PERTEMUAN 4:

PROSES

A. TUJUAN PEMBELAJARAN

Pada bab ini akan dijelaskan mengenai proses, Anda harus mampu:

1.1 Status Proses

1.2 Distributed Processing

1.3 PCB

B. URAIAN MATERI

Tujuan Pembelajaran 1.1:
Proses

Proses adalah sebuah program yang sedang dieksekusi atau program yang sedang di jalankan atau software yang sedang dilaksanakan termasuk sistem operasi yang disusun menjadi sejumlah proses sequential. Sedangkan program adalah kumpulan instruksi yang ditulis ke dalam bahasa yang dimengerti sistem operasi. Proses berisi instruksi dan data. program counter dan semua register pemroses, dan stack berisi data sementara seperti parameter rutin, alamat pengiriman dan variabel-variabel lokal.

Sistem operasi mengelola semua proses di sistem dan mengalokasikan sumber daya ke proses-proses sesuai kebijaksanaan untuk memenuhi sasaran sistem. Salah satunya adalah program yang sedang dieksekusi yang merupakan unit kerja terkecil yang secara individu memiliki sumber daya-sumber daya dan dijadwalkan sistem operasi. Sistem operasi mengelola semua proses di sistem dan mengalokasikan sumber daya ke proses-proses sesuai kebijaksanaan untuk memenuhi sasaran sistem.

MULTIPROGAMMING (MULTITASKING)

Multiprogamming adalah manajemen banyak proses pada satu pemroses. Istilah yang digunakai multiprogrammaing (multitasking) bukan multiprocessing. Multprocessing telah digunakan untuk konsep lain, yaitu komputer dengan banyak pemroses di satu sistem komputer dengan masing-masing pemroses melakukan pemrosesan secara independen. Saat ini, kebanyakan komputer pribadi, workstation adalah sistem pemroses tunggal yang menjalankan sistem operasi multiprogramming (multitasking) seperti MS-Windows 3.0, MS-Windows NT, OS/2 dan Macintosh System 7. Banyak proses dijalankan bersamaan, masing-

masing proses mendapat bagian memori dan kendali tersendiri. Sisem operasi mengalih-alihkan pemroses di antara proses-proses tersebut.

MULTIPROCESSING

Multiprocessing adalah manajemen banyak proses di komputer multiprocessor (banyak pemroses di dalamnya). Dulunya sistem ini lanya terdapat di sistem besar, mainframe dan minikomputer. Saat ini komputer workstation telah dapat dilengkapi multiprocessor. Menggunakan komputer semaksimal mungkin dengan beberapa CPU sehingga beberapa program bisa dijalankan secara bersama-sama , masing-masing dengan menggunakan prosesornya sendiri-sendiri. Sistem operasi Microsoft Windows NT, UNIX, Linux menyediakan dukungan multiprocessing.

DISTRIBUTED PROCESSING / COMPUTING

Distributed Processing adalah manajemen banyak proses yang dieksekusi di banyak sistem komputer yang tersebar (terdistribusi). Trend masa datang adalah menuju komputasi tersebar (distributed computing). Banyak riset dan pengembangan sistem operasi tersebar di antaranya AMOEBA, MACH, dan sebagainya.

KEBUTUHAN UTAMA PENGENDALIAN PROSES OLEH SO

Pengendalian proses oleh SO yang mengacu pada Proses:

1. Saling melanjutkan (Interleave).
2. Mengikuti kebijaksanaan tertentu..
3. Mendukung komunikasi antar proses dan penciptaan proses.

STATUS PROSES

Status proses atau bagian keadaan proses memiliki tiga elemen. yaitu:

RUNNING

Running / kerja, benar-benar menggunakan CPU pada saat itu (sedang mengeksekusi instruksi proses itu).

Ada tiga kemungkinan bila sebuah proses memiliki status Running:

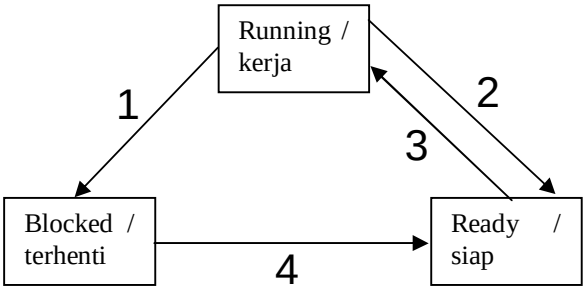
1. Jika program telah selesai dieksekusi maka status dari proses tersebut akan berubah menjadi Terminated.
2. Jika waktu yang disediakan oleh OS untuk proses tersebut sudah habis maka akan terjadi interrupt dan proses tersebut kini berstatus Ready.
3. Jika suatu event terjadi pada saat proses dieksekusi (seperti ada permintaan M / K) maka proses tersebut akan menunggu event tersebut selesai dan proses berstatus Waiting.

BLOCKED

Blocked / terhenti, tidak dapat berjalan sampai kegiatan eksternal terlaksana (proses menunggu kejadian untuk melengkapi tugasnya) Bisa berupa proses menunggu : Selesainya operasi perangkat I/O; Tersedianya memori; Tibanya pesan jawaban

READY

Ready / siap, proses siap dikerjakan tetapi menunggu giliran dengan proses lain yang sedang dikerjakan (bisa berjalan, sementara berhenti untuk memungkinkan proses lain dikerjakan).



Gambar II. 1 Diagram State 3 Keadaan

Keterangan:

1. Proses di blok untuk melayani input karena sumber daya yang diminta belum tersedia / meminta layanan I/O sehingga menunggu kejadian muncul.
2. Penjadwalan mengambil proses lain.
3. Penjadwalan mengambil proses ini (baru).
4. Input telah tersedia.

