadi karena proses 2 mulai menggunakan variabel bersama sebelum proses 1 menyelesaikan tugasnya.

Masalah menghindari *race conditions* dapat juga diformulasikan secara abstrak. Bagian dari waktu, sebuah proses sedang sibuk melakukan perhitungan internal dan hal lain yang tidak enggiring ke kondisi *race conditions*. Bagaimana pun setiap kali sebuah proses mengakses shared memory atau shared berkas atau melakukan sesuatu yang kitis akan menggiring kepada *race conditions*. Bagian dari program dimana shaed memory diakses disebut *Critical Section* atau *Critical Region*.

Walau pun dapat mencegah *race conditions*, tapi tidak cukup untuk melakukan kerjasama antar proses secara pararel dengan baik dan efisien dalam menggunakan shared data. Kita butuh 4 kondisi agar menghasilkan solusi yang baik:

I. Tidak ada dua proses secara bersamaan masuk ke dalam citical section.
II. Tidak ada asumsi mengenai kecepatan atau jumlah cpu.
III. Tidak ada proses yang berjalan di luar critical secion yang dapat mengeblok proses lain.
IV. Tidak ada proses yang menunggu selamamya untuk masuk critical section.

Critical Section adalah sebuah segmen kode di mana sebuah proses yang mana sumber daya bersama diakses. Terdiri dari: *Entry Section*: kode yang digunakan untuk masuk ke dalam *critical section*

Critical Section: Kode di mana hanya ada satu proses yang dapat dieksekusi pada satu waktu

Exit Section: akhir dari *critical section*, mengizinkan proses lain

Remainder Section: kode istirahat setelah masuk ke *critical section*

Solusi yang diberikan harus memuaskan permintaaan berikut:

Mutual exclution
Deadlock free
Starvation free

Pendekatan yang mungkin untuk solusi proses sinkronisasi

i. Solusi Piranti lunak (Software solution)

Tanpa Sinkronisasi.
Dengan Sinkronisasi.
Low-level primitives: *semaphore*
High-level primitives: *monitors*

ii. Solusi Piranti Keras (Hardware solution)

Mutual Exclusion

Tiga kondisi untuk menentukan *mutual Exclusion*

1. Tidak ada dua proses yang pada saat bersamaan berada di *critical region*.
2. Tidak ada proses yang berjalan diluar *critical region* yang bisa menghambat proses lain
3. Tidak ada proses yang tidak bisa masuk ke *critical region*

Semaphore

Jika kita ingin dapat melakukan proses tulis lebih rumit kita membutuhkan sebuah bahasa untuk melakukannya. Kita akhirnya medefinisikan semaphore yang kita asumsikan sebagai sebuah operasi atomik.

Semaphore adalah pendekatan yang diajukan oleh Djikstra, dengan prinsip bahwa dua proses atau lebih dapat bekerja sama dengan menggunakan penanda-penanda sederhana. Seperti proses dapat dipaksa berhenti pada suatu saat, sampai proses mendapatkan penanda tertentu itu. Sembarang kebutuhan koordinasi kompleks dapat dipenuhi dengan struktur penanda yang cocok untuk kebutuhan itu. Variabel khusus untuk penanda ini disebut semaphore.

Semaphore mempunyai dua sifat, yaitu:

i. Semaphore dapat diinisialisasi dengan nilai non-negatif.
ii. Terdapat dua operasi terhadap semaphore, yaitu Down dan Up. Usulan asli yang disampaikan Djikstra adalah operasi P dan V.

Operasi *Down*

Operasi ini menurunkan nilai semaphore, jika nilai semaphore menjadi non-positif maka proses yang mengeksekusinya diblocked.

Operasi Down adalah atomic, tak dapat diinterupsi sebelaum diselesaikan.Emnurunkan nilai, memeriksa nilai, menempatkan proses pada antrian dan memblocked sebagai instruksi tunggal. Sejak dimulai, tak ada proses alain yang dapat mengakses semaphore sampai operasi selesai atau diblocked.

Problem Klasik pada Sinkronisasi

Ada tiga hal yang selalu memjadi masalah pada proses sinkronisasi:

i. Problem *Bounded buffer*.
ii. Problem *Reades and Writer*.
iii. Problem *Dining Philosophers*.

Problem *Readers-Writers*

Problem lain yang terkenal adalah readers-writer problem yang memodelkan proses yang mengakses database. Sebagai contoh sebuah sistem pemesanan sebuah perusahaan penerbangan, dimana banyak proses berkompetisi berharap untuk membaca (*read*) dan menulis (*write*). Hal ini dapat diterima bahwa banyak proses membaca database pada saat yang sama, tetapi jika suatu proses sedang menulis database, tidak boleh ada proses lain yang mengakses database tersebut, termasuk membaca database tersebut.

Dalam solusi ini, pertama-tama pembaca mengakses database kemudian melakukan DOWN pada semaphore db.. Langkah selanjutnya readers hanya menaikkkan nilai sebuah counter. Hasil dari pembaca nilai counter diturunkan dan nilai terakhir dilakukan UP pada semaphore, mengizinkan memblok writer.

Misalkan selama sebuah reader menggunakan database, reader lain terus berdatangan. Karena ada dua reader pada saat bersamaan bukanlah sebuah masalah, maka reader yang kedua diterima, reader yang ketiga juga dapat diterima jika terus berdatangan reader-reader baru.

Sekarang misalkan writer berdatangan terus menerus. Writer tidak dapat diterima ke database karena writer hanya bisa mengakses data ke database secara ekslusif, jadi

writer ditangguhkan. Nanti penambahan reader akan menunjukkan peningkatan. Selama paling tidak ada satu reader yang aktif, reader berikutnya jika datang akan diterima.

Sebagai konsekuensi dari strategi ini, selama terdapat suplai reader yang terus-menerus, mereka akan dilayani segera sesuai kedatanga mereka. Writer akan ditunda sampai tidak ada reader lagi. Jika sebuah reader baru tiba, katakan, setiap dua detik, dan masing-masing reader mendapatkan lima detik untuk melakukan tugasnya, writer tudak akan pernah mendapatkan kesempatan.

Untuk mencegah situasi seperti itu, program dapat ditulis agak sedikit berbeda: Ketika reader tiba dan writer menunggu, reader ditunda dibelakang writer yang justru diterima dengan segera. Dengan cara ini, writer tidak harus menunggu reader yang sedang aktif menyelesaikan pekerjaannya, tapi tidak perlu menunggu reader lain yang datang berturut-turut setelah itu.

Problem *Dining Philosopers*

Pada tahun 1965, Djikstra menyelesaikan sebuah masalah sinkronisasi yang beliau sebut dengan dining philisophers problem. Dining philosophers dapat diuraikan sebagai berikut: Lima orang filosuf duduk mengelilingi sebuah meja bundar. Masing-masing filosof mempunyai sepiring spageti. Spageti-spageti tersebut sangat licin dan membutuhkan dua garpu untuk memakannya. Diantara sepiring spageti terdapat satu garpu.

Kehidupan para filosof terdiri dari dua periode, yaitu makan atau berpikir. Ketika seorang filosof lapar, dia berusaha untuk mendapatkan garpu kiri dan garpu kanan sekaligus. Jika sukses dalam mengambil dua garpu, filosof tersebut makan untuk sementara waktu, kemudian meletakkan kedua garpu dan melanjutkan berpikir.

Pertanyaan kuncinya adalah, dapatkah anda menulis program untuk masing-masing filosof yang melakukan apa yang harus mereka lakukan dan tidak pernah mengalami kebuntuan.

Prosedur take-fork menunggu sampai garpu-garpu yang sesuaididapatkan dan kemudian menggunakannya. Sayangnya dari solusi ini ternyata salah. Seharusnya lima orang filosof mengambil garpu kirinya secara bersamaan. Tidak akan mungkin mereka mengambil garpu kanan mereka, dan akan terjadi deadlock.

Kita dapat memodifikasi program sehingga setelah mengambil garpu kiri, program memeriksa apakah garpu kanan meungkinkan untuk diambil. Jika garpu kanan tidak mungkin diambil, filosof tersebut meletakkan kembali garpu kirinya, menunggu untuk beberapa waktu, kemudia mengulangi proses yang sama. Usulan tersebut juga salah, walau pun dengan alasan yang berbeda. Dengan sedikit nasib buruk, semua filosof dapat memulai algoritma secara bersamaan, mengambil garpu kiri mereka, melihat garpu kanan mereka yang tidak mungkin untuk diambil, meletakkan kembali garpu kiri mereka, menunggu, mengambil garpu kiri mereka lagi secara bersamaan, dan begitu seterusnya. Situasi seperti ini dimana semua program terus berjalan secara tidak terbatas tetapi tidak ada perubahan/kemajuan yang dihasilkan disebut starvation.

Sekarang anda dapat berpikir "jika filosof dapat saja menunggu sebuah waktu acak sebagai pengganti waktu yang sama setelah tidak dapat mengambil garpu kiri dan kanan, kesempatan bahwa segala sesuatau akan berlanjut dalam kemandegan untuk beberapa jam adalah sangat kecil." Pemikiran seperti itu adalah benar,tapi beberapa aplikasi mengirimkan sebuah solusi yang selalu bekerja dan tidak ada kesalahan tidak seperti hsk nomor acak yang selalu berubah.

Sebelum mulai mengambil garpu, seorang filosof melakukan DOWN di mutex. Setelah menggantikan garpu dia harus melakukan UP di mutex. Dari segi teori, solusi ini cukup memadai. Dari segi praktek, solusi ini tetap memiliki masalah. Hanya ada satu filosof yang dapat makan spageti dalam berbagai kesempatan. Dengan lima buah garpu, seharusnya kita bisa menyaksikan dua orang filosof makan spageti pada saat bersamaan.

Solusi yang diberikan diatas benar dan juga mengizinkan jumlah maksimum kegiatan paralel untuk sebuah jumlah filosf yang berubah-ubah ini menggunakan sebuah array, state, untuk merekam status seorang filosof apakah sedang makan (*eating*), berpikir (*think*), atau sedang lapar (*hungry*) karena sedang berusaha mengambil garpu. Seorang filosof hanya dapat berstatus makan (*eating*) jika tidak ada tetangganya yang sedang makan juga. Tetangga seorang filosof didefinisikan ole LEFT dan RIGHT.

Dengan kata lain, jika i = 2, maka tetangga kirinya (LEFT) = 1 dan tetangga kanannya (RIGHT) = 3. Program ini menggunakan sebuah array dari semaphore yang lapar (hungry) dapat ditahan jika garpu kiri atau kanannya sedang dipakai tetangganya. Catatan bahwa masing-masing proses menjalankan prosedur filosof sebagai kode utama, tetapi prosedur

yang lain seperti take-forks, dan test adalah prosedur biasa dan bukan proses-proses yang terpisah.

Monitors

Solusi sinkronisasi ini dikemukakan oleh Hoare pada tahun 1974. Monitor adalah kumpulan prosedur, variabel dan struktur data di satu modul atau paket khusus. Proses dapat memanggil prosedur-prosedur kapan pun diinginkan. Tapi proses tak dapat mengakses struktur data internal dalam monitor secara langsung. Hanya lewat prosedur-prosedur yang dideklarasikan minitor untuk mengakses struktur internal.

Properti-properti monitor adalah sebagai berikut:

i. Variabel-variabel data lokal, hanya dapat diakses oleh prosedur-prosedur dala monitor dan tidak oleh prosedur di luar monitor.
ii. Hanya satu proses yang dapat aktif di monitor pada satu saat. Kompilator harus mengimplementasi ini(mutual exclusion).
iii. Terdapat cara agar proses yang tidak dapat berlangsung di-blocked. Menambahkan variabel-variabel kondisi, dengan dua operasi, yaitu Wait dan Signal.
iv. ait: Ketika prosedur monitor tidak dapat berlanjut (misal producer menemui buffer penuh) menyebabkan proses pemanggil diblocked dan mengizinkan proses lain masuk monitor.
v. Signal: Proses membangunkan partner-nya yang sedang diblocked dengan signal pada variabel kondisi yang sedang ditunggu partnernya.
vi. Versi Hoare: Setelah signal, membangunkan proses baru agar berjalan dan menunda proses lain. vii. Versi Brinch Hansen: Setelah melakukan signal, proses segera keluar dari monitor.

Dengan memaksakan disiplin hanya satu proses pada satu saat yang berjalan pada monitor, monitor menyediakan fasilitas mutual exclusion. Variabel-variabel data dalam monitor hanya dapat diakses oleh satu proses pada satu saat. Struktur data bersama dapat dilindungi dengan menempatkannya dalam monitor. Jika data pada monitor merepresentasikan sumber daya, maka monitor menyediakan fasilitas mutual exclusion dalam mengakses sumber daya itu.

Daftar Pustaka

1. Hariyanto, B.,"*Sistem Operasi*", Bandung: Informatika, Desember 1997
2. Silberschatz, A., Gagne, G. dan Galvin, P., "*Applied Operating System Concept*", John Wiley and Sons Inc., 2000
3. Tanenbaum, Andrew S., "*Modern Operating Systems*", Englewood Cliffs, New Jersey: Prentice-Hall Inc., 1992

MODUL PERKULIAHAN

Deadlock

Dua atau lebih proses dikatakan berada dalam kondisi deadlock, bila setiap proses yang ada menunggu suatu kejadian yang hanya dapat dilakukan oleh proses lain dalam himpunan tersebut.

Abstract

Secara sederhana *deadlock* dapat terjadi dan menjadi hal yang merugikan, jika pada suatu saat ada suatu proses yang memakai sumber daya dan ada proses lain yang menunggunya.

Kompetensi

Memahami konsep deadlock, apa saja dapat menimbulkan deadlock dan bagaimana mengatasinya.

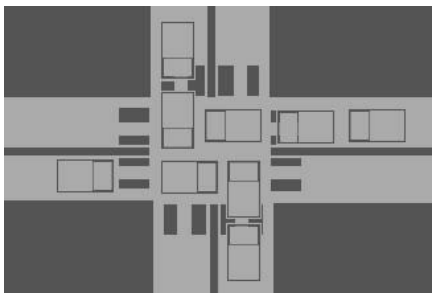
Deadlock

Latar Belakang

Misalkan pada suatu komputer terdapat dua buah program, sebuah *tape drive* dan sebuah *printer*. Program A mengontrol *tape* drive, sementara program B mengontrol *printer*. Setelah beberapa saat, program A meminta *printer*, tapi *printer* masih digunakan. Berikutnya, B meminta *tape drive*, sedangkan A masih mengontrol *tape drive*. Dua program tersebut memegang kontrol terhadap sumber daya yang dibutuhkan oleh program yang lain. Tidak ada yang dapat melanjutkan proses masing-masing sampai program yang lain memberikan sumber dayanya, tetapi tidak ada yang mengalah. Kondisi inilah yang disebut *Deadlock* atau pada beberapa buku disebut *Deadly Embrace Deadlock* yang mungkin dapat terjadi pada suatu proses disebabkan proses itu menunggu suatu kejadian tertentu yang tidak akan pernah terjadi.

Dua atau lebih proses dikatakan berada dalam kondisi *deadlock*, bila setiap proses yang ada menunggu suatu kejadian yang hanya dapat dilakukan oleh proses lain dalam himpunan tersebut.

Contoh berikut ini ilustrasi dipersimpangan jalan: Terdapat satu jalur pada jalan. Mobil digambarkan sebagai proses yang sedang menuju sumber daya. Untuk mengatasinya beberapa mobil harus preempt (mundur). Sangat memungkinkan untuk terjadinya starvation (kondisi proses tak akan mendapatkan sumber daya).

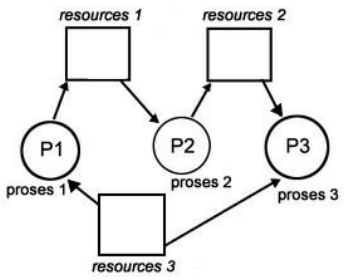


Gambar. Persimpangan.

Resources-Allocation Graph

Sebuah cara visual (matematika) untuk menentukan apakah ada *deadlock*, atau kemungkinan terjadinya. G = (V, E) Graf berisi *node and edge*. *Node* V terdiri dari proses-proses = {P1, P2, P3, ...} dan jenis *resource*. {R1, R2, ...} *Edge* E adalah (Pi, Rj) atau (Ri, Pj) Sebuah panah dari *process* ke *resource* menandakan proses meminta *resource*.

Sebuah panah dari *resource* ke *process* menunjukkan sebuah *instance* dari *resource* telah dtempatkan ke proses. *Proses* adalah lingkaran, *resource* adalah kotak; titik-titik merepresentasikan jumlah *instance* dari *resource* Dalam tipe. Meminta poin-poin ke kotak, perintah datang dari titik.

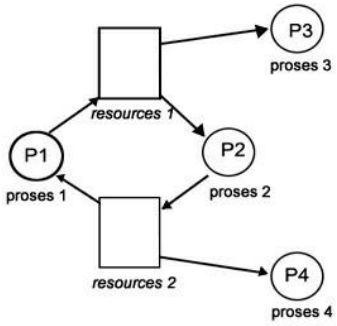


Gambar Graph.

Jika graf tidak berisi lingkaran, maka t .

Jika membentuk lingkaran, maka:

i. jika tipe *resource* memiliki banyak *instance*, maka *deadlock* **bisa terjadi.**



Gambar Non Deadlock.

ii. jika setiap tipe *resource* mempunyai satu *instance*, maka *deadlock* telah terjadi.

Gambar Deadlock.

Model Sistem

Menurut *Coffman* dalam bukunya *"Operating System"* menyebutkan empat syarat bagi terjadinya *deadlock*, yaitu:

i. *Mutual Exclusion*
Suatu kondisi dimana setiap sumber daya diberikan tepat pada satu proses pada suatu waktu.

ii. *Hold and Wait*
Kondisi yang menyatakan proses-proses yang sedang memakai suatu sumber daya dapat meminta sumber daya yang lain.

iii. *Non-pre-emptive*
Kondisi dimana suatu sumber daya yang sedang berada pada suatu proses tidak dapat diambil secara paksa dari proses tersebut,sampai proses itu melepaskannya.

iv. *Circular Wait*
Kondisi yang menyatakan bahwa adanya rantai saling meminta sumber daya yang dimiliki oleh suatu proses oleh proses lainnya.

Strategi menghadapi *Deadlock*

Strategi untuk menghadapi *deadlock* dapat dibagi menjadi tiga pendekatan, yaitu:

i. Mengabaikan adanya *deadlock*.
ii. Memastikan bahwa *deadlock* tidak akan pernah ada, baik dengan metode. Pencegahan, dengan mencegah empat kondisi *deadlock* agar tidak akan pernah terjadi. Metode Menghindari *deadlock*, yaitu mengizinkan empat kondisi *deadlock*, tetapi menghentikan setiap proses yang kemungkinan mencapai *deadlock*.
iii. Membiarkan *deadlock* untuk terjadi, pendekatan ini membutuhkan dua metode yang saling mendukung, yaitu:

- Pendeteksian *deadlock*, untuk mengidentifikasi ketika *deadlock* terjadi.

• Pemulihan *deadlock*, mengembalikan kembali sumber daya yang dibutuhkan pada proses yang memintanya.

Strategi *Ostrich*

Pendekatan yang paling sederhana adalah dengan menggunakan strategi burung unta: masukkan kepala dalam pasir dan seolah-olah tidak pernah ada masalah sama sekali. Beragam pendapat muncul berkaitan dengan strategi ini. Menurut para ahli Matematika, cara ini sama sekali tidak dapat diterima dan semua keadaan *deadlock* harus ditangani. Sementara menurut para ahli Teknik, jika komputer lebih sering mengalami kerusakkan disebabkan oleh kegagalan *hardware*, *error* pada kompilator atau *bugs* pada sistem operasi. Maka ongkos yang dibayar untuk melakukan penanganan *deadlock* sangatlah besar dan lebih baik mengabaikan keadaan *deadlock* tersebut. Metode ini diterapkan pada sistem operasi UNIX dan MINIX.

Mencegah *Deadlock*

Metode ini merupakan metode yang paling sering digunakan. Metode Pencegahan dianggap sebagai solusi yang bersih dipandang dari sudut tercegahnya *deadlock*. Tetapi pencgahan akan mengakibatkan kinerja utilisasi sumber daya yang buruk.

Metode pencegahan menggunakan pendekatan dengan cara meniadakan empat syarat yang dapat menyebabkan deadlock terjadi pada saat eksekusi Coffman (1971).

Syarat pertama yang akan dapat ditiadakan adalah *Mutual Exclusion*, jika tidak ada sumber daya yang secara khusus diperuntukkan bagi suatu proses maka tidak akan pernah terjadi *deadlock*. Namun jika membiarkan ada dua atau lebih proses mengakses sebuah sumber daya yang sama akan menyebabkan *chaos*. Langkah yang digunakan adalah dengan *spooling* sumber daya, yaitu dengan mengantrikan *job-job* pada antrian dan akan dilayani satu-satu.

Beberapa masalah yang mungkin terjadi adalah:

i. Tidak semua dapat di-*spool*, tabel proses sendiri tidak mungkin untuk di-*spool*
ii. Kompetisi pada ruang disk untuk *spooling* sendiri dapat mengarah pada *deadlock*

Hal inilah yang menyebabkan mengapa syarat pertama tidak dapat ditiadakan, jadi *mutual exclusion* benar-benar tidak dapat dihilangkan.

Cara kedua dengan meniadakan kondisi *hold and wait* terlihat lebih menjanjikan. Jika suatu proses yang sedang menggunakan sumber daya dapat dicegah agar tidak dapat menunggu sumber daya yang lain, maka *deadlock* dapat dicegah. Langkah yang digunakan adalah dengan membuat proses agar meminta sumber daya yang mereka butuhkan pada awal proses sehingga dapat dialokasikan sumber daya yang dibutuhkan. Namun jika terdapat sumber daya yang sedang terpakai maka proses tersebut tidak dapat memulai prosesnya.

Masalah yang mungkin terjadi:

i. Sulitnya mengetahui berapa sumber daya yang dibutuhkan pada awal proses
ii. Tidak optimalnya pengunaan sumber daya jika ada sumber daya yang digunakan hanya beberapa waktu dan tidak digunakan tapi tetap dimiliki oleh suatu proses yang telah memintanya dari awal.

Meniadakan syarat ketiga *non preemptive* ternyata tidak lebih menjanjikan dari meniadakan syarat kedua, karena dengan meniadakan syarat ketiga maka suatu proses dapat dihentikan ditengah jalan. Hal ini tidak dimungkinkan karena hasil dari suatu proses yang dihentikan menjadi tidak baik.

Cara terakhir adalah dengan meniadakan syarat keempat *circular wait*. Terdapat dua pendekatan, yaitu:

i. Mengatur agar setiap proses hanya dapat menggunakan sebuah sumber daya pada suatu waktu, jika menginginkan sumber daya lain maka sumber daya yang dimiliki harus dilepas.

ii. Membuat penomoran pada proses-proses yang mengakses sumber daya. Suatu proses dimungkinkan untuk dapat meminta sumber daya kapan pun, tetapi permintaannya harus dibuat terurut.

Masalah yang mungkin terjadi dengan mengatur bahwa setiap proses hanya dapat memiliki satu proses adalah bahwa tidak semua proses hanya membutuhkan satu sumber daya, untuk suatu proses yang kompleks dibutuhkan banyak sumber daya pada saat yang bersamaan. Sedangkan dengan penomoran masalah yang dihadapi adalah tidak terdapatnya suatu penomoran yang dapat memuaskan semua pihak.

Secara ringkas pendekatan yang digunakan pada metode pencegahan deadlock dan masalah-masalah yang menghambatnya, terangkum dalam tabel di bawah ini.

Tabel. Tabel Deadlock

Syarat	Langkah	Kelemahan

Mutual Exclusion	*Spooling* sumber daya	Dapat menyebabkan *chaos*
Hold and Wait	Meminta sumber daya di awal	Sulit memperkirakan di awal dan tidak optimal
No Pre-emptive	Mengambil sumber daya di tengah proses	Hasil proses tidak akan baik
Circular Wait	Penomoran permintaan sumber daya	Tidak ada penomoran yang memuaskan semua pihak

Menghindari *Deadlock*

Pendekatan metode ini adalah dengan hanya memberi kesempatan ke permintaan sumber daya yang tidak mungkin akan menyebabkan *deadlock*. Metode ini memeriksa dampak pemberian akses pada suatu proses, jika pemberian akses tidak mungkin menuju kepada *deadlock*, maka sumber daya akan diberikan pada proses yang meminta. Jika tidak aman, proses yang meminta akan di-*suspend* sampai suatu waktu permintaannya aman untuk diberikan. Kondisi ini terjadi ketika setelah sumber daya yang sebelumnya dipegang oleh proses lain telah dilepaskan.

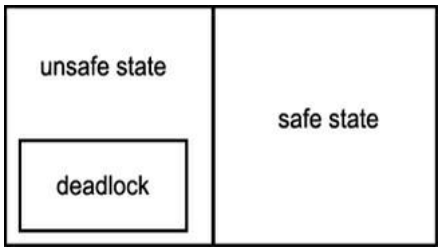
Kondisi aman yang dimaksudkan selanjutnya disebut sebagai *safe-state*, sedangkan keadaan yang tidak memungkinkan untuk diberikan sumber daya yang diminta disebut *unsafe-state*.

Kondisi Aman (*Safe state*)

Suatu keadaan dapat dinyatakan sebagai *safe state* jika tidak terjadi *deadlock* dan terdapat cara untuk memenuhi semua permintaan sumber daya yang ditunda tanpa menghasilkan *deadlock*. Dengan cara mengikuti urutan tertentu.

Kondisi Tak Aman (*Unsafe state*)

Suatu *state* dinyatakan sebagai *state* tak selamat (*unsafe state*) jika tidak terdapat cara untuk memenuhi semua permintaaan yang saat ini ditunda dengan menjalankan proses-proses dengan suatu urutan.



Gambar. Safe.

Mendeteksi *Deadlock* dan Memulihkan *Deadlock*

Metode ini mengunakan pendekatan dengan teknik untuk menentukan apakah *deadlock* sedang terjadi serta proses-proses dan sumber daya yang terlibat dalam deadlock tersebut. Setelah kondisi *deadlock* dapat dideteksi, maka langkah pemulihan dari kondisi *deadlock* dapat segera dilakukan. Langkah pemulihan tersebut adalah dengan memperoleh sumber daya yang diperlukan oleh proses-proses yang membutuhkannya. Beberapa cara digunakan untuk mendapatkan sumber daya yang diperlukan, yaitu dengan terminasi proses dan *pre-emption* (mundur) suatu proses. Metode ini banyak digunakan pada komputer *mainframe* berukuran besar.

Terminasi Proses

Metode ini akan menghapus proses-proses yang terlibat pada kondisi *deadlock* dengan mengacu pada beberapa syarat. Beberapa syarat yang termasuk dalam metode ini adalah, sebagai berikut:

- Menghapus semua proses yang terlibat dalam kondisi *deadlock* (solusi ini terlalu mahal).
- Menghapus satu persatu proses yang terlibat, sampai kondisi *deadlock* dapat diatasi (memakan banyak waktu).
- Menghapus proses berdasarkan prioritas, waktu eksekusi, waktu untuk selesai, dan kedalaman dari *rollback*.

Resources Preemption

Metode ini lebih menekankan kepada bagaimana menghambat suatu proses dan sumber daya, agar tidak terjebak pada *unsafe condition*.

Beberapa langkahnya, yaitu:

- Pilih salah satu - proses dan sumber daya yang akan di-*preempt*.

- *Rollback* ke *safe state* yang sebelumnya telah terjadi.

- Mencegah suatu proses agar tidak terjebak pada *starvation* karena metode ini.

Contoh soal:

1. Suatu sistem komputer mempunyai 2 buah sumber daya yaitu sumber daya A (RA) sebanyak 10 buah, sumber daya B (RB) sebanyak 11 buah. Sumber daya tersebut akan digunakan oleh 4 buah proses yaitu P1, P2, P3 dan P4 dengan alokasi dan kebutuhan sumber daya sebagai berikut :

Proses	Alokasi (digenggam)		Kebutuhan maksimum	
	A	B	A	B
P_1	3	2	7	4
P_2	2	3	3	7
P_3	1	0	4	3
P_4	1	3	6	7

   A. Buatlah matriks kebutuhan untuk proses-proses tersebut.
   B. Buatlah urutan penanganan proses sehingga sistem komputer dalam keadaan *safe* (tidak terjadi *deadlock*).

Jawab :

A. Tabel kebutuhan dari proses tersebut adalah :

Proses	**Sumber Daya**	
	A	**B**
P_1	7-3=4	4-2=2
P_2	3-2=1	7-3=4
P_3	3	3
P_4	5	4

B. Urutan penganan proses sehingga tidak terjadi deadlock adalah :

Proses	**Alokasi (digenggam)**		**Kebutuhan maksimum**	
	A	**B**	**A**	**B**
P_1	3	2	7	4
P_2	2	3	3	7
P_3	1	0	4	3
P_4	1	3	6	7
Sumber daya tersedia	3	3		

Misalkan sumber daya tersedia tersebut diberikan kepada proses P3 maka kondisinya menjadi :

Proses	Alokasi (digenggam)		Kebutuhan maksimum	
	A	B	A	B
P_1	3	2	7	4
P_2	2	3	3	7

P_3	4	3	4	3
P_4	1	3	6	7

Setelah P3 selesai maka kondisi menjadi :

Proses	**Alokasi (digenggam)**		**Kebutuhan maksimum**	
	A	**B**	**A**	**B**
P_1	3	2	7	4
P_2	2	3	3	7
P_3	-	-	-	-
P_4	1	3	6	7
Sumber daya tersedia	4	3		

Misalkan sumber daya tersedia tersebut diberikan kepada proses P1 (4, 2) maka kondisinya menjadi :

Proses	Alokasi (digenggam)		Kebutuhan maksimum	
	A	B	A	B
P_1	7	4	7	4

P_2	2	3	3	7
P_3	-	-	-	-
P_4	1	3	6	7

Setelah P1 selesai maka kondisi menjadi :

Proses	**Alokasi (digenggam)**		**Kebutuhan maksimum**	
	A	**B**	**A**	**B**
P_1	-	-	-	-
P_2	2	3	3	7
P_3	-	-	-	-
P_4	1	3	6	7
Sumber daya tersedia	7	5		

Misalkan sumber daya tersedia tersebut diberikan kepada proses P2 (1, 4) maka kondisinya menjadi :

Proses	Alokasi (digenggam)		Kebutuhan maksimum	
	A	B	A	B
P_1	-	-	-	-

P_2	3	7	3	7
P_3	-	-	-	-
P_4	1	3	6	7

Setelah P2 selesai maka kondisi menjadi :

Proses	**Alokasi (digenggam)**		**Kebutuhan maksimum**	
	A	**B**	**A**	**B**
P_1	-	-	-	-
P_2	-	-	-	-
P_3	-	-	-	-
P_4	1	3	6	7
Sumber daya tersedia	9	8		

Sisa sumber daya yang tersedia dapat diberikan kepada proses P4 untuk menyelesaiakan proses tersebut. Jadi urutan penanganan proses supaya tidak terjadi deadlock adalah P3 – P1 – P2 – P4 atau P3 – P1 – P4 – P2

Daftar Pustaka

1. Sri Kusuma Dewi, Sistem Operasi, Graha Ilmu, 2002.
2. William Stallings, Sistem Operasi Edisi Empat, PT. Indeks, Kelompok Gramedia, 2005.
3. Hariyanto, B.,"Sistem Operasi", Bandung: Informatika, Desember 1997
4. Silberschatz, A., Gagne, G. dan Galvin, P., "*Applied Operating System Concept*", John Wiley and Sons Inc., 2000

MODUL PERKULIAHAN

MANAJEMEN MEMORI

Agar proses dapat berjalan dengan efisien maka data dan program harus tersimpan dalam lokasi yang sama yaitu memori.

Abstract

Setiap proses yang akan dijalankan, harus melalui memori terlebih dahulu. Memori harus digunakan dengan baik, sehingga dapat memuat banyak proses dalam suatu waktu.

Kompetensi

Memahami penerapan partisi statis, partisi dimanis.

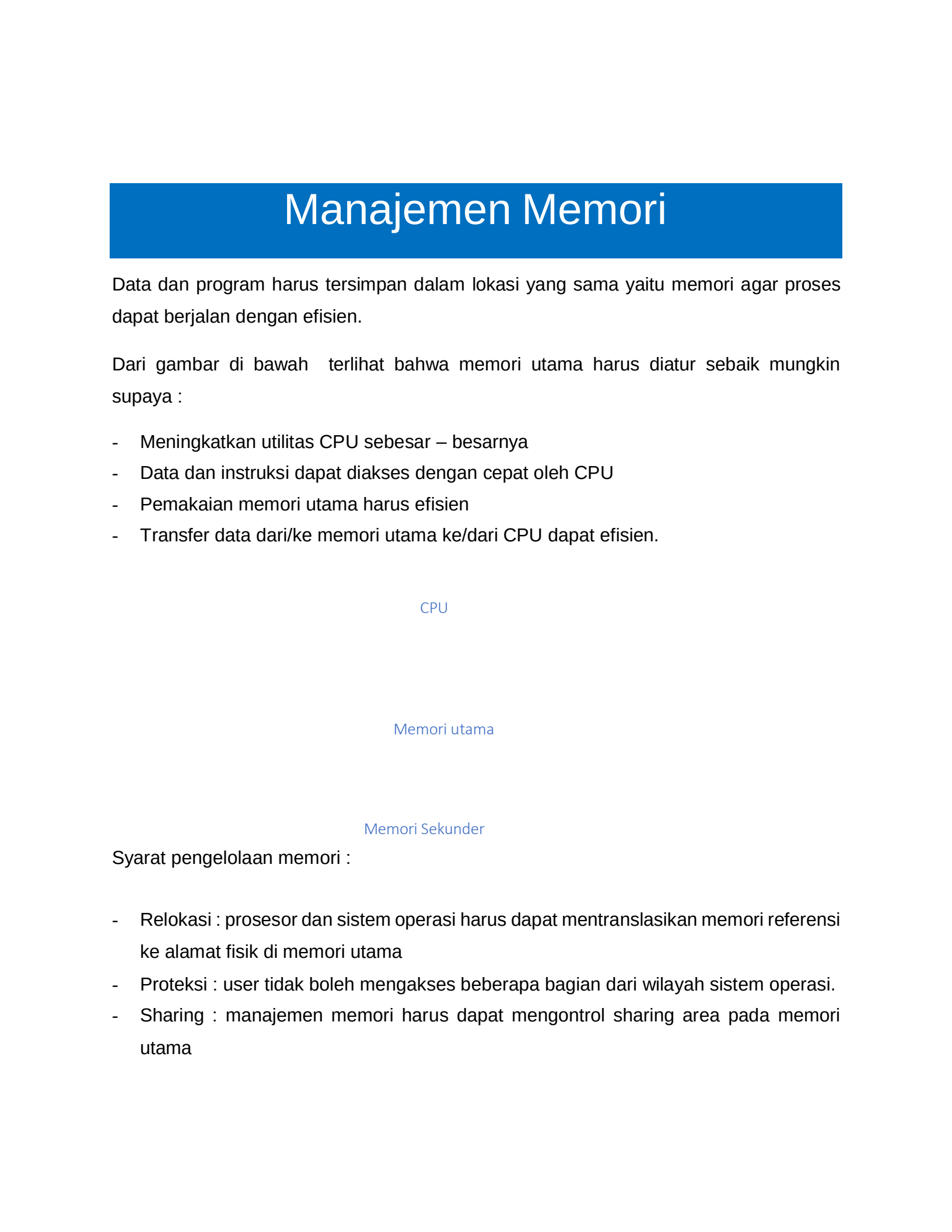
Memahami system paging dan segmentasi.

Manajemen Memori

Data dan program harus tersimpan dalam lokasi yang sama yaitu memori agar proses dapat berjalan dengan efisien.