STATUS	DESKRIPSI
Running	Pemroses sedang mengeksekusi instruksi prose situ.
Ready	Proses siap (ready) dieksekusi, tapi tidak tersedia untuk eksekusi proses ini.
Blocked	Proses menunggu kejadian untuk melengkapi tugasnya. Contoh Proses Menunggu : • Selesainya operasi perangkat I / O • Tersedianya memori • Tibanya pesan jawaban • Dsb.

Gambar II. 2 Tabel Status Proses

STATUS TAMBAHAN

Terdapat dua, yaitu saat pembentukan dan terminasi:

1. New adalah status yang dimiliki pada saat proses baru saja dibuat.
2. Terminated adalah status yang dimiliki pada saat proses telah selesai dieksekusi.

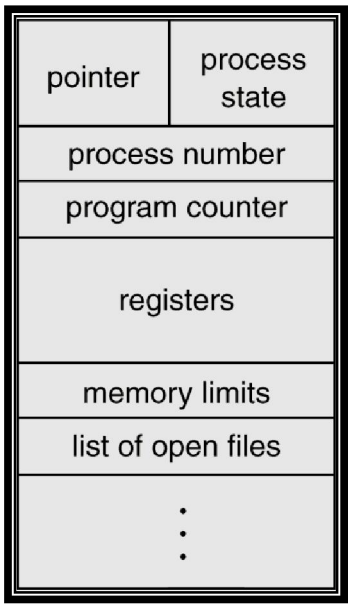
SIKLUS HIDUP PROSES

Layaknya makhluk hidup, proses juga memiliki siklus hidup dengan nama siklus hidup proses. Siklus ini memiliki 5 proses. Yaitu:

1. Proses yang baru diciptakan akan segera mempunyai state ready.
2. Proses berstate running menjadi blocked karena sumber daya yang diminta belum tersedia atau meminta layanan perangkat masukan/keluaran sehingga menunggu kejadian muncul. Proses menunggu kejadian alokasi sumber daya atau selesainya layanan perangkat masukan/keluaran (event wait).
3. Proses berstate running menjadi ready karena penjadwal memutuskan eksekusi proses lain karena jatah waktu untuk proses tersebut telah habis (time-out).
4. Proses berstate blocked menjadi ready saat sumber daya yang diminta/diperlukan telah tersedia atau layanan perangkat masukan/keluaran selesai (event occurs).
5. Proses berstate ready menjadi running karena penjadwal memutuskan penggunaan pemroses untuk proses itu karena proses yang saat itu running berubah statenya (menjadi ready atau blocked) atau telah menyelesaikan sehingga disingkirkan dari sistem Proses menjadi mendapatkan jatah pemroses.

PCB (Program Control Block)

PCB berisikan banyak bagian dari informasi yang berhubungan dengan sebuah proses yang spesifik, yaitu:



Gambar IV. 1 Diagram PCB

INFORMASI PCB

Sistem operasi memerlukan banyak informasi mengenai proses guna pengelolaan proses, Informasi ini berada di PCB. Didalam PCB terdapat 3 informasi, yaitu :

1. Informasi Identifikasi Proses

2. Informasi Status Pemroses

3. Informasi Kendsali Proses

Identifikasi Process
Identifier
Identifier numerik meliputi ▪ Identifier proses ▪ Identifier proses yang menciptakan ▪ Identifier pemakai
Informasi Status Pemroses
Register-register yang terlihat pemakai
Register-register yang dapat ditunjuk instruksi bahasa assembly untuk diproses pemroses
Register-register kendali dan status
Register-register yang digunakan untuk mengendalikan operasi proses, anatara lain : ▪ Program counter ▪ PSW ▪ Dsb
Pointer Stack
Tiap proses mempunyai satu stack atau lebih. Stack digunakan untuk parameter atau alamat prosedur pemangil dan system call. Ponter stack menunjuk posisi paling atas dari stack.

Informasi Kendali Proses
Informasi penjadwakan dan status
Informasi-informasi yang digunakan untuk menjalankan fungsi penjadwalan antara lain : ▪Status proses : mendefinisikan keadaan/status proses (running, ready, blocked, dsb) ▪Prioritas : menjelaskan prioritas proses ▪Informasi berkaitan dengan penjadwalan : berkaitan dengan informasi penjadwalan seperti lama menunggu, lama proses terakhir dieksekuisi, dsb. ▪Kejadian : identitas kejadian yang ditunggu proses.
Penstrukturan data
Satu proses dapat dikaitkan dengan proses lain dalam satu antrian atau ring, atau struktur lainnya. PCB harus memiliki pointer untuk mendukung struktur ini.
Komunikasi antar proses
Beragam flag, sinyal dan pesan dapat diasosiasikan dengan komunikasi antara dua proses yang terpisah. Informasi ini disimpan dalam PCB

ELEMEN – ELEMEN PCB

1. Kewenangan Proses

    - Proses dapat mempunyai kewenangan berkaitan dengan memori dan tipe instruksi yang dapat dijalankan

2. Manajemen Memori

    - Bagian ini berisi pointer ke tabel segmen atau page yang menyatakan memori maya (virtual memori) proses

3. Kepemilikan dan utilitasi sumber daya

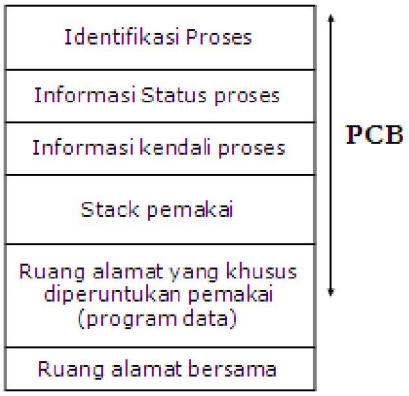
    - Sumber daya yang dikendalikan proses harus diberi tanda, misalnya :

- Berkas yang dibuka
- Pemakaian pemroses
- Pemakaian sumber daya lainnya

Informasi ini diperlukan oleh penjadwal.

PROSES PEMAKAI

Proses Pemakai mempunyai struktur berisi PCB, stack pemakai (user stack), ruang alamat proses ekslusif dan ruang alamat yang dipakai bersama proses lain. Struktur ini ditunjukkan pada tabel sebelumnya. Pada tabel diatas, struktur citra proses digambarkan kontinyuu (berturutan) di satu ruang alamat. Implementasi penempatan citra proses yang sesungguhnya bergantung skema manajemen memori yang digunakan dan organisasi struktur kendali sistem operasi.



Gambar V. 1 Diagram Proses Pemakai

OPERASI – OPERASI PADA PROSES

Penciptaan Proses (create a process)

Penciptaan proses melibatkan banyak aktivitas, yaitu:

1. Menamai (memberi identitas) proses.
2. Menyisipkan proses pada senarai proses atau tabel proses.
3. Menentukan prioritas awal proses.
4. Menciptakan PCB.
5. Mengalokasikan sumber daya awal bagi proses.

Penyebab penciptaan proses

1. Pada lingkungan batch, sebagai tanggapan atas pemberian satu kerja (job).
2. Pada lingkungan interaktif, ketika pemakai baru berusaha log on.
3. Sebagai tanggapan suatu aplikasi, seperti permintaan pencetakan file, sistem operasi dapat menciptakan proses yang akan mengelola pencetakan itu.

4. Proses menciptakan proses lain (proses anak). Proses yang menciptakan proses disebut proses induk (parent process). Proses anak-pun kembali dapat menciptakan proses-proses anak. Proses-proses dapat membentuk pohon hirarki proses.

Alasan – alasan penciptaan proses

Alasan penciptaan proses memiliki penyebab dan deskripsi seperti yang tercantum pada tabel di bawah ini:

Penyebab Penciptaan	Deskripsi
Terdapat batch baru	SO dengan kendali batch job, setelah menciptakan proses baru, kemudian melanjutkan membaca job selanjutnya
Satu pemakai interaktif logon	Seorang pemakai pada satu terminal sedang melakukan logon ke sistem
Sistem operasi menciptakan proses untuk memberi layanan	SO menciptakan proses untuk memenuhi satu fungsi pada program pemakai, tanpa mengharuskan pemakai menunggu
Proses menciptakan proses anak	Untuk mencapai modularitas atau mengekploitasi kongkurensi, program pemakai memerintahkan pembuatan sejumlah proses.

Gambar VI. 1 Tabel Alasan – Alasan Penciptaan Proses

Penghancuran Proses (destroy a process)

Penghancuran proses melibatkan pembebasan proses dari sistem, yaitu: Sumber daya-sumber daya yang dipakai dikembalikan, Proses dihancurkan dari senarai atau tabel system, PCB dihapus (ruang memori PCB dikembalikan ke pool memori bebas)

Penghancuran lebih rumit bila proses telah menciptakan proses-proses lain. Terdapat dua pendekatan. Pendekatan pertama ada beberapa sistem, proses-proses turunan dihancurkan saat proses induk dihancurkan secara otomatis.Pendekatan kedua beberapa sistem lain menganggap proses anak independen terhadap proses induk Proses anak tidak secara otomatis dihancurkan saat proses induk dihancurkan.

Alasan Penghancuran proses

Dibawah ini adalah tabel alasan – alasan pengahncuran proses.

Selesainya proses secara normal	Proses mengeksekusi panggilan layanan SO untuk menandakan bahwa proses telah berjalan secara lengkap.
Batas waktu telah terlewati	Proses telah berjalan melebihi batas waktu total yang dispesifikasikan. Terdapat banyak kemungkinan untuk tipe waktu yang diukur, termasuk waktu total yang dijalani ("walk clock time") jumlah waktu yang dipakai untuk eksekusi, dan jumlah waktu sejak pemakai terakhir kali memberi masukan (pada proses interaktif) .
Memori tidak tersedia	Proses memerlukan memori lebih banyak daripada yang dapat disediakan oleh sistem.
Pelanggaran terhadap batas memori	Proses memcoba mengakses lokasi memori yang tidak diijinkan untuk diakses
Terjadi kesalahan karena pelanggaran proteksi	Proses berusaha menggunakan sumber daya atau file yang tidak diijinkan dipakainya, atau proses mencoba menggunakannya tidak untuk peruntukannya, seperti menulis file read only
Terjadi kesalahan aritmatika	Proses mencoba perhitungan terlarang, seperti pembagian dengan nol, atau mencoba menyimpan angka yang lebih besar daripada yang ddapat diakomodasi oleh H/W
Waktu telah kadaluwarsa	Proses telah menunggu lebih lama daripada maksimum yang telah ditentukan untuk terjadinya suatu kejadian spesfiik
Terjadi kegagalan masukan/keluaran	Kesalahan muncul pada masukan atau keluaran, seperti ketidakmampuan menemukan file, kegagalan membaca atau mennulis setelah sejumlah maksimum percobaan yang ditentukan (misalnya area rusak didapatkan pada tape,atau operasi tidak valid seperti membaca dari line printer)
Intruksi yang tidak benar	Proses berusaha mengeksekusi instruksi yang tidak ada (sering sebagi akibat pencabangan ke daerah data dan berusaha mengeksekusi data tersebut)
Terjadi usaha memakai instruksi yang tidak diijinkan	Proses berusaha mengeksekusi instruksi yang disimpan untuk SO
Kesalahan penggunaan data	Bagian data adalah tipe yang salah atau tidak diinisialisasi
Diintervensi oleh SO atau operator	Untuk suatu alasan, operator atau sistem operasi mengakhiri proses (misalnya terjadi deadlock)

Berakhirnya proses induk	Ketika parent berakhir. SO mungkin dirancannng secara otomatis mengakhiri semua anak proses dari parent itu
Atas permintaan proses induk	Parent process biasanya mempunyai otoritas mengakhiri suatu anak proses