Dari gambar di bawah terlihat bahwa memori utama harus diatur sebaik mungkin supaya :

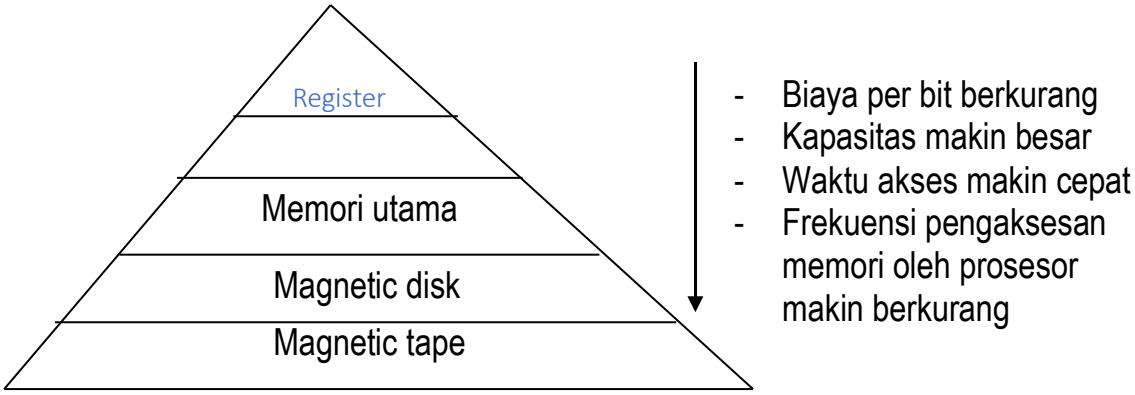
- Meningkatkan utilitas CPU sebesar – besarnya
- Data dan instruksi dapat diakses dengan cepat oleh CPU
- Pemakaian memori utama harus efisien
- Transfer data dari/ke memori utama ke/dari CPU dapat efisien.



Syarat pengelolaan memori :

- Relokasi : prosesor dan sistem operasi harus dapat mentranslasikan memori referensi ke alamat fisik di memori utama
- Proteksi : user tidak boleh mengakses beberapa bagian dari wilayah sistem operasi.
- Sharing : manajemen memori harus dapat mengontrol sharing area pada memori utama

- Organisasi logika : sistem operasi dan hardware diusahakan untuk dapat berhubungan dengan user program dalam satu modul.
- Organisasi fisik : harus ada pengaturan yang jelas antara memori utama dengan memori sekunder pada long term schedulling.

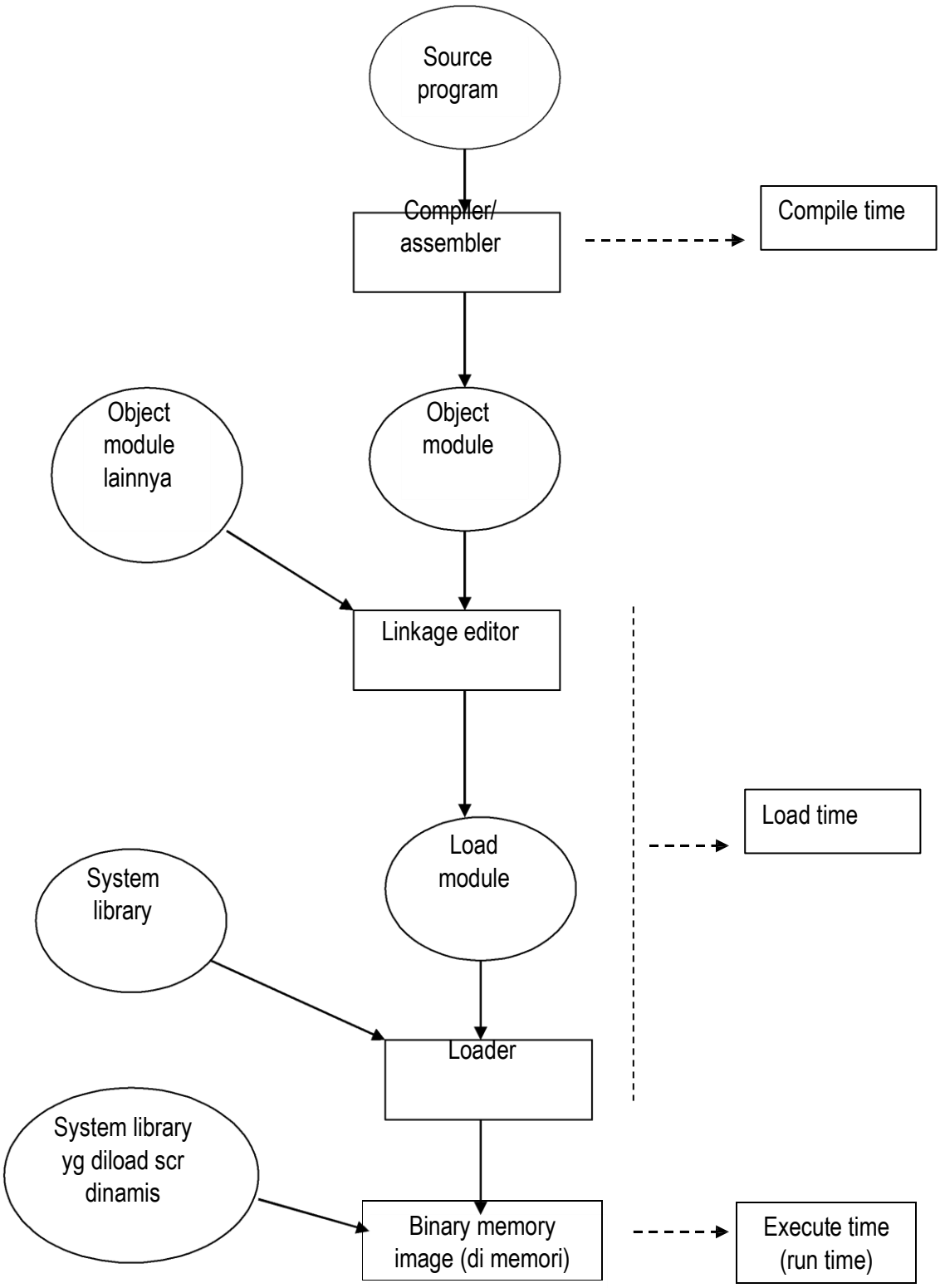


Hirarki pengelolaan memori

a. Konsep binding

Binding adalah proses penempatan suatu item ke dalam lokasi memori tertentu. Binding dapat terjadi pada :

- Compile time : bila letak dan alamat memori sudah diketahui sebelumnya. Contoh: program DOS dengan eksestensi COM.
- Load time : alamat awal belum diketahui. Saat program dibuat yang ada hanya konstanta, variabel dan instruksi, pada saat di compile maka urutan dari konstanta, variabel dan instruksi tersebut adalah tetap. Saat di-link, sejumlah modul disusun lebih teratur dan pada saat akan dieksekusi program diletakkan secara fisik sesuai dengan lokasi yang tersedia.
- Execute time : memungkinkan suatu proses dapat berpindah alamatnya.


Source program
Compiler/ assembler
Compile time
Object module lainnya
Object module
Linkage editor
Load time
Load module
System library
Loader
System library yg diload scr dinamis
Binary memory image (di memori)
Execute time (run time)

Pemrosesan user program

b. Dynamic loading

Tidak semua bagian program diambil dari memori, hanya bagian yang diperlukan saja yang harus tetap tinggal di memori, rutin – rutin yang dibutuhkan baru dipanggil saat program memerlukan rutin tersebut. Sistem operasi tidak ikut campur dalam dynamic loading tetai hanya menyediakan librarynya.

c. Dynamic linking

Konsep sama dengan dynamic loading, bedanya dilakukan pada saat link. Dengan adanya dynamic link tidak perlu membuat duplikat rutin yang banyak, cukup dilakukan sharing. Dynamic linking memerlukan campur tangan sistem operasi yaitu dalam hal pengecekan rutin – rutin yang diminta proses di memori sistem operasi dapat mengijinkan beberapa proses untuk mengakses memori pada alamat yang sama.

d. Overlay

Membagi program yang besar menjadi bagian yang lebih kecil sehingga dapat dimuat dalam memori utama. Yang ada di memori utama adalah program penggerak sedang bagian pendukungnya berada pada memori sekunder.

1. Strategi manajemen memori

Beberapa hal yang harus dipertimbangkan dalam manajemen memori :

a. Dukungan hardware :

- untuk single dan multiple partition : menggunakan base register (pasangan antara base dan limit register)
- untuk paging dan segmentasi : perlu tabel pemetaan alamat.

b. Kinerja :

- untuk yang sederhana : dibutuhkan untuk membandingkan atau menambah alamt logika (proses cepat)
- untuk paging dan segmentasi : proses akan cepat apabila tabel menggunakan register cepat

c. Fragmentation :

- untuk single partition dan paging : ada masalah dengan internal fragmentation
- untuk multiple partition dan segmentasi : masalah pada eksternal fragmentation

d. Relokasi : untuk mengatasi eksternal fragmentasi dengan cara compaction (mengijinkan suatu program digeser di memori tanpa ada perubahan pada program tersebut).

e. Swapping :

- Semua algoritma manajemen memori pasti mengalaminya
- Diatur oleh sistem operasi melalui kebijaksanaan CPU-schedulling

f. Sharing : umumnya menggunakan sistem paging atau segmentasi.

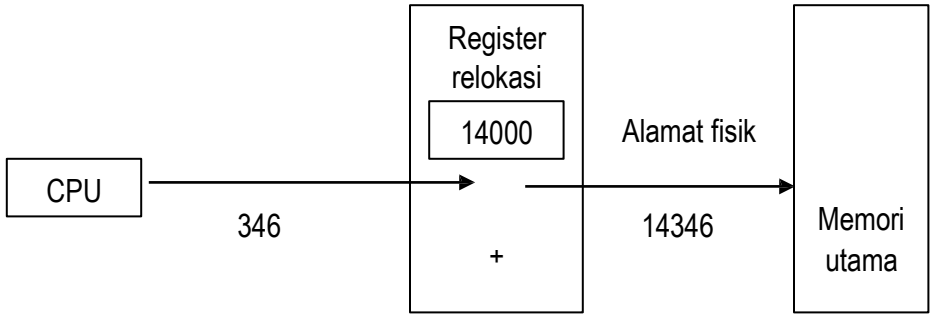
g. Proteksi : pentingnya batasan pada seksi (rutin) dalam bentuk read-only, execute-only dan read-write.

2. Ruang alamat logika dan fisik

- alamat logika : diturunkan oleh CPU
- alamat fisik : diturunkan oleh memori.

Pada saat execute kedua alamat tersebut harus sama, jika terjadi perbedaan maka di CPU dibuat alamat maya (virtual address).

Relokasi dinamis pengalokasian alamat oleh memori manajemen unit (MMU) dapat digambarkan sebagai berikut :



3. **Swapping**

Suatu proses dapat diambil alih sementara dari memori ke suatu tempat penyimpanan, swapping terjadi pada multiprogramming dengan partisi dinamis, sedang pada monoprogramming dan multiprogramming dengan partisi statis tidak ada swapping.

4. Pencatatan pemakaian memori

Tujuan untuk mengetahui memori utama yang masih kosong dan yang sudah terisi. Ada dua cara pencatatan pemakaian memori yaitu :

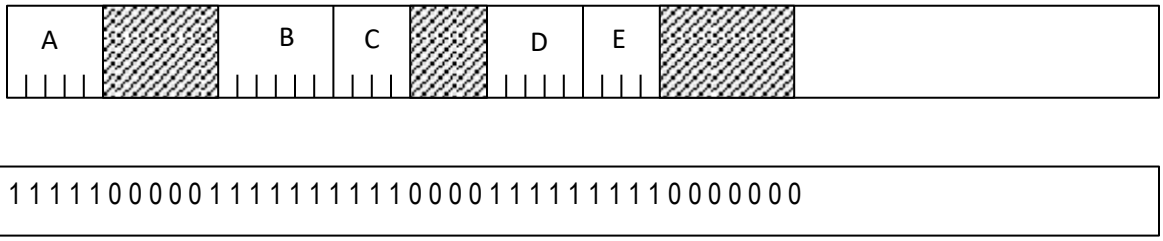
a. Peta bit (bit map)

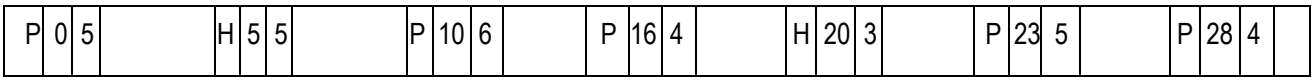
Memori dibagi menjadi beberapa alokasi unit, tiap unit terdiri dari beberapa word atau kilobyte. Setiap unit berhubungan dengan 1 bit yang akan bernilai 0 bila masih kosong dan 1 bila sudah terisi. Ukuran dari unit sangat penting karena :

- ukuran unit makin kecil, maka bitmap akan makin besar
- ukuran unit besar, tidak efisien

b. Linked list

Setiap node list terdiri atas informasi yang menyatakan adanya proses (P) atau hole (H), lokasi awal dan panjang lokasi. Memori yang digunakan lebih kecil, tidak memerlukan penghitungan blok lubang memori yang sudah tercatat, sulit dealokasi karena terjadi penggabungan beberapa mode.





Pencatatan memori dengan Bit map

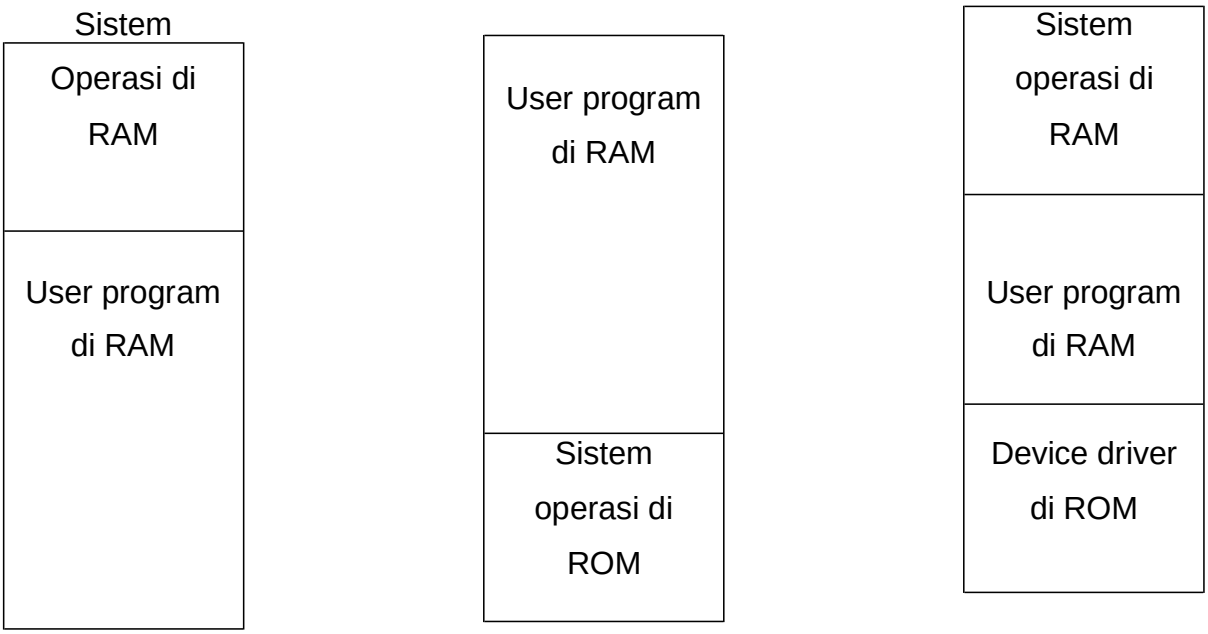
Pencatatan memori dengan linked list

5. Monoprogramming

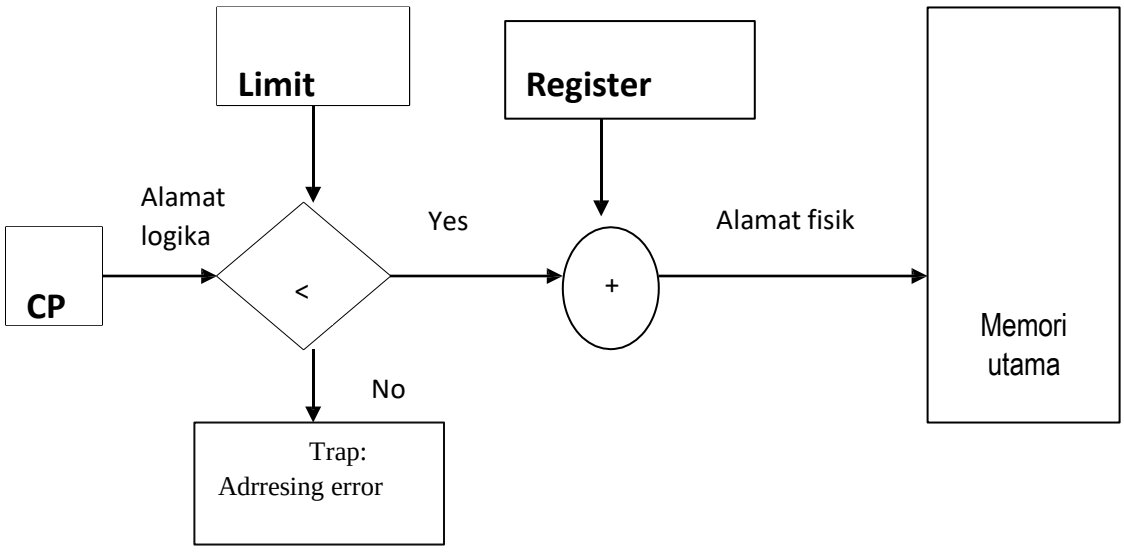
Tiga cara alokasi memori untuk monoprogramming :

a. Sistem operasi dan user program ditempatkan pada RAM

b. Sistem operasi diletakkan di ROM dan user program diletakkan di RAM
c. Sistem operasi diletakkan di RAM alamat bawah, user program di atasnya dan device driver diletakkan di ROM dalam bentuk ROM BIOS.