Tabel Alasan – Alasan Penghancuran Proses

Penundaan Proses (suspend a process)

Penundaan (suspension) adalah operasi penting dan telah diterapkan dengan beragam cara. Penundaan dapat diinisialisasi oleh proses itu sendiri atau proses lain. Penundaan biasanya berlangsung singkat dan sering dilakukan sistem untuk memindahkan proses-proses tertentu guna mereduksi beban sistem selama beban puncak. Proses yang ditunda (suspended process) tidak berlanjut sampai proses lain me-resume. Untuk jangka panjang, sumber daya-sumber daya proses dibebaskan. Pada sistem monoprocessor, proses running dapat men-suspend dirinya sendiri karena lak ada proses lain yang juga running yang dapat memerintahkan suspend. Pada sistem multiprocessor, proses running dapat di-suspend proses running lain pada pemroses berbeda. Proses ready hanya dapat di-suspend oleh proses lain.

Proses Pelanjutan Kembali (resume a process)

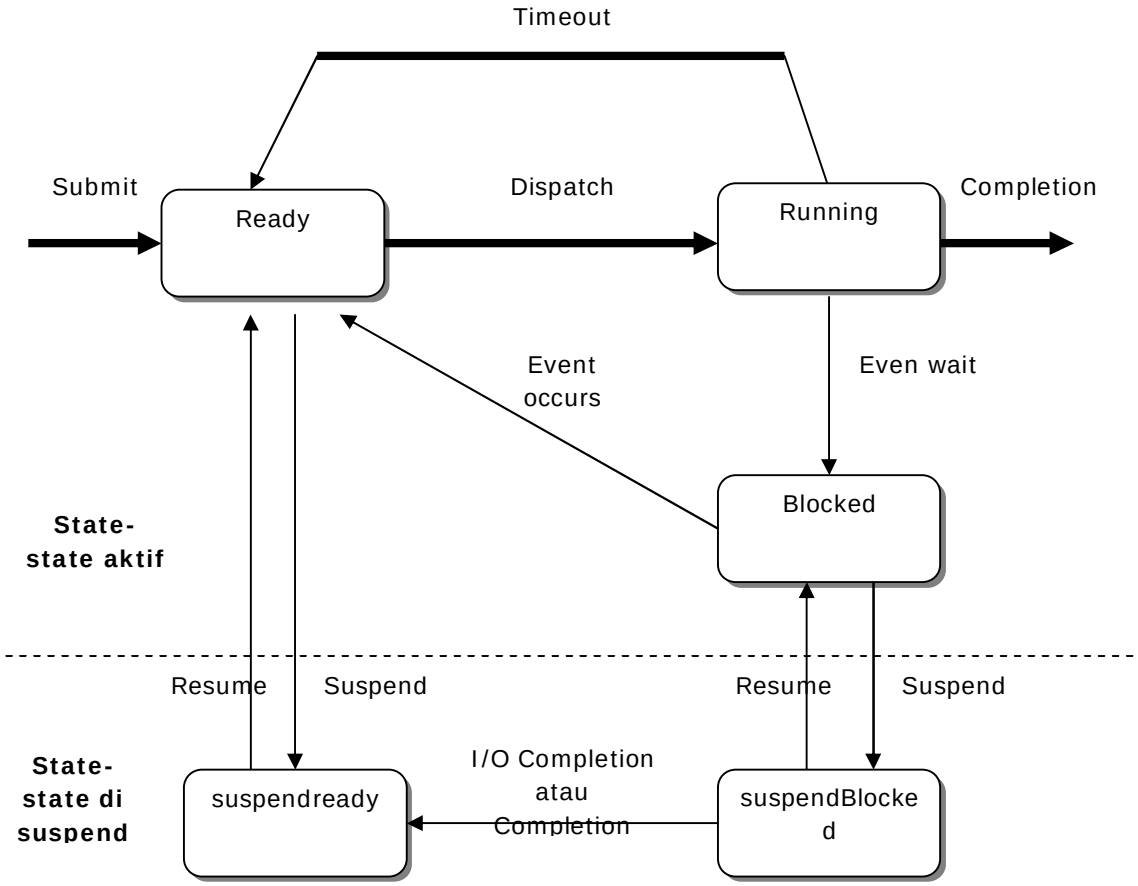
Jika sistem berfungsi secara buruk dan mungkin gagal maka proses-proses dapat di- suspend agar di-resume setelah masalab diselesaikan. Pemakai yang ragu / khawatir mengenai basil proses dapat men-suspend proses [bukan membuang (abort) proses]. Saat pemakai yakin proses akan berfungsi secara benar maka dapat me-resume (melanjutkan kembali di instruksi saat di-suspend) proses yang di-suspend.

Sebagai tanggapan terhadap fluktuasi jangka pendek beban sistem, beberapa proses dapal di-suspend dan di- resume saat beban kembali ke tingkat normal.

Proses Blocked

Pada proses blocked terdapat transisi menjadi suspendedblocked. Pilihan ini dirasa aneh. Apakah tidak cukup menunggu selesainya operasi masukan/keluaran atau kejadian yang membual proses ready atau suspendedready? Bukankah state blocked, readyblocked, suspendedblocked sama-sama tidak mendapatjatah waktu pemroses? Kenapa dibedakan ?. Jawabannya adalah karena penyelesaian operasi masukan/keluaran bagi proses blocked mungkin tak pernah terjadi atau dalam waktu tak terdefinisikan sehingga lebih baik di-suspend agar sumber daya-sumber daya yang dialokasikan untuk proses tersebut dapat digunakan proses-proses lain.

Proses blocked di-suspend sistem atau secara manual menjadi suspendedblocked. Bila akhirnya operasi masukan/keluaran berakhir maka segera proses suspendedblocked mengalami transisi. Karena resume dan suspend mempunyai prioritas tinggi maka transisi segera dilakukan. Suspend dan resume dapat digunakan untuk menyeimbangkan beban sistem saat mengalami lonjakan di atas normal.