Jika sistem operasi diletakkan pada lokasi yang lebih rendah sedang user diletakkan pada lokasi yang lebih tinggi maka perlu adanya proteksi terhadap memori dengan menggunakan base register dan limit register.



Daftar Pustaka

5. Sri Kusuma Dewi, Sistem Operasi, Graha Ilmu, 2002.
6. William Stallings, Sistem Operasi Edisi Empat, PT. Indeks, Kelompok Gramedia, 2005.
7. Hariyanto, B.,"Sistem Operasi", Bandung: Informatika, Desember 1997
8. Silberschatz, A., Gagne, G. dan Galvin, P., "*Applied Operating System Concept*", John Wiley and Sons Inc., 2000

MODUL PERKULIAHAN

MANAJEMEN MEMORI

Agar proses dapat berjalan dengan efisien maka data dan program harus tersimpan dalam lokasi yang sama yaitu memori.

Abstract **Kompetensi**

Multiprogramming dengan partisi statis terdiri atas partisi statis dengan ukuran sama dan partisi statis dengan ukuran berbeda.

Memahami konsep strategi pengalokasian memori.
Memahami penerapan partisi statis, partisi dimanis.
Memahami system paging dan segmentasi.

Manajemen Memori (Lanjutan)

1. **Pengalokasian berurutan.**

   Pada multiprogrraming biasanya memori utama dibagi menjadi 2 bagian yaitu alokasi untuk residen sistem operasi (ditempatkan pada bagian atas atau bawah) dan sisanya untuk user. Pengalokasian secara berurutan terbagi atas : **multiprogramming dengan partisi statis, multiprogramming dengan partisi dinamis dan buddy sistem.**

a. Multiprogramming dengan partisi statis

Alasan digunakan multiprogramming :

- mempermudah pemrograman
- agar dapat memberi pelayanan interaktif ke beberapa user secara simultan
- efisien penggunaan sumber daya
- eksekusi lebih murah jika proses dipecah menjadi beberapa proses kecil
- dapat mengerjakan sejumlah job secara simultan

Multiprogramming dengan partisi statis terdiri atas partisi statis dengan **ukuran sama dan partisi statis dengan ukuran berbeda.**

Partisi statis dengan ukuran sama : ukuran dari bagian dibuat sama, mempunyai kelemahan :

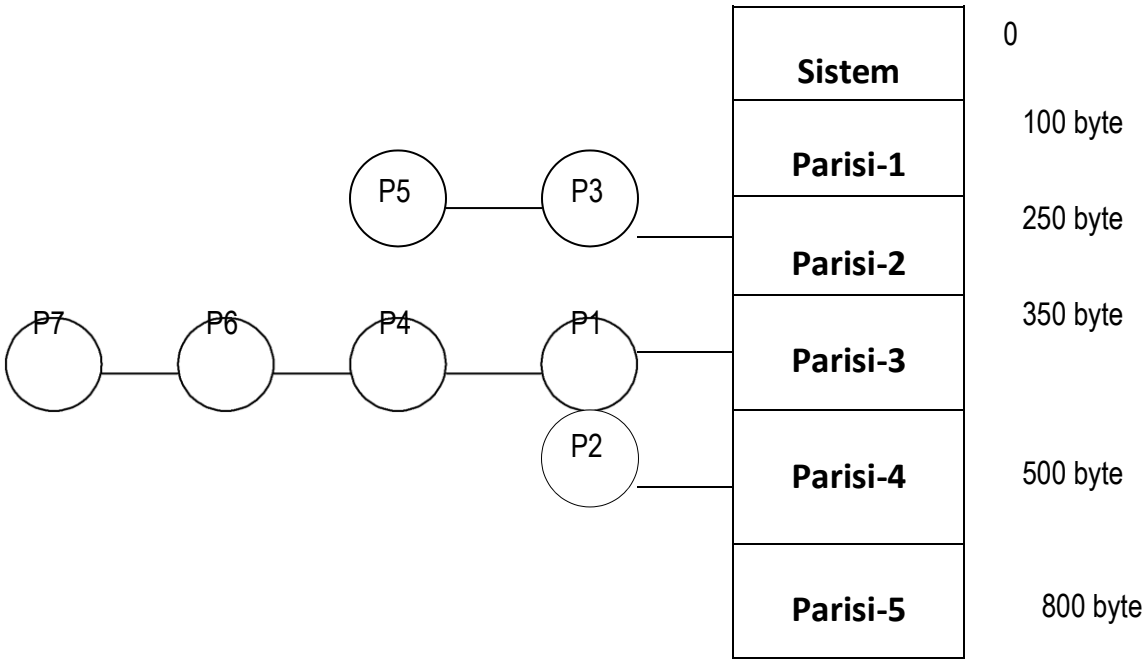
- bila program ukurannya lebih besar maka tidak dapat dimuatkan sehingga tidak dapat dijalankan. Untuk itu program tersebut akan mengalami overlay, sehingga diperlukan sistem operasi yang mendukung swapping.
- Bila program ukuran jauh lebih kecil dari partisi maka akan banyak ruang kosong sehingga tidak efisien, perlu fragmentasi internal.

Partisi statis dengan ukuran berbeda

Terdapat dua strategi untuk pengalokasian memori :

- **Banyak antrian (satu antrian untuk tiap partisi)**
  Kelemahan : ada partisi yang antriannya kosong, disisi lain ada partisi yang antriannya penuh

  Keuntungan : meminimalkan pemborosan

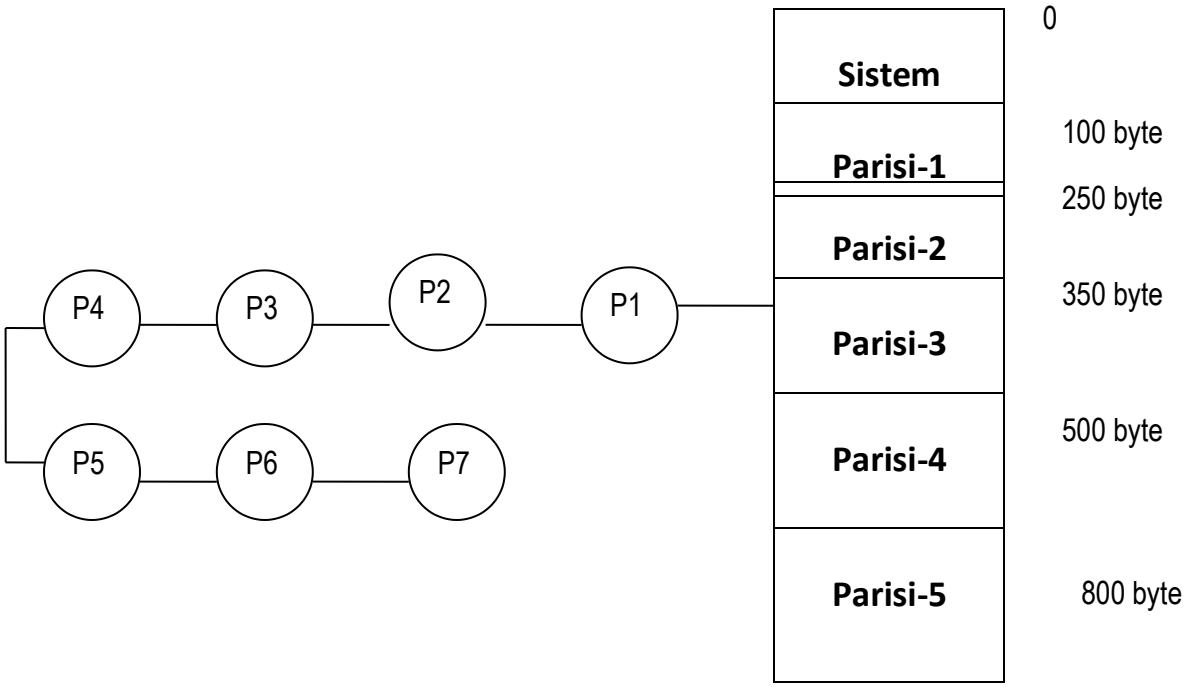


Partisi statis ukuran berbeda & banyak antrian

- **Satu antrian untuk seluruh partisi**

  Kelemahan : pemborosan (ada proses kecil yang ditempatkan pada partisi besar)

  Keuntungan : lebih fleksibel, implementasi dan sistem operasi lebih minimal karena hanya mengelola satu antrian.



Partisi statis ukuran berbeda & satu antrian

b. Multiprogramming dengan partisi dinamis

partisi baru dibuat setelah proses masuk ke memori utama.

Kelemahan :

- terjadi lubang - lubang kecil memori diantara partisi yang dipakai (terjadi eksternal fragmentation yaitu terdapat lubang yang cukup besar tetapi letaknya tidak berdekatan sehingga tidak dapat dipergunakan).
- merumitkan alokasi dan dealokasi memori

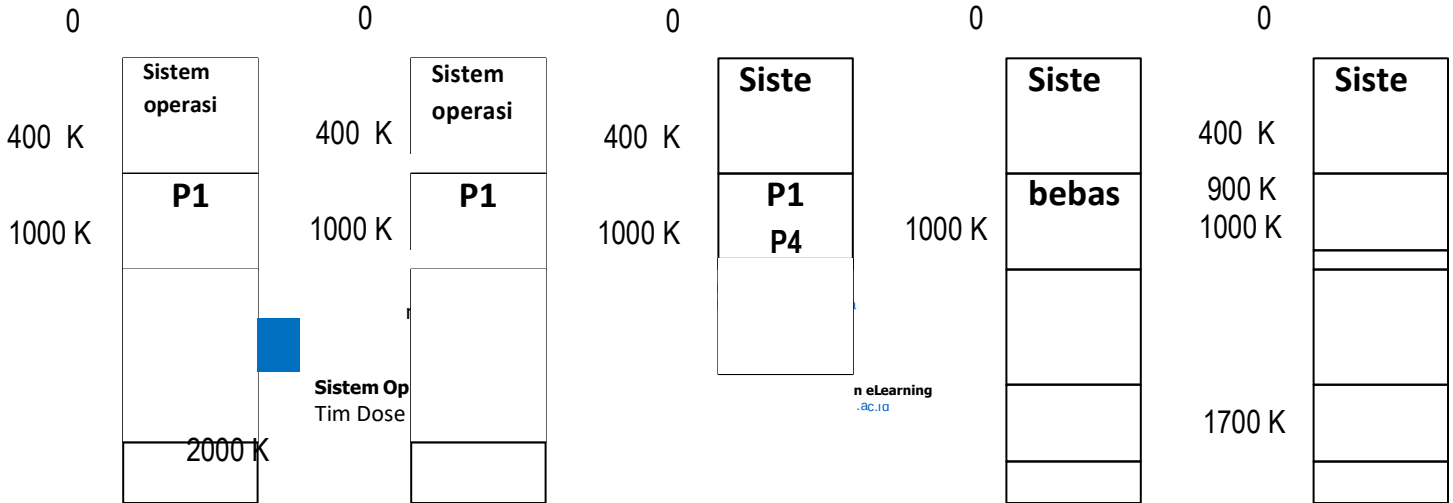
  Keuntungan : efisien karena proses akan ditempatkan sesuai dengan ukurannya

Contoh : terdapat 5 proses dengan ukuran memori dan waktu yang diperlukan seperti tabel berikut :



Job Queue		
Proses	Memori	Waktu
P1	600 K	10
P2	1000 K	5
P3	300 K	20
P4	700 K	8
P5	500 K	15

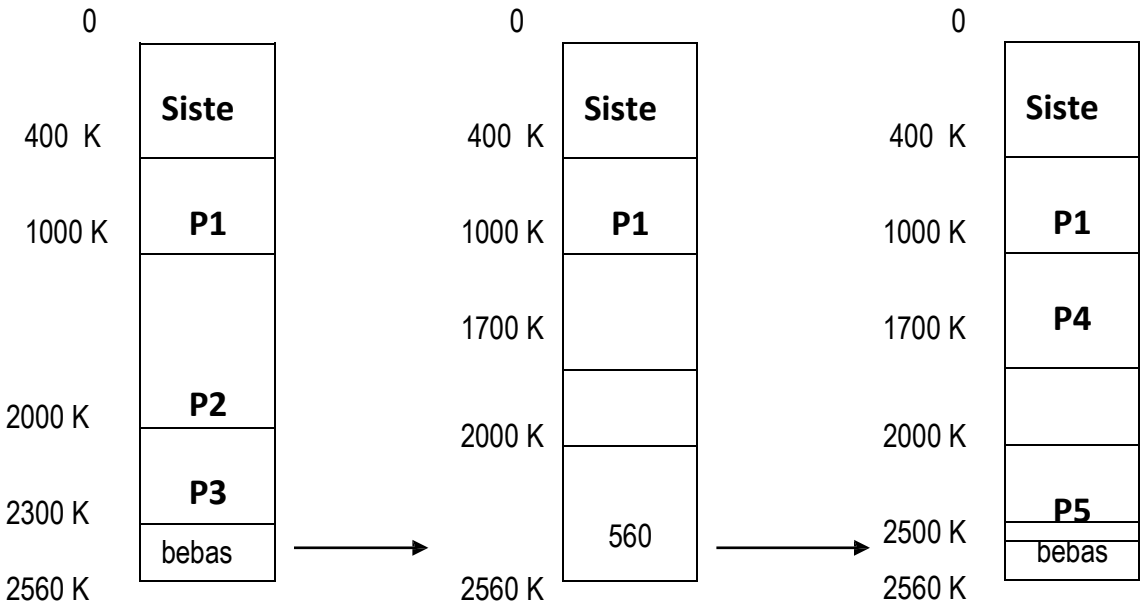
Alokasi memori dan long term scheduling dari proses di atas dapat dilihat berikut ini :

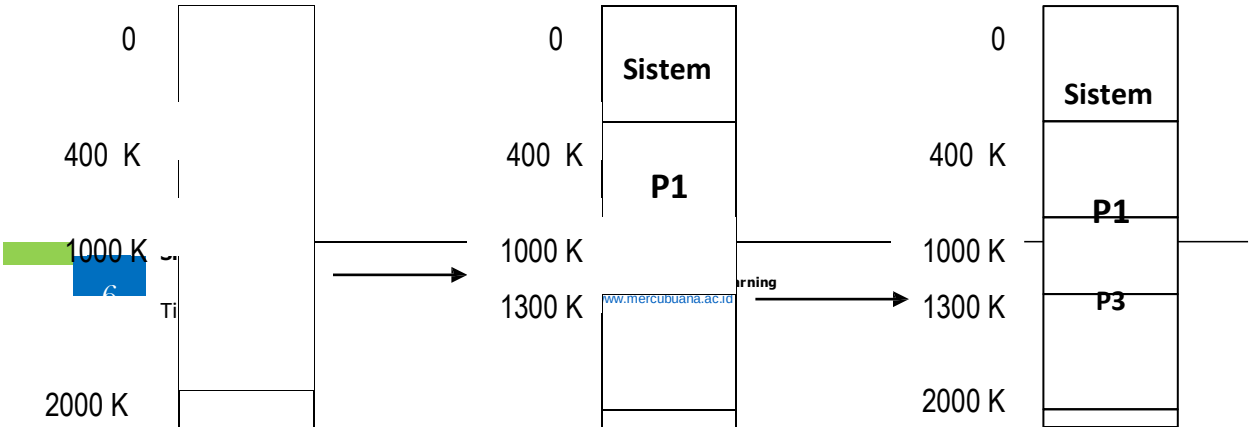


P2	1700 K	1700 K		bebas	bebas
2000 K					
P2	2000 K	2000 K		2000 K	

bebas

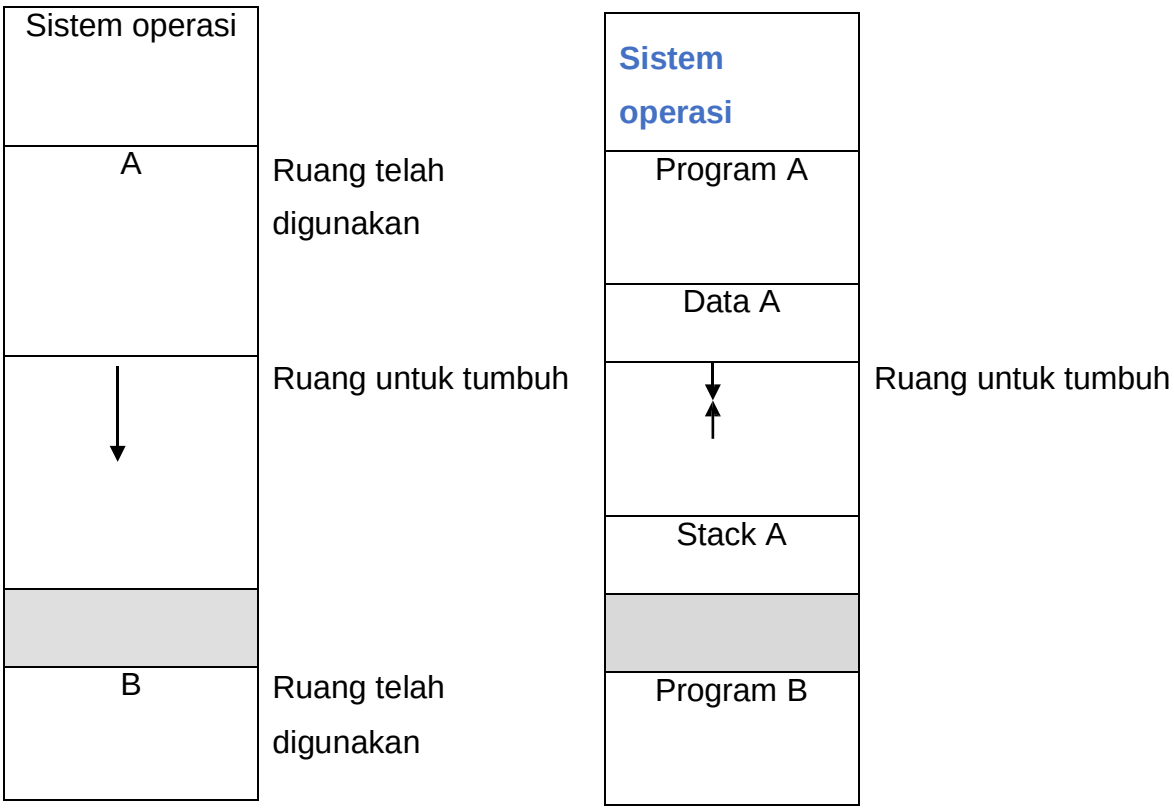
Cara penyelesaian eksternal fragmentation dengan compaction yaitu penempatan ulang proses – proses yang ada di memori utama, seperti dapat dilihat berikut ini :