Gambar VI. 3 Diagram State Lanjut

Menjadwalkan proses

Penjadwalan proses merupakan kumpulan kebijaksanaan dan mekanisme di sistem operasi yang berkaitan dengan urutan kerja yang dilakukan sistem komputer. Adapun penjadwalan bertugas memutuskan :

a. Proses yang harus berjalan

b. Kapan dan selama berapa lama proses itu berjalan

Kriteria untuk mengukur dan optimasi kinerje penjadwalan :

Adil (fairness)

Adalah proses-proses yang diperlakukan sama, yaitu mendapat jatah waktu pemroses yang sama dan tak ada proses yang tak kebagian layanan pemroses sehingga mengalami kekurangan waktu.

Efisiensi (eficiency)

Efisiensi atau utilisasi pemroses dihitung dengan perbandingan (rasio) waktu sibuk pemroses.

Waktu tanggap (response time)

Waktu tanggap berbeda untuk :

1. Sistem interaktif

Didefinisikan sebagai waktu yang dihabiskan dari saat karakter terakhir dari perintah dimasukkan atau transaksi sampai hasil pertama muncul di layar. Waktu tanggap ini disebut terminal response time.

2. Sistem waktu nyata

Didefinisikan sebagai waktu dari saat kejadian (internal atau eksternal) sampai instruksi pertama rutin layanan yang dimaksud dieksekusi, disebut event response time.

Turn around time

Adalah waktu yang dihabiskan dari saat program atau job mulai masuk ke system sampai proses diselesaikan sistem. Waktu yang dimaksud adalah waktu yang dihabiskan di dalam sistem, diekspresikan sebagai penjumlah waktu eksekusi (waktu pelayanan job) dan waktu menunggu, yaitu : Turn arround time = waktu eksekusi + waktu menunggu.

Throughput

Adalah jumlah kerja yang dapat diselesaikan dalam satu unit waktu. Cara untuk mengekspresikan throughput adalah dengan jumlah job pemakai yang dapat dieksekusi dalam satu unit / interval waktu.

Kriteria-kriteria tersebut saling bergantung dan dapat pula saling bertentangan sehingga tidak dimungkinkan optimasi semua kriteria secara simultan. Contoh : untuk memberi waktu tanggap kecil memerlukan penjadwalan yang sering beralih ke antara proses-proses itu. Cara ini meningkatkan overhead sistem dan mengurangi throughput.

Oleh karena itu dalam menentukan kebijaksanaan perancangan penjadwalan sebaiknya melibatkan kompromi diantara kebutuhan-kebutuhan yang saling bertentangan. Kompromi ini bergantung sifat dan penggunaan sistem komputer.

Sasaran penjadwalan

Sasaran penjadwalan berdasarkan kriteria-kriteria optimasi harus :

a. Menjamin tiap proses mendapat pelayanan dari pemroses yang adil.

b. Menjaga agar pemroses tetap dalam keadaan sibuk sehingga efisiensi mencapai maksimum. Pengertian sibuk adalah pemroses tidak menganggur, termasuk waktu yang dihabiskan untuk mengeksekusi program pemakai dan sistem operasi.

c. Meminimalkan waktu tanggap.

d. Meminimalkan turn arround time.

e. Memaksimalkan jumlah job yang diproses persatu interval waktu. Lebih besar angka throughput, lebih banyak kerja yang dilakukan sistem.

Tipe Penjadwalan

Terdapat 3 tipe penjadwal berada secara bersama-sama pada sistem operasi yang kompleks, yaitu:

1. Penjadwal jangka pendek (short term scheduller)

Bertugas menjadwalkan alokasi pemroses di antara proses-proses ready di memori utama. Penjadwalan dijalankan setiap terjadi pengalihan proses untuk memilih proses berikutnya yang harus dijalankan.

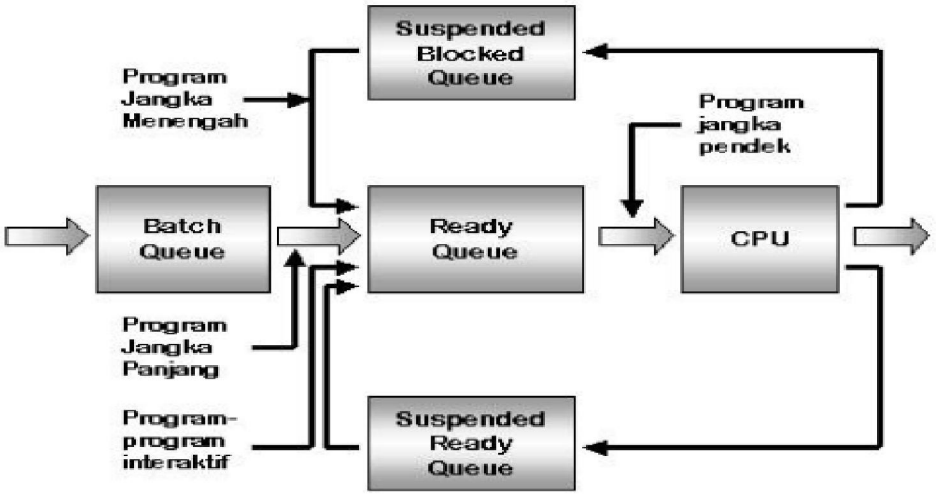
2. Penjadwal jangka menengah (medium term scheduller)

Setelah eksekusi selama suatu waktu, proses mungkin menunda sebuah eksekusi karena membuat permintaan layanan masukan / keluaran atau memanggil suatu system call. Proses-proses tertunda tidak dapat membuat suatu kemajuan menuju selesai sampai kondisi-kondisi yang menyebabkan tertunda dihilangkan. Agar ruang memori dapat bermanfaat, maka proses dipindah dari memori utama ke memori sekunder agar tersedia ruang untuk proses-proses lain. Kapasitas memori utama terbatas untuk sejumlah proses aktif.

Aktivitas pemindahan proses yang tertunda dari memori utama ke memori sekunder disebut swapping. Proses-proses mempunyai kepentingan kecil saat itu sebagai proses yang tertunda. Tetapi, begitu kondisi yang membuatnya tertunda hilang dan proses dimasukkan kembali ke memori utama dan ready.

3. Penjadwal jangka panjang (long term scheduller)

Penjadwal ini bekerja terhadap antrian batch dan memilih batch berikutnya yang harus dieksekusi. Batch biasanya adalah proses-proses dengan penggunaanm sumber daya yang intensif (yaitu waktu pemroses, memori, perangkat masukan / keluaran), program-program ini berprioritas rendah, digunakan sebagai pengisi (agar pemroses sibuk) selama periode aktivitas job-job interaktif rendah.



Gambar VI. 4 Diagram Penjadwalan

Strategi penjadwalan

Terdapat dua strategi penjadwalan, yaitu :

1. Penjadwalan nonpreemptive (run to completion)

Proses diberi jatah waktu oleh pemroses, maka pemroses tidak dapat diambil alih oleh proses lain sampai proses itu selesai. ketika CPU memberikan kepada proses itu tidak bisa ditunda hingga selesai

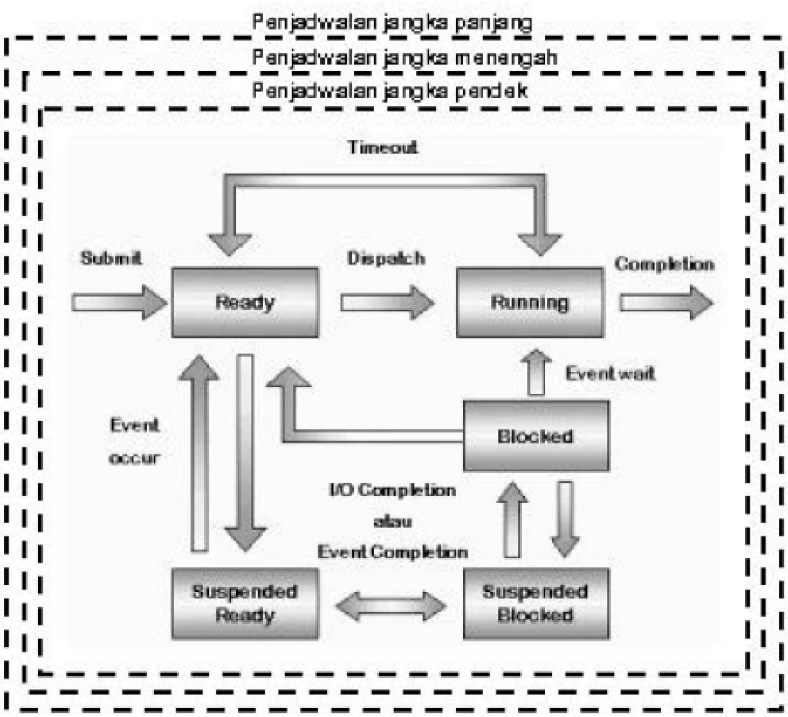
2. Penjadwalan preemptive

Proses diberi jatah waktu oleh pemroses, maka pemroses dapat diambil alih proses lain, sehingga proses disela sebelum selesai dan harus dilanjutkan menunggu jatah waktu pemroses tiba kembali pada proses itu. bila sebuah proses datang dengan waktu prose lebih rendah dibandingkan dengan waktu proses yang sedang dieksekusi ooleh CPU maka proses yang waktunya lebih rendah mendapatkan prioritas. Skema ini disebut juga Short -Remaining Time First (SRTF). Berguna pada sistem dimana proses-proses yang mendapat perhatian/tanggapan pemroses secara cepat, misalnya :

a. Pada sistem realtime, kehilangan interupsi (tidak layani segera) dapat berakibat fatal.

b. Pada sistem interaktif, agar dapat menjamin waktu tanggap yang memadai. Penjadwalan secara preemptive baik tetapi harus dibayar mahal.

Proses memerlukan overhead (banyak tabel yang dikelola). Supaya efektif, banyak proses harus berada di memori utama sehingga proses-prose tersebut dapat segera running begitu diperlukan. Menyimpan banyak proses tak running benar-benar di memori utama merupakan suatu overhead tersendiri.



Gambar VI. 5 Diagram Strategi Penjadwalan

Sinkronisasi

Komunikasi antara proses membutuhkan place by calls untuk mengirim dan menerima data primitive. Terdapat design yang berbeda-beda dalam implementasi setiap primitive. Pengiriman pesan mungkin dapat diblok (blocking) atau tidak dapat dibloking (nonblocking) - juga dikenal dengan nama sinkron atau asinkron.

- Pengiriman yang diblok : Proses pengiriman di blok sampai pesan diterima oleh proses penerima (receiving process) atau oleh mailbox.
- Pengiriman yang tidak diblok : Proses pengiriman pesan dan mengkalkulasi operasi.
- Penerimaan yang diblok : Penerima mem blok sampai pesan tersedia.
- Penerimaan yang tidak diblok : Penerima mengembalikan pesan valid atau null.

Ketika dalam keadaan sinkron, terjadi dua kejadian:

1. Blocking send, yaitu pemblokiran pengirim sampai pesan sebelumnya diterima.
2. Blocking receive, yaitu pemblokiran penerima sampai terdapat pesan yang akan dikirim.

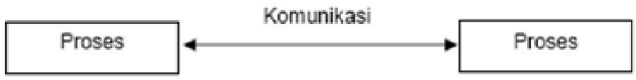
Sedangkan untuk keadaan asinkron, yang terjadi adalah:

1. Non-blocking send, yaitu pengirim dapat terus mengirim pesan tanpa memperdulikan apakah pesan sebelumnya sampai atau tidak.