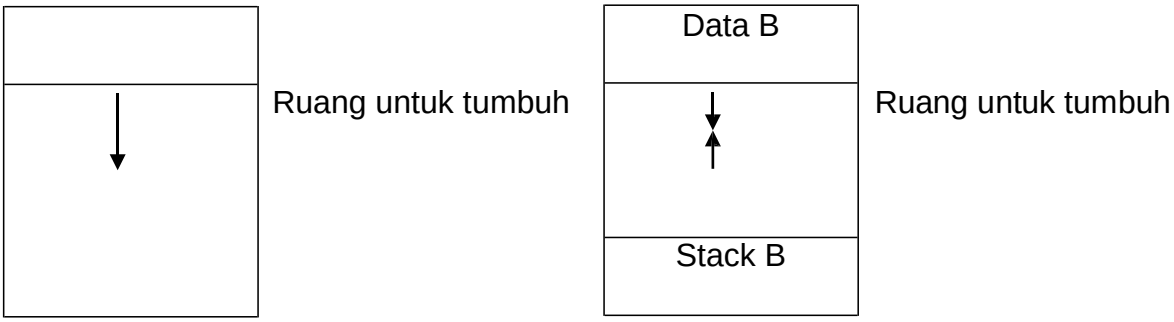




2000 K

Masalah lain yang timbul dari partisi dinamis yaitu adanya segmen data proses yang terus berkembang sebagai akibat adanya heap atau stack yang memanggil prosedur atau variabel lokal, misalkan ada dua proses A dan B yang akan berkembang seperti pada gambar berikut :





Ada 5 strategi pengalokasian memori untuk mencari posisi yang tepat :

1. **First-fit** : pencarian dimulai dari awal dan akan berhenti jika ditemukan lokasi pertama yang cukup besar untuk penempatan proses tersebut.
2. **Next-fit** : sama dengan first-fit bedanya pencarian dimulai dari lokasi terakhir kali menemukan segmen yang cocok dan akan berhenti jika ditemukan lokasi pertama yang cukup besar untuk menempatkan proses tersebut.
3. **Best-fit** : pencarian dimulai dari awal dan akan berhenti jika ditemukan lokasi terkecil pertama yang cukup untuk menempatkan proses tersebut.
4. **Worst-fit** : pencarian dimulai dari awal dan akan berhenti jika ditemukan lokasi yang besar yang cukup untuk menempatkan proses tersebut.
5. **Quick-fit** : cocok untuk pencatatan dengan linked list.

c. Sistem Buddy

Merupakan cara mengelola memori utama dengan memanfaatkan kelebihan penggunaan bilangan biner (2^k; k = 1, 2, 3,…….)

Contoh : suatu memori utama pada awalnya mempunyai 1 Mbyte. Terdapat beberapa proses yang akan menggunakan memori tersebut, missal Proses A ukuran 80 kb, proses B ukuran 35 kb, proses C ukuran 85 kb dan proses D ukuran 122 kb.

Maka penanganan keempat proses tersebut adalah : untuk proses A dengan ukuran 80 Kbyte maka permintaan tersebut akan dialokasikan ke lokasi terdekat yaitu 128 kbyte. Karena tidak tersedia lokasi tersebut maka memori utama akan dibagi menjadi

2 x 512 Kbyte, dibagi lagi menjadi 2 x 256 Kbyte, dibagi lagi menjadi 2 x 128 Kbyte. Sehingga proses A dengan ukuran 80 kbyte tadi akan menggunakan lokasi 128 Kbyte, begitu juga untuk proses yang lainnya. Terlihat bahwa pada penanganan proses A akan menyisakan 48 Kbyte yang tidak terpakai. Sisa ini disebut **internal fragmentation**. Untuk lebih jelasnya dapat dilihat pada gambar berikut :

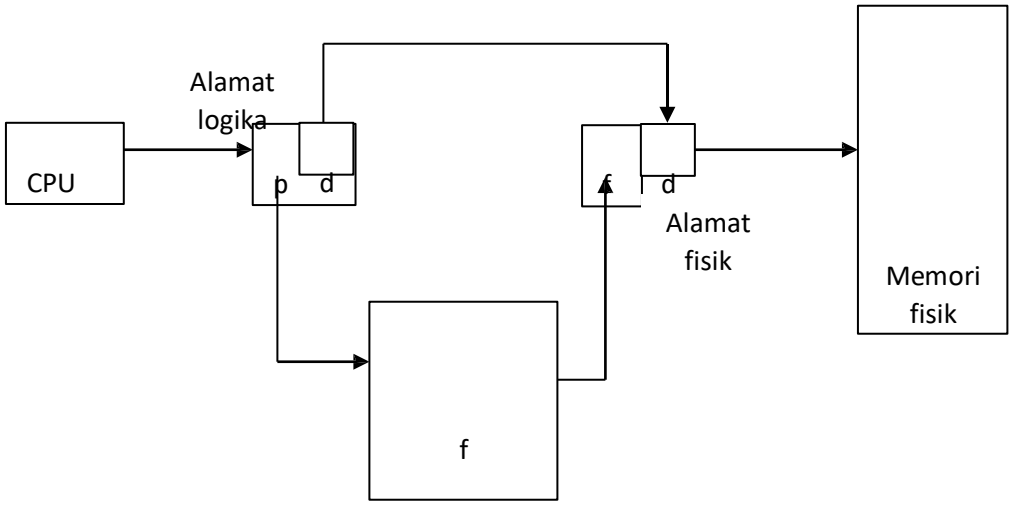
<table>
<tr><td colspan="5">Ukuran memori semula</td></tr>
<tr><td>A</td><td colspan="2"></td><td colspan="2"></td><td>Alokasi memori utk proses A</td></tr>
<tr><td>A</td><td>B</td><td></td><td colspan="2"></td><td>Alokasi memori utk proses B</td></tr>
<tr><td>A</td><td>B</td><td></td><td>C</td><td></td><td>Alokasi memori utk proses C</td></tr>
<tr><td></td><td>B</td><td></td><td>C</td><td></td><td>Dealokasi memori utk proses A</td></tr>
<tr><td></td><td>B</td><td>D</td><td>C</td><td></td><td>Alokasi memori utk proses D</td></tr>
<tr><td colspan="2"></td><td>D</td><td>C</td><td></td><td>Dealokasi memori utk proses B</td></tr>
<tr><td colspan="2"></td><td>D</td><td>C</td><td></td><td></td></tr>
<tr><td colspan="3"></td><td>C</td><td></td><td>Dealokasi memori utk proses D</td></tr>
<tr><td colspan="3"></td><td></td><td></td><td>Dealokasi memori utk proses C</td></tr>
<tr><td>0 128</td><td colspan="2">256</td><td colspan="2">512</td><td>1 MB</td></tr>
</table>

2. Pengalokasian Tak berurutan

a. Sistem Paging

Salah satu cara untuk mengatasi external fragmentation adalah membentuk teknik pengalokasian noncontinous. Konsep dasarnya adalah membagi **memori fisik**

menjadi blok – blok dengan ukuran tertentu yaitu **frame (f)**. **Memori logika** dibagi atas atas beberapa **page**. Setiap alamat yang diberikan oleh CPU dibagi menjadi **nomor page (p)** dan **page offset (d)**.



Page table

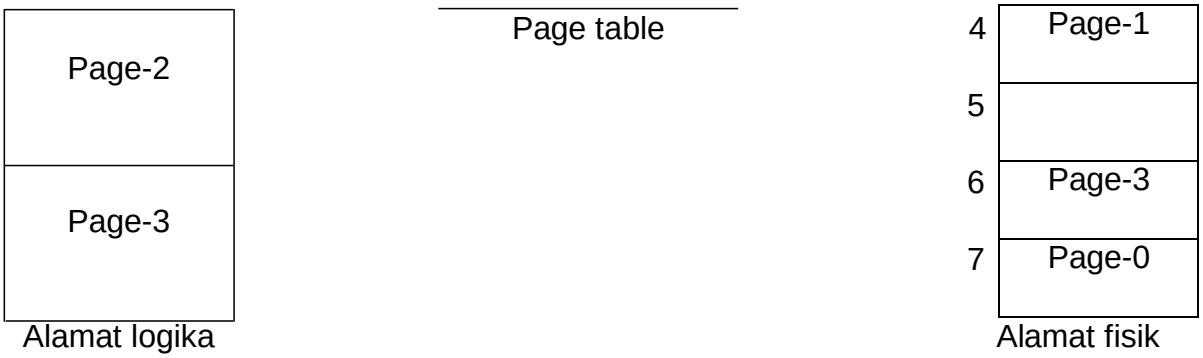
Contoh : Misal suatu memori mempunyai ukuran page = 4 byte, setiap page berisi 4 offset dan memori fisik = 32 byte (8 page) seperti pada gambar berikut :

Page-0

Page-1

0	7
1	4
2	1
3	6

0	
1	Page-2
2	
3	



SOAL:

1. Suatu aplikasi mempunyai alamat logika 0 ada di page 0 dan offset 0, tentukan alamat fisiknya.
   Alamat fisik = ((nomor frame x ukuran fisik) + alamat offset)

   = ((7 x 4) + 0) = 28
2. Suatu aplikasi mempunyai alamat logika 3 ada di page 3 dan offset 1, tentukan alamat fisiknya.
   Alamat fisik = ((nomor frame x ukuran fisik) + alamat offset)

   = ((6 x 4) + 1) = 25
3. Suatu aplikasi mempunyai alamat logika 1 ada di page 1 dan offset 3, tentukan alamat fisiknya.
   Alamat fisik = ((nomor frame x ukuran fisik) + alamat offset)

   = ((4 x 4) + 3) = 19

b. Segmentasi

Konsep dasar memori utama sebagai kumpulan segmen dengan ukuran yang berbeda –beda. Segmentasi adalah skema pengaturan memori yang mendukung user untuk melihat memori tersebut.

Daftar Pustaka

9. Sri Kusuma Dewi, Sistem Operasi, Graha Ilmu, 2002.
10. William Stallings, Sistem Operasi Edisi Empat, PT. Indeks, Kelompok Gramedia, 2005.
11. Hariyanto, B.,"Sistem Operasi", Bandung: Informatika, Desember 1997
12. Silberschatz, A., Gagne, G. dan Galvin, P., "*Applied Operating System Concept*", John Wiley and Sons Inc., 2000

MODUL PERKULIAHAN

MANAJEMEN PERANGKAT I/O

Manajemen perangkat masukan/keluaran merupakan aspek perancangan sistem operasi terluas dan kompleks karena sangat beragamnya perangkat dan aplikasinya.

Abstract

Sasaran memenuhi Kebutuhan Manajemen Data bagi pemakai (retrive, insert, delete, update), Menjamin data pada file adalah Valid, Optimasi Kinerja (Trouhgput dan waktu tanggap ditingkatkan), Menyediakan dukungan masukan/keluaran beragam tipe perangkat penyimpanan, Meminimalkan potensi kehilangan data

Kompetensi

Memahami konsep klasifikasi perangkat I/O

Memahami meknisme perangkat I/O

Manajemen Perangkat I/O

Modul XIII

MANAJEMEN PERANGKAT MASUKAN/KELUARAN

Manajemen perangkat masukan/keluaran merupakan aspek perancangan sistem operasi terluas dan kompleks karena sangat beragamnya perangkat dan aplikasinya.

Beberapa fungsi manajemen input/ouput (I/O) :

a. Mengirim perintah ke perangkat I/O agar menyediakan layanan.
b. Menangani interupsi perangkat I/O.
c. Menangani kesalahan perangkat I/O.
d. Menyediakan interface ke pemakai.

A. KLASIFIKASI PERANGKAT MASUKAN/KELUARAN

Perangkat masukan/keluaran merupakan komponen yang paling banyak jenisnya dan dapat dikelompokkan dengan beragam kriteria, yaitu:

1.Berdasarkan Sifat Aliran Datanya, terbagi menjadi 2 yaitu :

a) Perangkat berorientasi Blok (Block Stream Oriented Device).
   - Dapat menyimpan informasi/menukarkan (menerima/mengirim) informasi sebagai blok-blok berukuran tetap.
   - Tiap blok mempunyai alamat sendiri.
   - Ukuran blok mulai dari 128 byte dampai 1024 byte.
   - Ciri utamanya, yaitu: dapat membaca/menulis blok-blok secara independen,
   - arti indpenden : dapat membaca/menulis sembarang blok tanpa hrs melewati blok-blok lainnya.
   - Contoh : disk, tape, CD ROM, Optical Disk, dsb.

b) Perangkat berorientasi Aliran Karakter (*CharacterStreamOriented Device*)
   - Artinya perangkat yang mengantarkan/menerima aliran karakter tanpa peduli membentuk suatu struktur blok., contoh : terminal, line printer, pita kertas, kartu-kartu berlobang, interface jaringan, mouse dan perangkat-perangkat lain yang tidak seperti disk dapat dipandang sebagai perangkat karakter.

- Klasifikasi tersebut tidak mutlak, ada beberapa perangkat tidak termasuk ke-2 kategori tersebut, yaitu :
  a. Clock;

Tidak teralamati secara blok, juga tidak menghasilkan/menerima aliran karakter dan menyebabkan interupsi pada interval-interval yang didefinisikan.

b. Memory-Mapped Screen disebut " I/O Device Independence ".
c. Sensor-sensor.

2. Berdasarkan Sasaran Komunikasi.

Terbagi menjadi 3, yaitu :

a) Perangkat yang terbaca oleh Manusia (Human Readable Devices).
   - Untuk komunikasi dengan manusia.
   - Contoh : VDT (Video Display Terminal) terdiri dari monitor, keyboard dan mouse.

b) Perangkat yang terbaca oleh Mesin (Machine Readable Devices).
   - Untuk komunikasi dengan perangkat elektronika.
   - contoh : disk & tape, sensor , controller, dan aktuator.

c) Untuk komunikasi.
   - Untuk komunikasi dengan perangkat jarak jauh.
   - contoh : modem. [Modulasi Demodulasi = prinsip kerjanya berdasarkan data digital 0 & 1. sebagai penghubung/ pengubah antara digital dan analog]

Perbedaan pokok antara kelas perangkat I/O, antara lain:

1. Date rate.
2. Aplikasi.
3. Kompleksitas pengendalian.
4. Unit yang di transfer.
5. Representasi data.
6. Kondisi-kondisi kesalahan.

Keberagaman perangkat masukan/keluaran yang sangat besar membuat pendekatan yang seragam & konsisten dalam pandangan sistem maupun proses pemakai sangat sulit diperoleh.

B. TEKNIK PEMROGRAMAN PERANGKAT MASUKAN/KELUARAN

Ada 3 teknik berhubungan dengan (pemrograman) perangkat masukan/keluaran berdasar mekanisme hubungan pemroses dengan pengendali perangkat masukan/keluaran, yaitu:

1. Masukan/keluaran terprogram (Programmed I/O) atau polling system.
2. Masukan/keluaran dikendalikan interupsi (Interupt Driven I/O).
3. Dengan DMA (Direct Memory Access).

1. Masukan/keluaran terprogram.

- Ketika perangkat masukan/keluaran menangani permintaan, maka perangkat men-set bit status di register status perangkat.
- Perangkat tidak memberitahukan ke pemroses saat tugas telah selesai dilakukan, oleh karena itu pemrosesan harus selalu memeriksa register status perangkat secara periodik & melakukan tindakan berdasar status yang dibaca.
- Perangkat lunak pada pengendali perangkat (driver) di pemroses harus mentransfer data ke/dari perangkat & menunggu sampai operasi yang dilakukan perangkat selesai.

Driver harus berisi kumpulan instruksi di 3 kategori, yaitu :

a. Pengendalian (instruksi untuk mengendalikan operasi perangkat keras).
   - Fungsinya untuk mengaktifkan perangkat eksternal & memberitahukan yang perlu dilakukan oleh perangkat.
   - contoh : unit tape magnetik diinstruksikan untuk kembali ke posisi awal , bergerak ke record berikutnya, dsb.

b. Pengujian (instuksi untuk memeriksa status perangkat keras).
   - Instruksi ini untuk memeriksa kondisi status berkaitan dengan perangkat masukan/keluaran.

c. Pembacaan/penulisan (instruksi untuk membaca/menulis dari atau ke Perangkat keras).
    - Instuksi ini untuk mentransfer data antara register pemroses & perangakat eksternal.

Kelemahan:

Pemborosan waktu dalam pemrosesan. (=harus menunggu proses satu selesai, baru dikerjakan yang lainnya).

contoh : ada 5 terminal dengan 1 printer, maka harus mengantri untuk pemrosesan yang lain

Keunggulan:

Data tidak hilang karena sudah terprogram.

2. Masukan/keluaran Dituntun Interupsi.

Mekanisme kerjanya yaitu:

a. Pemroses memberi instruksi ke perangkat masukan/keluaran kemudian dilanjutkan dengan melakukan kerja berguna yang lain.
b. Perangkat masukan/keluaran akan menginterupsi meminta layanan saat perangkat telah siap bertukaran data dengan pemroses.
c. Saat menerima interupsi perangkat keras (yang memberitahukan bahwa perangkat siap melakukan transfer), pemroses segera mengeksekusi transfer data.

Keunggulan:

- pemroses tidak disibukan menunggui & menjaga perangkat masukan/keluaran untuk memeriksa status perangkat.
- kinerjanya lebih baik dibanding dengan teknik masukan/keluaran terprogram.

3. DMA (Direct Memory Access).

- Perangkat masukan/keluaran dikendalikan interupsi lebih efisien dibanding masukan/keluaran terprogram, tetapi masih memerlukan intervensi aktif pemroses untuk transfer data antara memori & buffer perangkat masukan/keluaran. Pemroses masih disibukan dengan operasi transfer data.