2. Non-blocking receive, yaitu penerima menerima semua pesan baik berupa pesan yang valid atau pesan yang salah (null).

Komunikasi antar proses

(Inter Process Communinication / IPC) :



Beberapa proses biasanya berkomunikasi dengan proses lainnya. Contohnya pada shell pipe line. Output dari proses pertama harus diberikan kepada proses ke dua dan seterusnya. Pada beberapa sistem operasi, proses-proses yang bekerja bersama sering sharing (berbagi) media penyimpanan, dimana suatu proses dapat membaca dan menulis pada shared storage (main memory atau files).

KOMUNIKASI LANGSUNG

Setiap proses yang ingin berkomunikasi harus memiliki nama yang bersifat eksplisit baik penerimaan atau pengirim dari komunikasi tersebut. Dalam konteks ini, pengiriman dan penerimaan pesan secara primitive dapat dijabarkan sebagai :

- Send (P, message) - mengirim sebuah pesan ke proses P.
- Receive (Q, message) - menerima sebuah pesan dari proses Q.

Sebuah jaringan komunikasi pada bahasan ini memiliki beberapa sifat, yaitu :

- Sebuah jaringan yang didirikan secara otomatis diantara setiap pasang dari proses yang ingin dikomunikasikan.Proses tersebut harus mengetahui identitas dari semua yang ingin dikomunikasikan.
- Sebuah jaringan adalah terdiri dari penggabungan 2 proses.
- Diantara setiap pesan dari proses terdapat tepat sebuah jaringan.

Pembahasan ini memperlihatkan sebuah cara simetris dalam pemberian alamat. Oleh karena itu, baik keduanya yaitu pengirim dan penerima proses harus memberi nama bagi yang lain untuk berkomunikasi, hanya pengirim yang memberikan nama bagi penerima sedangkan penerima tidak menyediakan nama bagi pengirim. Dalam konteks ini, pengirim dan penerima secara sederhana dapat dijabarkan sebagai :

- Send (P, message) - mengirim sebuah pesan kepada proses P.
- Receive (id, message) - menerima sebuah pesan dari semua proses. Variabel id diatur sebagai nama dari proses dengan komunikasi.

Komunikasi Tidak Langsung

Dengan komunikasi tidak langsung, pesan akan dikirimkan pada dan diterima dari / melalui mailbox (Kotak Surat) atau terminal-terminal, sebuah mailbox dapat dilihat secara abstrak sebagai sebuah objek didalam setiap pesan yang dapat ditempatkan dari proses dan dari setipap pesan yang bias dipindahkan. Setiap kotak surat memiliki sebuah identifikasi (identitas) yang unik, sebuah proses dapat berkomunikasi dengan beberapa proses lain melalui sebuah nomor dari mailbox yang berbeda. Dua proses dapat saling berkomunikasi apabila kedua proses tersebut sharing mailbox. Pengirim dan penerima dapat dijabarkan sebagai :

- Send (A, message) - mengirim pesan ke mailbox A.
- Receive (A, message) - menerima pesan dari mailbox A.

Dalam masalah ini, link komunikasi mempunyai sifat sebagai berikut :

- Sebuah link dibangun diantara sepasang proses dimana kedua proses tersebut membagi mailbox.
- Sebuah link mungkin dapat berasosiasi dengan lebih dari 2 proses.
- Diantara setiap pasang proses komunikasi, mungkin terdapat link yang berbeda-beda, dimana setiap link berhubungan pada satu mailbox.

Misalkan terdapat proses P1, P2 dan P3 yang semuanya share mailbox. Proses P1 mengirim pesan ke A, ketika P2 dan P3 masing-masing mengeksekusi sebuah kiriman dari A. Proses mana yang akan menerima pesan yang dikirim P1?. Jawabannya tergantung dari jalur yang kita pilih :

- Mengijinkan sebuah link berasosiasi dengan paling banyak 2 proses.

- Mengijinkan paling banyak 1 proses pada suatu waktu untuk mengeksekusi hasil kiriman (receive operation).

- Mengijinkan sistem untuk memilih secara mutlak proses mana yang akan menerima pesan (apakah itu P2 atau P3 tetapi tidak keduanya, tidak akan menerima pesan). Sistem mungkin mengidentifikasi penerima kepada pengirim.

Mailbox mungkin dapat dimiliki oleh sebuah proses atau sistem operasi. Jika mailbox dimiliki oleh proses, maka kita mendefinisikan antara pemilik (yang hanya dapat menerima pesan melalui mailbox) dan pengguna dari mailbox (yang hanya dapat mengirim pesan ke mailbox). Selama setiap mailbox mempunyai kepemilikan yang unik, maka tidak akan ada kebingungan tentang siapa yang harus menerima pesan dari mailbox. Ketika proses yang memiliki mailbox tersebut diterminasi, mailbox akan hilang. Semua proses yang mengirim pesan ke mailbox ini diberi pesan bahwa mailbox tersebut tidak lagi ada.

Dengan kata lain, mempunyai mailbox sendiri yang independent, dan tidak melibatkan proses yang lain. Maka sistem operasi harus memiliki mekanisme yang mengijinkan proses untuk melakukan hal-hal dibawah ini :

- Membuat mailbox baru.

- Mengirim dan menerima pesan melalui mailbox.

- Menghapus mailbox.

Proses yang membuat mailbox pertama kali secara default akan memiliki mailbox tersebut. Untuk pertama kali, pemilik adalah satu-satunya proses yang dapat menerima pesan melalui mailbox ini. Bagaimanapun, kepemilikan dan hak menerima pesan mungkin dapat dialihkan ke proses lain melalui sistem pemanggilan.