Pemrograman masukan/keluaran dikendalikan interupsi mempunyai 2 kelemahan, yaitu:

a. Rate Transfer masukan/keluaran dibatasi kecepatan menguji & melayani operasi perangkat.
b. Pemroses terikat ketat dalam mengelola transfer masukan/keluaran. sejumlah instruksi ini harus dieksekusi untuk tiap transfer masukan/keluaran.

DMA berfungsi membebaskan pemroses menunggui transfer data yang dilakukan perangkat masukan/keluaran. Saat pemroses ingin membaca/menulis data, pemroses memerintahkan DMA Controller dengan mengirim informasi berikut :

- Perintah penulisan/pembacaan.
- Alamat perangkat masukan/keluaran.
- Awal lokasi memori yang ditulis/dibaca.
- Jumlah word (byte) yang ditulis/dibaca.

Keunggulan:

1. penghematan waktu pemroses.
2. peningkatan kinerja masukan/keluaran.

C. EVOLUSI FUNGSI PERANGKAT MASUKAN/KELUARAN

Evolusi sangat tampak pada fungsi masukan/keluaran, sbb : [STA-95].

1. Pemroses Mengendalikan Perangkat masukan/keluaran secara langsung.
   Teknik ini masih dilakukan sampai saat ini untuk perangkat sederhana yang dikendalikan mikroprosesor sehingga menjadi perangkat berintelijen (intelegent device).
2. Perangkat dilengkapi Pengendali masukan/keluaran (I/O Contoller).
   Pemroses masih menggunakan masukan/keluaran terprogram tanpa interupsi.
   Pemroses tidak perlu memperhatikan rincian-rincian spesifik antarmuka perangkat.
3. Perangkat dilengkapi Fasilitas Interupsi.

- Pemroses tidak perlu menghabiskan waktu menunggu selesainya operasi masukan/keluaran.
- Teknik ini meningkatkan efisiensi pemroses.

4. I/O Controller mengendalikan memori secara langsung lewat DMA.
   - Pengendali dapat memindahkan blok data ke/dari memori tanpa melibatkan pemroses keculai di awal & akhir transfer.
5. Pengendali masukan/keluaran menjadi pemroses terpisah.
   - Pemroses pusat mengendalikan/memerintahkan pemroses khusus masukan/keluaran untuk mengeksekusi program masukan/keluaran di memori utama.
6. Pengendali masukan/keluaran mempunyai memori lokal.
   - Perangkat masukan/keluaran dapat dikendalikan dengan keterlibatan pemroses pusat yang minimum.
   - Arsitektur ini untuk pengendalian komunikasi dengan terminal-terminal interaktif, pemroses masukan/keluaran mengambil alih kebanyakan tugas yang melibatkan pengendalian terminal.

D. PRINSIP MANAJEMEN PERANGKAT M/K

Ada 2 sasaran perancangan manajemen perangkat masukan/keluaran, antara lain :

1. Efisiensi (eficiency).
   - Efisiensi merupakan aspek penting karena operasi masukan/keluaran sering merupakan operasi yang menimbulkan bottleneck [=hambatan].
2. Generalitas (generality).
   - Manajemen perangkat masukan/keluaran selain berkaitan dengan simplisitas & bebas kesalahan, juga menangani perangkat secara seragam baik dari cara proses memandang maupun cara system operasi mengelola perangkat dan operasi masukan/keluaran.
   - Disebut : “ Device Independence “.

Masalah yang timbul.

1. Penamaan yang Seragam (uniform naming).

- Nama berkas/perangkat berbentuk string atau integer.
- Tidak bergantung pada perangkat sama sekali.

2. Penanganan Kesalahan (error handling).
   - Umumnya ditangani sedekat mungkin dengan perangkat keras.
3. Transfer Sinkron vs Asinkron.
   - Kebanyakan I/O adalah sinkron.
   - Pemroses mulai transfer dan mengabaikan untuk melakukan kerja lain sampai interupsi tiba.
   - Program pemakai sangat lebih mudah ditulis jika operasi masukan/keluaran berorientasi blok setelah perintah read, program kemudian ditunda secara otomatis sampai data tersedia di buffer. Terserah system operasi untuk menangani operasi yang sesungguhnya interrupt-driven.
4. Sharable vs Dedicated.
   - Sharable = dapat digunakan pada saat yg bersamaan. [disket]
   - Dedicated = tidak bisa digunakan pada saat yg bersamaan/adanya antrian. [printer]

E. HIRARKI MANAJEMEN PERANGKAT MASUKAN/KELUARAN

1. Interrupt Handler.
   - Interupsi harus disembunyikan agar tidak terlihat rutin berikutnya.
   - Device driver di blocked saat perintah m/k diberikan & menunggu interupsi,
   - Ketika interupsi terjadi, prosedur penanganan interupsi bekerja agar device driver keluar dari state blocked.

2. Device Drivers.
   - Semua kode bergantung perangkat di tempatkan di device driver
   - Tiap device driver menangani satu tipe/satu kelas perangkat.
   - Device driver bertugas menerima permintaan abstrak perangkat lunak device-independent diatasnya & melakukan layanan sesuai permintaan.

Mekanisme kerja device driver:

1. menterjemahkan perintah abstrak menjadi perintah kongkret.
2. begitu telah dpt ditentukan perintah yang harus diberikan ke pengendali, device driver mulai menulis ke register-register pengendali perangkat.
3. setelah operasi selesai dilakukan perangkat, device driver memeriksa status kesalahan yang terjadi.
4. jika berjalan baik, device driver melewatkan data ke perangkat lunak device independent.
5. device driver melaporkan status operasinya ke pemanggil.

3. Perangkat Lunak Device-Independent.
    - Fungsi utamanya adalah, membentuk fungsi-fungsi masukan/keluaran yang berlaku untuk semua perangkat & memberi antarmuka yang seragam ke perangkat lunak tingkat pemakai.
    - Fungsi yang biasa dilakukan, antara lain:

1. interface seragam untuk seluruh device-driver.
2. penamaan perangkat.
3. proteksi perangkat.
4. memberi ukuran blok perangkat agar bersifat device-independent.
5. melakukan buffering.
6. alokasi penyimpanan pada blok device.
7. alokasi & pelepasan dedicated devices.
8. pelaporan kesalahan.

4. Perangkat Lunak Level Pemakai.
    - Kebanyakan perangkat lunak masukan/keluaran terdapat di system operasi. satu bagian kecil berisi pustaka-pustaka yang ditautkan pada program pemakai & berjalan di luar kernel.

- System call masukan/keluaran umumnya dibuat sebagai prosedur-prosedur pustaka.
- Kumpulan prosedur pustaka masukan/keluaran merupakan bagian sistem masukan/keluaran.
- Tidak semua perangkat lunak masukan/keluaran level pemakai berupa prosedur-prosedur pustaka kategori penting adalah : sistem spooling.
- Spooling adalah, cara khusus berurusan dengan perangkat masukan/keluaran yang harus didedikasikan(dedicated I/O devices) pada sistem multi programming.

BUFFERING MASUKAN/KELUARAN.

- Buffering adalah untuk melembutkan lonjakan-lonjakan kebutuhan pengaksesan masukan/keluaran.
- Tujuan buffering adalah meningkatkan efisiensi system operasi & kinerja proses.
- Terdapat beragam cara buffering, antara lain:

1. Single Buffering.
    - Teknik ini paling sederhana.
    - Ketika proses memberi perintah untuk perangkat masukan/keluaran, system operasi menyediakan buffer memori utama sistem untuk operasi.

    Untuk perangkat Berorientasi Blok.

    - Transfer masukan dibuat ke buffer sistem. Ketika trasfer selesai, proses memindahkan blok ke ruang pemakai & segera meminta blok lain.
    - Teknik ini disebut : " reading ahead " atau " anticipated input ".

    Keunggulan:

    1. meningkatkan kecepatan.
    2. proses pemakai dapat memroses blok data sementara blok berikutnya sedang dibaca.
    3. System operasi dapat men-swap keluar proses karena operasi masukan berada di memori sistem bukan memori proses pemakai.

Kelemahan :

1. teknik ini merumitkan system operasi, karena harus mencatat pemberikan buffer-buffer sistem ke proses pemakai.
2. logika swapping juga dipengaruhi.

Untuk perangkat Berorientasi Aliran Karakter.

a. Operasi Line-at-a- time.
   - Cocok untuk terminal mode gulung (scroll terminal atau dumb terminal).
   - Masukan pemakai adalah satu baris perwaktu dengan carriage return menandai akhir baris.
   - Keluaran terminal juga serupa yaitu satu baris perwaktu.
   - Pada line-at-a-time, buffer digunakan untuk menyimpan satu baris tunggal, proses pemakai ditunda selama masukan, menunggu kedatangan catu baris seluruhnya. Untuk keluaran, proses pemakai menempatkan satu baris keluaran pada buffer & melanjutkan pemrosesan. Proses tidak perlu suspend kecuali bila baris ke-2 dikirim sebelum buffer dikosongkan.
   - contoh : line printer.

b. Operasi Byte-at-a-time.

   Digunakan untuk terminal mode form dimana tiap ketikan adalah penting & untuk peripheral lain seperti sensor & pengendali.

   Pada byte-at-a-time, interaksi system operasi & proses pemakai mengikuti model producer-consumer.

2. Double Buffering.
   - Proses dapat transfer ke/dari satu buffer sementara system operasi mengosongkan/mengisi buffer lain.
   - Teknik ini disebut : " double buffering " atau " buffer swapping ".

Untuk Berorientasi Aliran Karakter.

a. Mode line-at-a-time.

   Proses pemakai tidak perlu ditunda untuk masukan/keluaran kecuali proses secepatnya mengosongkan buffer ganda.

b. Mode-byte-at-a-time.

   Buffer ganda tidak memberi keunggulan berarti atas buffer tunggal.

   Double buffering mengikuti mode producer-consumer.

3. Circular Buffering.
   - Double buffering tidak mencukupi jika proses melakukan operasi masukan/keluaran yang berurutan dengan cepat.
   - Masalah sering dapat dihindari dengan menggunakan lebih dari 2 buffer.
   - Ketika lebih dari 2 buffer yang digunakan, kumpulan buffer itu sendiri diacu sebagai circular buffer.

Tiap buffer individu adalag satu unit di cirucular buffer

Daftar Pustaka

13. Sri Kusuma Dewi, Sistem Operasi, Graha Ilmu, 2002.
14. William Stallings, Sistem Operasi Edisi Empat, PT. Indeks, Kelompok Gramedia, 2005.
15. Hariyanto, B.,"Sistem Operasi", Bandung: Informatika, Desember 1997
16. Silberschatz, A., Gagne, G. dan Galvin, P., "*Applied Operating System Concept*", John Wiley and Sons Inc., 2000

MODUL PERKULIAHAN

MANAJEMEN FILE DAN PENERAPANNYA

Pengelolaan file adalah kumpulan perangkat lunak sistem yang menyediakan layanan-layanan berhubungan dengan penggunaan file ke pemakai dan/atau aplikasi.

Abstract

Program dapat mengakses file di sistem melalui sistem manajemen basisdata (DBMS) ataupun secara langsung melalui fasilitas yang disediakan system operasi.

Kompetensi

Memahami penerapan sasaran dan fungsi sistem manajemen file.

Memahami fungsi manajemen file.

Manajemen FILE

A. Sasaran Dan Fungsi Sistem Manajemen File.

File mempunyai sifat sbb:

1. Persistence.

   Informasi dapat bertahan meski proses yang membangkitkannya berakhir atau meskipun catu daya dihilangkan.

2. Size.

   File umumnya berukuran besar. Memungkinkan menyimpan informasi yang sangat besar.

3. Sharability.

   File dapat digunakan banyak proses untuk mengakses informasi secara kongkuren.

Sasaran Manajemen File.

- Pengelolaan file adalah kumpulan perangkat lunak sistem yang menyediakan layanan-layanan berhubungan dengan penggunaan file ke pemakai dan/atau aplikasi.
- Biasanya, satu-satunya cara pemakai atau aplikasi mengakses file adalah lewat sistem file.
- Pemakai atau pemrogram tidak perlu mengembangkan perangkat lunak khusus untuk mengakses data di tiap aplikasi.

Grosshan [GRO-86] menyarankan sasaran sistem file sbb:

1. Memenuhi kebutuhan manajemen data bagi pemakai.

   Kebutuhan manajemen data bagi pemakai, yaitu kemampuan melakukan operasi-operasi berikut :

   Retrieve All : menampilkan seluruh record data.
   Retrieve One : menampilkan seluruh satu rekord data tertentu.

Retrieve Next : menampilkan satu record berikutnya.
Retrieve Previous : menampilkan satu record data sebelumnya.
Insert One : menyisipkan satu record data.
Delete One : menghapus satu rekord data.
Update One : memperbaharui satu rekord tertentu.
Update few : memperbaharui bbrp rekord data tertentu yang 1 kriteria.

2. Menjamin data pada file adalah valid.
3. Optimasi kinerja.
   - Menurut sistem yaitu : meningkatkan jumlah throughput keseluruhan.
   - Menurut pemakai yaitu : cepatnya waktu tanggap.
4. Menyediakan dukungan masukan/keluaran beragam tipe perangkat penyimpan.
5. Meminimalkan atau mengeliminasi potensi kehilangan atau perusakan data.
6. Menyediakan sekumpulan rutin interface masukan/keluaran.
7. Menyediakan dukungan masukan/keluaranbanyak pemakai di sistem multiuser.

Fungsi Manajemen File.

Beberapa fungsi yang diharapkan dari pengelolaan file adalah:

1. Penciptaan, modifikasi dan penghapusan file.
2. Mekanisme pemakaian file secara bersama.
   Mekanisme beragam tipe pengaksesan terkendali seperti:

   Read access : pengendalian terhadap akses membaca.
   Write access : pengendalian terhadap akses memodifikasi.
   Execute access : pengendalian terhadap akses menjalankan program.
3. Kemampuan backup dan recovery untuk mencegah kehilangan karena kecelakaan atau dari upaya penghancuran informasi.
4. Pemakai dapat mengacu file dengan nama simbolik (symbolic name) bukan menggunakan penamaan yang mengacu perangkat fisik.

5. Pada lingkungan sensitif dikehendaki informasi tersimpan aman dan rahasia.
   Lingkungan ini seperti :

   Electronic fund transfer system.
   Criminal record system.
   Medical record system.

   Sistem file menyediakan enkripsi dan dekripsi untuk menjaga informasi hanya digunakan oleh pemakai yang diotorisasi saja.

6. Sistem file harus menyediakan interface user-friendly.
   Pandangan secara logik (logical view) bukan pandangan secara fisik (physical view) terhadap data.
   Fungsi yang dapat dilakukan terhadap data.

   Pemakai tidak berkutat pada perangkat fisik dimana data disimpan, bentuk data harus diambil dari perangkat atau cara-cara fisik transfer data ke/dari perangkat-perangkat itu.

B. Arsitektur Pengelolaan File

Pengelolaan file terdiri dari :

1. Sistem akses.
   Berkaitan dengan bagaimana cara pengaksesan data yang disimpan pada file.

2. Manajemen file.
   Berkaitan dengan penyediaan mekanisme operasi pada file, seperti:

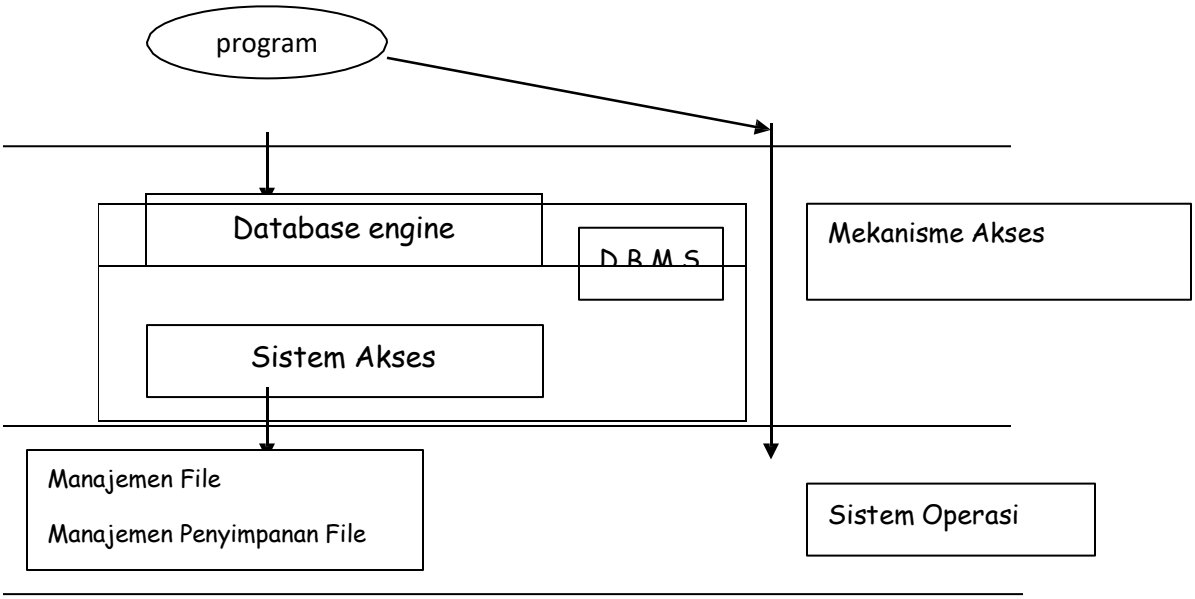
   Penyimpanan
   Pengacuan.
   Pemakaian bersama.
   Pengamanan.

3. Manajemen ruang penyimpan.
   Berkaitan dengan alokasi ruang untuk file di perangkat penyimpan.

4. Mekanisme integritas file.

Berkaitan denang jaminan informasi pada file tak terkorupsi.

Mekanisme program mengakses file di penyimpanan data sekunder dapat digambarkan sbb:



Gambar : Cakupan Sistem File Dapat Dilihat Dari Arsitekturnya.

Program dapat mengakses file di sistem melalui sistem manajemen basisdata (DBMS) ataupun secara langsung melalui fasilitas yang disediakan system operasi. Umumnya system operasi menyediakan :

Manajemen file.

Manajemen penyimpanan file.

Mekanisme integritas.

DBMS umumnya memuat bagian berikut :

Database engine, diantaranya berisi mekanisme integritas.

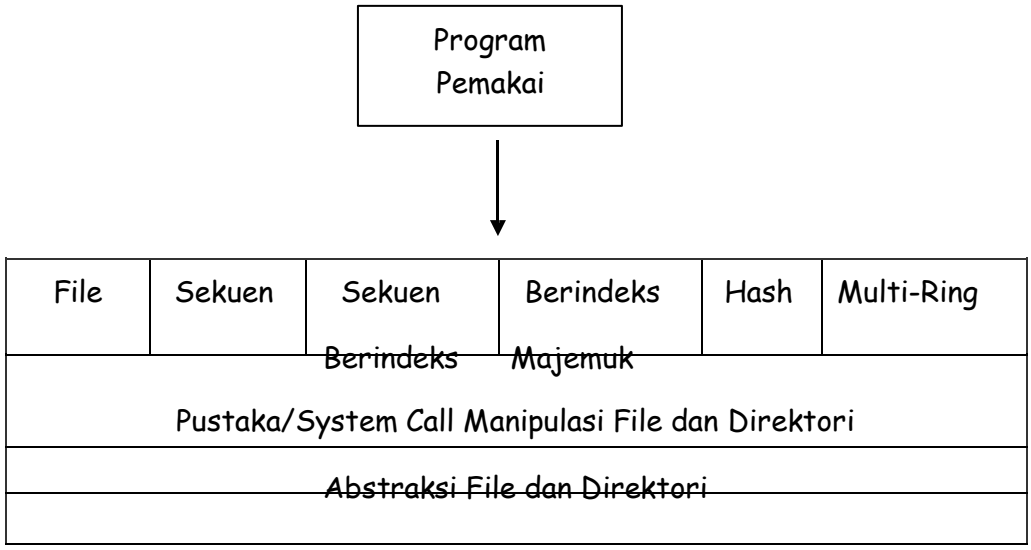
Sistem akses.

DBMS menggunakan fasilitas yang disediakan system operasi untuk memberikan layanan-layanannya. Mekanisme integritas merupakan masalah yang dilakukan baik di tingkat system operasi maupun di DBMS.

Hanya sistem operasi yang dikhususkan untuk basisdata yang secara langsung menyatukan sistem akses di system operasi agar diperoleh kinerja yang lebih bagus.

Kebanyakan system operasi hanya menyediakan fasilitas pengelolaan umum yg akan digunakan perangkat lunak aplikasi diatasnya.

Arsitektur sistem pengelolaan file adl sbb:



Gambar : Arsitektur Sistem File.

Pengelolaan File melibatkan banyak subsistem penting, yaitu :

1. Manajemen Perangkat masukan/keluaran di Sistem Operasi.

   Device Driver merupakan lapisan terbawah, berkomunikasi dengan perangkat secara langsung.

2. Sistem File di Sistem Operasi

   Sistem File Dasar.

   Sistem file dasar atau tingkat masukan/keluaran fisik merupakan interface utama dengan perangkat keras.

   Lapisan ini berurusan dengan blok-blok data yang dipertukarkan antara sistem dengan disk dan tape.

   Abstraksi File dan Direktori.

   Sistem file memberikan abstraksi ke pemakai berupa file dan direktori

   Pemakai yaitu :manusia ataupun proses tidak lagi berkaitan dengan blok-blok data melainkan beroperasi terhadap abstraksi file dan/atau direktori.

   Operasi-operasi terhadap File dan Direktori.

   Kumpulan system call dan/atau pustaka untuk manipulasi file dan direktori.

3. Sistem Akses dan/atau Sistem Manajemen Basis data.

   Metode akses merupakan lapisan terakhir.

   Lapisan ini menyediakan interface standar antara aplikasi-aplikasi dan sistem file serta perangkat yang menyimpan data.

   Metode-metode pengaksesan yang berbeda merefleksikan struktur file berbeda dan cara-cara pengaksesan dan pemrosesan yang berbeda.

   Metode-metode pengaksesan yang paling dikenal antara lain:

   File pile (pile file).

   File sekuen (sequential file).

File sekuen berindeks (multiple-sequential file).
File berindex majemuk (multiple-indexed fle).
File ber-hash (hashed file).
File multiring (multi ring file).

C. Sistem File.

Konsep dasar dari pengolahan file di system operasi adalah:

1. FILE

Abstraksi penyimpanan dan pengambilan sebagaian file atau direktori,
Ada beberapa pandangan mengenai file, yaitu :
Pandangan Pemakai.
Terhadap file pemakai berkepentingan memahami berikut :

a. Penamaan Untuk File.

Pemakaian mengacu file dengan nama simbolik.
Tiap file di sistem harus mempunyai nama unik agar tidak bingung.

b. Tipe File.

Ada 3 tipe file di system operasi, yaitu :

File regular.

- File berisi informasi, terdiri dari file ASCII dan biner.
- File ASCII berisi teks.
- File biner adalah file yang bukan file ASCII.
- Untuk file biner eksekusi (exe) mempunyai struktur internal yang hanya diketahui system operasi.
- Untuk file biner hasil program aplikasi, struktur internalnya hanya diketahui program aplikasi yang menggunakan file tersebut.

File Direktori.

- File direktori merupakan file yang dimiliki sistem untuk mengelola struktur sistem file.
- File direktori merupakan file berisi informasi-informasi mengenai file-file yang termasuk dalam direktori itu.

File Spesial.

- File spesial merupakan nama logik perangkat masukan/keluaran
- Perangkat masukan/keluaran dapat dipandang sebagai file.

File spesial terbagi menjadi 2,yaitu :

File spesial karakter.

File spesial karakter berhubungan dengan perangkat masukan/keluaran aliran karakter.

File ini memodelkan perangkat masukan/keluaran seperti : Terminal, Printer, Port jaringan, Modem, dan alat-alat yang bukan penyimpnan sekunder.

File Spesial Blok.

File spesial blok berhubungan dengan perangkat masukan/keluaran sebagai kumpulan blok-blok data (berorientasi blok).

c. Atribute File.

Informasi tambahan mengenai file untuk memperjelas dan membatasi operasi-operasi yang dapat diterapkan.

Atribut dipergunakan untuk pengelolaan file.

contoh atributnya : protection, password, creator, owner, read-only-flag, hidden-flag, dsb

d. Perintah-perintah Memanipulasi File.

Perintah-perintah memanipulasi file merupakan perintah yang dapat diberikan pemakai di baris perintah ke shell (command interpreter).

Perintah-perintah tersebut dapat dikategorikan menjadi:

- Perintah penciptaan file.
- Perintah penghapusan file.
- Perintah pengkopian.
- Perintah penggantian nama.

- Perintah memanipulasi yang lain.

Pandangan Pemrogram.

Selain perlu memahami sebagai pemakai, pemrogram perlu memahami :

Operasi-operasi terhadap file.

- Beragam operasi dpt diterapkan pada file.
- contoh operasi thd file : create, delete, open, close, read, write, append, seek, dsb

Pandangan Perancang.

Implementasi pengelolaan file.

2. DIREKTORI

Direktori sendiri adalah file, yang dimiliki oleh system operasi dan dapat diakses dengan rutin oleh system operasi.

Meski beberapa informasi direktori tersedia bagi pemakai atau aplikasi, informasi itu umumnya disediakan secara tidak langsung.

Menurut Pandangan Pemakai.

- Direktori menyediakan pemetaan nama file ke file.
- Informasi penting pada direktori adalah berkaitan dengan penyimpanan, termasuk lokasi dan ukuran penyimpanan file.
- Pada sistem bersama (shared system), informasi yang penting adalah informasi pengendali akses file.
- Aturan penamaan direktori mengikuti aturan penamaan file karena direktori merupakan file yang khusus.

Beberapa konsep penting yang dipahami oleh pemakai :

a. Hirarki Direktori.

Kebanyakan sistem menggunakan hirarki direktori atau berstruktur pohon..

Penamaan direktori sama aturannya dengan penamaan file karena direktori adalah file yang mempunyai arti khusus.

b. Nama Jalur Pengaksesan.

Bila sistem file diorganisasikan dengan pohon direktori, maka diperlukan cara menspesifikasikan nama file.

Masalah penamaan file diselesaikan dengan penamaan absolut dan penamaan file relatif.

Terdapat 2 jalur yaitu :

Nama jalur Absolut.

Nama jalur dari direktori root ke file, selalu dimulai dari direktori root dan akan bernilai unik.

Nama jalur Relatif.

- Jalur relatif terhadap direktori kerja/saat itu (working directory atau current directory).
- Pemakai dapat menyatakan satu directory sebagai current directory.
- Nama jalur yang tidak dimulai directory root berarti relatif terhadap current directory.

c. Perintah-perintah Memanipulasi Direktori.

Perintah-perintah meliputi :

Pindah direktori.

Penciptaan direktori.

Penghapusan direktori.

Penghapusan direktori mensyaratkan :

Direktori tidak sedang digunakan.

Direktori telah kosong.

d. Operasi Pada Direktori.

Create, delete, open-directory, close-directory, read-directory, rename, link, unlink.

3. MEMANIPULASI SELURUH SISTEM FILE

Macam-macam perintahnya yaitu :

- Pembentukan sistem file.
- Pemeriksaan sistem file.
- Pengkopian seluruh sistem file.
- Manipulasi lain.

D. Shared File

Yaitu file yang tidak hanya diacu oleh satu direktori (pemakai), tapi juga oleh direktori-direktori lain.

Sistem file tidak lagi berupa pohon melainkan directory acyclic graph (DAG).

Masalah-masalah yang terdapat pada shared file adalah sbb :

- Metode implementasi shared file.
- Metede pemberian hak akses pada shared file.
- Metode pengendalian atau penanganan terhadap pengaksesan yang secara simultan dilakukan pemakai-pemakai yang mengacu file. Persoalan pengaksesan simultan ini menyangkut integritas atau koherensi data.

E. Sistem Akses File

Sistem akses merupakan pilihan, yaitu :

Dapat menjadi bagian dari system operasi atau

Sistem operasi sama sekali tidak mempunyai komponen sistem akses.

Sistem Operasi bertujuan umum tidak mengimplementasikan sistem akses sebagai komponen system operasi, terserah sistem manajemen basisdata yg dijalankan di system operasi untuk menangani sistem akses. Sistem Operasi hanya memberikan pengelolaan sistem file dasar.

Sistem Operasi tertentu sering mengimplementasikan sistem akses sebagai bagiannya seperti system operasi mainframe untuk tujuan khusus. Implementasi sistem akses ditingkat system operasi untuk meningkatkan kinerja sistem manajemen basisdata.

1. Cara akses Perangkat Penyimpanan.

   Perangkat penyimpanan berdasar disiplin pengaksesan dibagi 2 , yaitu :

   a. Perangkat Akses Sekuen (sequential access devices).

      Proses harus membaca semua byte atau rekord file secara berurutan mulai dari awal, tidak dapat meloncati dan membaca di luar urutan.

      contoh : tape.

   b. Perangkat Akses Acak (random access devices).

      Dimungkinkan dapat membaca byte atau rekord file di luar urutan atau mengakses rekord berdasar kunci bukan posisinya.

2. Organisasi File.

   Elemen pokok perancangan sistem akses adalah cara rekord-rekord diorganisasikan atau distrukturkan.

   Beberapa kriteria untuk pemilihan organisasi file adalah : [WIE-87].

   - Redundansi yang kecil.
   - Pengaksesan yang cepat.
   - Kemudahan dalam memperbaharui.
   - Pemeliharaan yang sederhana.
   - Kehendalan yang tinggi.

Terdpt 6 organisasi dasar, (kebanyakan organisasi file sistem nyata termasuk salah satu atau kombinasi kategori-kategori ini).

6 kategori tsb adalah sbb :

- File pile (pile).
- File sekuen (sequential file).
- File sekuen berindeks (indexed-sequential file).
- File berindeks majemuk (multiple-indexed file).
- File ber-hash (hashed or direct file).
- File cincin (multi-ring file).

Daftar Pustaka

17. Sri Kusuma Dewi, Sistem Operasi, Graha Ilmu, 2002.
18. William Stallings, Sistem Operasi Edisi Empat, PT. Indeks, Kelompok Gramedia, 2005.
19. Hariyanto, B.,"Sistem Operasi", Bandung: Informatika, Desember 1997
20. Silberschatz, A., Gagne, G. dan Galvin, P., "*Applied Operating System Concept*", John Wiley and Sons Inc., 2000

MODUL PERKULIAHAN

KEAMANAN SISTEM

Keamanan sistem komputer untuk menjamin sumber daya tidak digunakan atau dimodifikasi orang yang tidak punya otorisasi. Pengamanan termasuk masalah teknis, manajerial, legalitas dan politis.

Abstract

Keamanan (security) dan proteksi (protection) sering di gunakan secara bergantian. Untuk menghindari kesalah pahaman, istilah keamanan mengacu ke seluruh masalah keamanan dan istilah mekanisme proteksi mengacu ke mekanisme sistem yang digunakan untuk memproteksi/ melindungi informasi pada sistem komputer

Kompetensi

Memahami konsep Autentifikasi, proteksi system.

Memahami virus dan anti virus.

KEAMANAN SISTEM

A. KEAMANAN

Keamanan sistem komputer adalah untuk menjamin sumber daya tidak digunakan atau dimodifikasi orang yang tidak punya otorisasi. Pengamanan termasuk masalah teknis, manajerial, legalitas dan politis.

Keamanan sistem terbagi menjadi 3 yaitu : [DEI-90].

1. Keamanan Eksternal (External Security).

   Berkaitan dengan pengamanan fasilitas komputer dari penyusup dan bencana seperti kebakaran atau kebanjiran.

2. Keamanan Interface Pemakai (User Interface Security).

   Berkaitan dengan identifikasi pemakai sebelum pemakai diijinkan mengakses program dan data yang disimpan.

3. Keamanan Internal (Internal Security).

   Berkaitan dengan pengamanan beragam kendali yang dibangun pada perangkat keras dan system operasi yang menjamin operasi yang handal dan tak terkorupsi untuk menjaga integritas program dan data.

Istilah keamanan (security) dan proteksi (protection) sering digunakan secara bergantian. Untuk menghindari kesalah pahaman, istilah keamanan mengacu ke seluruh masalah keamanan dan istilah mekanisme proteksi mengacu ke mekanisme sistem yang digunakan untuk memproteksi/melindungi informasi pada sistem komputer.

1**. Masalah-masalah Keamanan.**

Terdapat 2 masalah penting, yaitu : [TAN-95].

a. Kehilangan Data (data loss).
   Dapat disebabkan antara lain:

   - Bencana.
     Kebakaran, banjir, gempa bumi, perang, kerusuhan, gerogotan tikus pada pita rekaman data atau floppy disk.

   - Kesalahan Perangkat Keras dan Perangkat Lunak.
     Ketidakberfungsian pemroses, disk atau tape yang tak terbaca, kesalahan telekomunikasi, kesalahan program (bugs).

   - Kesalahan/kelalaian manusia.
     Kesalahan pemasukan data, memasang tape atau disk yang salah, eksekusi program yang salah, kehilangan disk atau tape.

Kehilangan data dapat diatasi dengan mengelola beberapa backup dan backup ditempatkan jauh dari data yang online.

b. Penyusup (intruder).
   Terdiri dari :

   1. Penyusup pasif, yaitu yang membaca data yang tak diotorisasi.
   2. Penyusup aktif, yaitu mengubah data yang tak diotorisasi.

2. Ancaman Keamanan.

Kebutuhan keamanan sistem komputer dikategorikan 3 aspek, yaitu: [STA-95]

a. Kerahasiaan (secrecy, diantaranya adalah privasi).

Kerahasiaan adalah, keterjaminan bahwa informasi di sistem komputer hanya dapat diakses oleh pihak-pihak yang diotorisasi dan modifikasi tetap menjaga konsistensi dan keutuhan data di sistem.

b. Integritas (integrity).

Integirtas adalah, keterjaminan bahwa sumber daya sistem komputer hanya dapat dimodifikasi oleh pihak-pihak yang diotorisasi.

c. Ketersediaan (availability).

Ketersediaan adalah, keterjaminan bahwa sumber daya sistem komputer tersedia bagi pihak-pihak yang diotorisasi saat diperlukan.

Tipe-tipe ancaman terhadap keamanan sistem komputer dapat dimodelkan dengan memandang fungsi sistem komputer sebagai penyedia informasi. Berdasarkan fungsi ini, ancaman terhadap sistem komputer dikategorikan menjadi 4 ancaman, yaitu :

1. Interupsi (interuption).
2. Intersepsi (interception).
3. Modifikasi (modification).
4. Fabrikasi (fabrication).

3. Petunjuk Pengamanan Sistem.

Saltzer dan Schrooder (1975) memberi petunjuk mengenai prinsip-prinsip pengamanan sistem komputer, yaitu :

a. Rancangan sistem seharusnya publik.
b. Dapat diterima.
c. Pemeriksaan otoritas saat itu.
d. Kewenangan serendah mungkin.
e. Mekanisme yang ekonomis.

B. OTENTIFIKASI PEMAKAI

Masalah pemakai ketika login disebur otentifikasi pemakai (user authentication).

Kebanyakan metode ini di dasarkan pada 3 cara yaitu :

1. Sesuatu yang diketahui pemakai,misalnya :
   - Password, kombinasi kunci, dsb
2. Sesuatu yang dimiliki pemakai, misalnya :
   - Badge, kartu identitas, kunci, dst
3. Sesuatu mengenai (merupakan ciri) pemakai, misalnya :
   - Sidik jari, sidik suara, foto, tanda tangan, dll

1. **Password.**

Pemakai memilih satu kata kode, mengingatnya dan mengetikkan saat akan mengakses sistem komputer. Saat diketikakan, komputer tdk menampilkan di layar.

Teknik ini mempunyai kelemahan yang sangat banyak dan mudah ditembus.

Percobaan Morris dan Thompson menyatakan proteksi password dapat ditembus dengan mudah. Percobaan yang dilakukan adalah :

- Terdapat file berisi nama depan, nama belakang, nama jalan, nama kota dari kamus ukuran sedang, disertai dengan pengejaan dibalik, nomor plat mobil yang valid, dan string-string pendek karakter acak.
- Isian di file dicocokan dengan file password.

Upaya untuk lebih mengamankan proteksi password, antara lain :

Salting.

Menambah string pendek ke *string password* yang diberikan pemakai sehingga mencapai panjang password tertentu.

One-time Password.

Pemakai harus mengganti password secara teratur. Dengan sistem ini pemakai direpotkan keharusan menjaga agar buku *password*-nya jangan sampai dicuri.

Satu Daftar Panjang Pertanyaan dan Jawaban.

Variasi terhadap password adalah mengharuskan pemakai memberi satu daftar pertanyaan panjang dan jawabannya. Pertanyaan-pertanyaan dan jawabannya dipilih pemakai sehingga pemakai mudah mengingatnya dan tak perlu menuliskan di kertas.

- Tantangan-Tanggapan (Chalenge-Response).
  Pemakai diberi kebebasan memilih suatu alogaritma.

2. Identifikasi Fisik.

Pendekatan lain adalah memeriksa yang dimiliki pemakai.

a. Kartu Berpita Magnetik.
b. Sidik Fisik.
c. Analisis Tanda Tangan.
d. Analisis Suatu yang Dipunyai Pemakai.
e. Analis Darah.

3. Pembatasan.

Pembatasan-pembatasan dapat dilakukan sehingga memperkecil peluang penembusan oleh pemakai yang tak diotorisasi. Misalnya :

a. Pembatasan Login.

Login hanya dibolehkan:

Pada terminal tertentu.

Hanya pada waktu dan hari tertentu. b.

Pembatasan dengan Call-back.

Login dapat dilakukan siapapun.

Bila telah sukses login, sistem segera memutuskan koneksi dan memanggil nomor telepon yang telah disepakati.

Penyusup tidak dapat menghubungi lewat sembarangan saluran telepon, tapi hanya pada saluran telepon tertentu.

c. Pembatasan jumlah usaha login.

Login dibatasi sampai 3 kali dan segera dikunci dan diberitahu ke administrator.

Semua login direkam dan system operasi melaporkan informasi-informasi berikut :

- Waktu, yaitu waktu pemakai login.
- Terminal, yaitu terminal dimana pemakai login.

C. MEKANISME PROTEKSI SISTEM KOMPUTER

Pada sistem komputer banyak objek yang perlu diproteksi, yaitu :

a. Objek perangkat keras.

Yang perlu diproteksi yaitu : pemroses, segment memori, terminal, disk drive, printer, dll.

b. Objek perangkat lunak.

Yang perlu diproteksi yaitu : proses, file, basisdata, semaphore, dsb.

1. **Matriks Pengaksesan Objek.**

Masalah proteksi adalah mengenai cara mencegah proses-proses mengakses objek-objek yang tidak diotorisasi.

Mekanisme ini juga harus memungkinkan membatasi proses-proses ke suatu subse operasi-operasi legal yang diperlukan.

Agar dapat menyediakan mekanisme proteksi berbeda dikembangkan berdasar konsep domain. Domain adalah himpunan pasangan (objek, hak). Tiap pasangan menspesifikasikan objek dan suatu subset operasi yang dapat dilakukan terhadapnya. Hak dalam konteks ini berarti ijin melakukan suatu operasi.

Proses berjalan pada suatu domain proteksi, yaitu proses merupakan anggota suatu domain atau beberapa *domain.* Terdapat kumpulan objek yang dapat diakses proses. Untuk tiap objek, proses mempunyai suatu kumpulan hak terhadap objek itu. Proses-proses dapat juga beralih dari satu *domain* ke *domain* lain selama eksekusi. Aturan peralihan domain ini bergantung pada sistem.

Domain ditetapkan dengan mendaftarkan pemakai-pemakai yang termasuk *domain* itu. Proses-proses yang dijalankan pemakai adalah proses-proses pada *domain* itu dan mempunyai hak akses terhadap objek seperti ditentukan *domain*-nya.

Cara Penyimpanan Informasi Anggota Domain

Secara konseptual adalah berupa satu matriks besar, dimana :

- Baris menunjukkan domain.
- Kolom menunjukkan objek.

Tiap elemen matriks mendaftarkan hak-hak yang dimiliki domain terhadap objek. Dengan matriks ini, sistem dapat mengetahui hak pengaksesan terhadap objek.

Gambar ini menunjukkan matriks pengaksesan objek.

	File-1	File-2	Printer-1	Plotter-1	Modem1
Domain 1	Read	Read Write	Write		
2	Read			Write	Write
3		Read Write Execute	Write	Write	Write

Gambar matriks pengaksesan objek

Untuk sistem-sistem yang mengijinkan peralihan domain dimodelkan dengan menganggap domain sebagai objek, yaitu :

* jika terdapat operasi ENTER berarti mempunyai hak berpindah domain.

	File1	File2	Printer1	Plotter1	Modem1	Domain1	Domain2	Domain3
Domain1	Read	Read	Write				Enter	

		Write						
2	Read			Write	Write	Enter		
3		Read Write execute	Write	Write	Write			

Gambar matriks pengaksesan objek dengan operasi peralihan domain

Gambar ini menunjukkan bahwa proses-proses pada domain1 dapat berpindah ke domain2 dan proses pada domain 2 dapat berpindah ke domain1.

2. ACL (Access Control List).

Matriks pengaksesan objek akan berbentuk matriks jarang (sparse matrix). Matriks jarang memboroskan ruang penyimpanan dan lambat karena memerlukan ruang besar.

Alternatif untuk memperbaikinya adalah :

Menyimpan matriks sebagai per baris.
Menyimpan matriks sebagai per kolom.

Penyimpanan dilakukan hanya untuk isian yang tak kosong.

Teknik Penyimpanan Perkolom.

Teknik yang digunakan adalah mengasosiasikan setiap objek dengan senarai terurut berisi semua domain yang boleh mengakses dan operasi-operasi yang dibolehkan.

Teknik ini menghasilkan senarai disebut ACL (Access Control List).

3. Kapabilitas.

Cara lain adalah memecah matriks per baris. Diasosiasikan setiap proses berupa satu daftar objek yang boleh diakses, bila terdapat tanda operasi yang diijinkan padanya atau domainnya. Senarai ini disebut senarai kapabilitas (capability list)

D. PROGRAM-PROGRAM JAHAT

Ancaman-ancaman canggih terhadap sistem komputer adalah program yang mengeksploitasi kelemahan sistem komputasi. Kita berurusan dengan program aplikasi begitu juga program utilitas seperti editor dan kompilator.

Bowles [BOW-92] memberikan taksonomi ancaman perangkat lunak atau klasifikasi program jahat (malicious program).

Ancaman-ancaman itu dapat dibagi menjadi 2 kategori, yaitu :

1. Program-program yang Memerlukan Program Inang (Host Program).

   Fragmen program tidak dapat mandiri secara independen dari suatu program aplikasi, program utilitas atau program sistem.

2. Program-program yang Tidak memrlukan Program Inang.

   Program sendiri yang dapat dijadwalkan dan dijalankan oleh system operasi.

Pembagian taksonomi Bowles menghasilkan tipe-tipe program jahat sbb:

1. Bacteria.
2. Logic bomb.
3. Trapdoor.
4. Trojan horse.
5. Virus.
6. Worm.

1. Bacteria.

Bacteria adalah program yang mengkonsumsi sumber daya sistem dengan mereplikasikan diriya sendiri.

Bacteria tidak secara eksplisit merusak file.

Tujuan program ini hanya satu yaitu mereplikasi dirinya.

Program bacteria yg sederhana bisa hanya mengeksekusi 2 kopian dirinya secara simultan pada sistem multiprogramming atau menciptakan 2 file baru, masing-masing adalah kopian file program bacteria. Ke-2 kopian ini kemudian mengkopi 2 kali dan seterusnya.

Bacteria bereproduksi secara eksponensial, dengan cepat mengambil alih seluruh kapasitas pemroses, memori atau ruang disk, mengakibatkan penolakan pengaksesan pemakai ke sumber daya.

2. Logic Bomb.

Logic bomb adalah logik yang ditempelkan pada program komputer agar memeriksa suatu kumpulan kondisi di sistem. Ketika kondisi-kondisi yang dimaksud ditemui, logik mengeksekusi suatu fungsi yang menghasilkan aksi-aksi tak diotorisasi.

Logic bomb menempel pada suatu program resmi yang di-set ‘meledak’ ketika kondisi-kondisi tertentu dipenuhi.

Contoh kondisi-konisi untuk memicu logic bomb adalah ada atau tidak adanya file-file tertentu, hari tertentu dari minggu atau tanggal, atau pemakai menjalankan aplikasi tertentu. Begitu terpicu, bomb mengubah atau menghapus data seluruh file, menyebabkan mesin berhenti atau mengerjakan perusakan lain.

3. Trapdoor.

Trapdoor adalah titik masuk tak terdokumentasi rahasia di satu program untuk memberikan akses tanpa metode-metode otentifikasi normal.

Trapdoor telah dipakai secara benar selama bertahun-tahun oleh pemrogram untuk mencari kesalahan program.

Debugging atau testing biasanya dilakukan pemrogram saat mengembangkan aplikasi.

Untuk program yang mempunyai prosedur otentifikasi atau setup lama atau memerlukan pemakai memasukkan nilai-nilai berbeda untuk menjalankan aplikasi maka debugging akan lama bila harus melewati prosedur-prosedur tsb.

Untuk debug program jenis ini, pengembang membuat kewenangan khusus untuk menghilangkan keperlukan setup dan otentifikasi.

Trapdoor adalah kokd yang menerima suatu barisan masukan khusus atau dipicu dengan menjalankan ID pemakai tertentu atau barisan kejadian tertentu.

Trapdoor menjadi ancaman ketika digunakan pemrogram jahat untuk memperoleh pengaksesan tak diotorisasi.

Adalah sulit mengimplementasikan kendali-kendali perangkat lunak untuk trapdoor.

4. Trojan Horse.

Trojan horse adalah rutin tak terdokumentasi rahasia ditempelkan dalam satu program berguna.

Program yang berguna mengandung kode tersembunyi yang ketika dijalankan melakukan suatu fungsi yang tidak diinginkan. Eksekusi pogram menyebabkan eksekusi rutin rahasia ini.

Progam-program trojan horse digunakan untuk melakukan fungsi-fungsi secara tidak langsung dimana pemakai tak diotorisasi tidak dapat melakukannya secara langsung.

Program trojan horse yang sulit dideteksi adalah kompilator yang dimodifikasi sehingga menyisipkan kode tambahan ke program-program tertentu begitu dikompilasi, seperti program login.

Kode menciptakan trapdoor pada program login yang mengijinkan pencipta login ke sistem menggunakan password khusus. Trojan horse jenis ini tak pernah dapat ditemukan jika hanya membaca program sumber.

Motivasi lain dari trojan horse adalah penghancuran data

Program muncul sebagai fungsi-fungsi berguna (seperti kalkulator) tapi juga secara diam-diam menhapus file-file pemkai. Trojan horse biasa ditempelkan pada program-program atau rutin-rutin yang diambil dari BBS, internet, dsb

5. Virus.

Virus adalah kode yang ditempelkan dalam satu program yang menyebabkan pengkopian dirinya disisipkan ke satu atau lebih program lain.

Program ‘menginfeksi’ program-program lain dengan memodifikasi program-program lain itu. Modifikasi itu termasuk memasukkan kopian program virus yang kemudian dapat

‘menginfeksi’ program-program lain. Selain hanya propagasi, virus biasanya melakukan fungsi yang tak diinginkan.

Seperti virus biologis, pada virus komputer terdapat kode insruksi yang dapat membuat kopian sempurna dirinya. Ketika komputer yang terinfeksi berhubungan (kontak) dengan perangkat lunak yang belum terinfeksi, kopian virus memasuki program baru.

Infeksi dapat menyebar dari komputer ke komputer melalui pemakai-pemakai yang menukarkan disk atau mengirim program melalui jaringan. Pada lingkungan jaringan, kemampuan mengakses aplikasi dan layanan-layanan komputer lain merupakan fasilitas sempurna penyebaran virus.

6. Worm.

Worm adalah program yang dapat mereplikasi dirinya dan mengirim kopian-kopian dari komputer ke komputer lewat hubungan jaringan. Begitu tiba, worm diaktifkan untuk mereplikasi dan propagasi kembali. Selain hanya propagasi, worm biasanya melakukan fungsi yang tak diinginkan.

Network menggunakan hubungan jaringan untuk menyebar dari sistem ke sistem lain.

Sekali aktif di suatu sistem, network worm dapat berlaku seperti virus atau bacteria atau menempelkan program trojan horse atau melakukan sejumlah aksi menjengkelkan atau menghancurkan.

Untuk mereplikasi dirinya, network worm menggunakan suatu layanan jaringan seperti :

- Fasilitas surat elektronik (electronic mail facility), yaitu worm mengirimkan kopian dirinya ke sistem-sistem lain.
- Kemampuan eksekusi jarak jauh (remote execution capability), yaitu worm mengeksekusi kopian dirinya di sistem lain.

  Kemampuan login jarak jauh (remote login capability), yaitu worm log pada sistem jauh sebagai pemakai dan kemudian menggunakan perintah untuk mengkopi dirinya sari satu sistem ke sistem lain.

Kopian program yang baru kemudian dijalankan di sistem jauh dan melakukan fungsi-fungsi lain yang dilakukan di sistem itu. Worm terus menyebar dengan cara yang sama.

Network worm mepunyai ciri-ciri yang sama dengan virus komputer, yaitu mempunyai fase-fase sama, yaitu :

Dormant phase.

Propagation phase.

- Trigerring phase.
- Execution phase.

E. VIRUS DAN ANTI-VIRUS

1. Siklus Hidup Virus.

Virus sama dengan program komputer lain. Virus dapat mencantolkan dirinya ke program lain dan mengeksekusi kodenya secara rahasia setiap kali program inang berjalan. Masalah yang ditimbulkan virus antara lain merusak sistem komputer seperti menghapus file, partisi disk atau mengacaukan program.

Virus mengalami siklus hidup 2 fase (tahap), yaitu :

a. Fase Tidur (Dormant Phase).

Virus dalam keadaan menganggur.

Virus akan tiba-tiba aktif oleh suatu kejadian seperti tibanya tanggal tertentu, kehadiran program atau file tertentu, atau kapasitas disk yang melewati batas.

Tidak semua virus mempunyai tahap ini.

b. Fase Propagasi (Propagation Phase).

Virus menempatkan kopian dirinya ke program lain atau daerah sistem tertentu di disk.

Program yang terinfeksi virus akan mempunyai kloning virus. Kloning virus itu dapat kembali memasuki fase propagasi.

c. Fase Pemicuan (Triggering Phase).

Virus diaktifkan untuk melakukan fungsi tertentu.

Seperti pada fase tidur, fase pemicuan dapat disebabkan beragam kejadian sistem termasuk penghitungan jumlah kopian dirinya.

d. Fase Eksekusi (Execution Phase).

Virus menjalankan fungsinya. Fungsinya mungkin sepele seperti sekedar menampilkan pesan di layar atau merusak seperti merusak program dan file-file data, dsb..

2. Infeksi Virus.

a. Sekali virus telah memasuki sistem dengan menginfeksi satu program, virus berada dalam posisi menginfeksi beberapa atau semua file exe lain di sistem itu saat program yang terinfeksi dieksekusi.
b. Infeksi virus dapat sepenuhnya dihindari dengan mencegah virus masuk sisem. Pencegahan ini sangat luar biasa sulit karena virus dapat menjadi bagian program di luar sistem.
c. Kebanyak virus mengawali infeksinya pengkopian disk yang telah terinfeksi virus. Banyak disk berisi game atau utilitas di rumah dikopikan ke mesin kantor. Disk berisi virus pun dapat berupa disk yang dikirim produsen aplikasi. Hanya sejumlah kecil infeksi virus yang dimulai dari hubungan jaringan.

3. Tipe-tipe Virus.

Klasifikasi tipe virus adalah sbb : [STE-93]

Parasite Virus.

Merupakan virus tradisional dan bentuk virus yang paling sering. Tipe ini mencantolkan dirinya ke file exe. Virus mereplikasi ketika program yang terinfeksi dieksekusi dengan mencari file-file exe lain untuk diinfeksi.

Memory-Resident Virus.

Virus memuatkan diri ke meori utama sebagai bagian program yang menetap. Virus menginfeksi setiap program yang dieksekusi.

Boot Sector Virus.

Virus menginfeksi master boot record atau boot record dan menyebar saat sistem di boot dari disk yang berisi virus.

Stealth Virus.

Virus yg bentuknya telah dirancang agar dpt menyembunyikan diri dari deteksi perangkat lunak antivirus.

Polymorphic Virus.

Virus bermutasi setiap melakukan infeksi. Deteksi dengan ‘penandaan’ virus tersebut tidak dimungkinkan.

4. **Anti Virus.**

Beberapa pendekatan untuk pencegahan masuknya virus yaitu :

a. Deteksi.
b. Identifikasi.
c. Penghilangan.

Jika deteksi sukses dilakukan, tapi identifikasi atau penghilangan tidak dapat dilakukan, maka alternatif yang dilakukan adalah hapus program yang terinfeksi dan kopi kembali backup program yang masih bersih.

Perkembangan anti-virus terbagi menjadi 4 generasi, yaitu :

1. Generasi pertama : sekedar scanner sederhana.
2. Generasi kedua : scanner yang pintar (heuristic scanner).
3. Generasi ketiga : jebakan-jebakan aktivitas (activity trap).
4. Generasi keempat : proteksi penuh (full-featured protection).

Daftar Pustaka

21. Sri Kusuma Dewi, Sistem Operasi, Graha Ilmu, 2002.
22. William Stallings, Sistem Operasi Edisi Empat, PT. Indeks, Kelompok Gramedia, 2005.
23. Hariyanto, B.,"Sistem Operasi", Bandung: Informatika, Desember 1997
24. Silberschatz, A., Gagne, G. dan Galvin, P., "*Applied Operating System Concept*", John Wiley and Sons Inc., 2000