- Pengiriman yang diblok : Proses pengiriman di blok sampai pesan diterima oleh proses penerima (receiving process) atau oleh mailbox.

- Pengiriman yang tidak diblok : Proses pengiriman pesan dan mengkalkulasi operasi.

- Penerimaan yang diblok : Penerima mem blok samapai pesan tersedia.

- Penerimaan yang tidak diblok : Penerima mengembalikan pesan valid atau null.

Penyangga

Dalam setiap jenis komunikasi, baik langsung atau tidak langsung, penukaran pesan oleh proses memerlukan antrian sementara. Pada dasarnya, terdapat tiga cara untuk mengimplementasikan antrian tersebut:

1. Kapasitas Nol. Antrian mempunyai panjang maksimum nol, sehingga tidak ada penungguan pesan (message waiting). Dalam kasus ini, pengirim harus memblok sampai penerima menerima pesan.

2. Kapasitas Terbatas. Antrian mempunyai panjang yang telah ditentukan, paling banyak n pesan dapat dimasukkan. Jika antrian tidak penuh ketika pesan dikirimkan, pesan yang baru akan menimpa, dan pengirim pengirim dapat melanjutkan eksekusi tanpa menunggu. Link mempunyai kapasitas terbatas. Jika link penuh, pengirim harus memblok sampai terdapat ruang pada antrian.

3. Kapasitas Tak Terbatas. Antrian mempunyai panjang yang tak terhingga, sehingga semua pesan dapat menunggu disini. Pengirim tidak akan pernah di blok.

Interprocess Communication (IPC)

Mekanisme proses untuk komunikasi dan sinkronisasi aksi. Sistem Pesan – komunikasi proses satu dengan yang lain dapat dilakukan tanpa perlu pembagian data. IPC menyediakan dua operasi :

- send(message) – pesan berukuran pasti atau variabel
- receive(message)

Jika P dan Q melakukan komunikasi, maka keduanya memerlukan :

- Membangun jalur komunikasi diantara keduanya
- Melakukan pertukaran pesan melaui send/receive

Implementasi jalur komunikasi ada 2, yaitu :

- physical (shared memory, hardware bus)
- logical (logical properties)

Masalah-masalah pada IPC

1. Rase condition

Suatu kondisi dimana dua atau lebih proses mengakses data pada saat yang bersamaan dan hasil akhirnya tidak sesuai dengan yang dikehendaki. Contoh Race Condition: Print spooler, yaitu berupa kumpulan data-data yang akan dicetak

2. Critical Section

Bagian dari program yang mengakses shared memory / data yang dapat menyebabkan terjadinya race condition.Empat kondisi untuk mencegah race condition, antara lain :

- Tidak ada 2 proses yang memasuki critical sectionnya secara bersamaan / simultan.
- Tidak ada asumsi yang dibuat yang berhubungan dengan kecepatan dan jumlah CPU
- Tidak ada proses yang berjalan diluar critical section-nya yang dapat memblokir proses-proses lain
- Tidak ada proses yang menunggu selamanya untuk masuk ke critical section-nya.

MANAJEMEN PROSES PADA LINUX DAN WINDOWS

Linux tidak mempunyai sistem pembedaan drive A, B, C dan seterusnya seperti pada Windows, jadi ketika linuxer masuk ke sistem operasi linux nanti jangan kaget kalau tiba-tiba drive C atau D nya hilang.

Sistem operasi linux bukanlah untuk mengeja alphabet dari A sampai dengan Z, justru linux mengenali komputer dengan sistem direktori-direktori, baik mulai dari harddisk, floppy disk drive, CD-ROM dan lainnya. Misalnya saja penamaan untuk CD-ROM, linuxer bisa cari di direktori /mnt/cdrom atau floppy disk drive di direktori /mnt/floppy. Linux menggunakan sistem (/) forward slash, beda sekali dengan Windows yang menggunakan system (\) backward slash.

Linux mempunyai sifat case-sensitive, yang berarti huruf besar dan huruf kecil mempunyai arti yang berbeda. Jadi huruf A dan a mempunyai arti yang berbeda. Linux tidak mempunyai .exe seperti di Windows. Jadi jangan kebingungan nanti kalau linuxer ingin menjalankan perintah-perintah linux. Jangan sampai berkata “koq, dot exe nya ga ada ?”. Linux mempunyai sistem executable file tersendiri, jadi jika ingin mengetahui suatu file bisa di

execute atau tidak, adalah dari attributnya file yang bersangkutan, jika attrributnya execute berarti bisa dijalankan. Cara mengetahuinya bisa dari perintah ls -l atau dari chmod. Linux mempunyai banyak GUI (Graphical User Interface) Window yang berbeda. Diantaranya ada KDE, GNOME, Sawfish, Enlightenment dan lain sebagainya. Tidak seperti MS Windows yang hanya mempunyai satu GUI. Misalnya GUI Windows 98 tidak bisa mempunyai GUI Windows 2000 atau GUI Windows XP yang wah. Window di linux mempunyai istilah tersendiri, yaitu Xwindow. Di Xwindow linuxer mampu menjalankan KDE atau GNOME, atau bertukaran sesuai dengan keinginan.

Proses

- Konsep Proses
- Penjadualan Eksekusi Proses
- Operasi pada Proses
- Proses yang saling Bekerjasama (Cooperating Processes)
- Komunikasi Antar Proses (Interprocess Communication)
- Komunikasi pada Sistem Client-Server

Konsep Proses

- Sistem operasi menjalankan banyak dan beragam program :
    - Batch system –jobs
    - Time-shared systems –user programs atau tasks
    - Istilah pada buku teks: job, task danprocess (dapatdiartikan sama)
- Proses adalah program yang dieksekusi;
    - Aktif (proses=>memori) vs pasif (program => file)
    - Instruksi pada program (code) akan dieksekusi secara berurut (sekwensial) sesuai dengan “line code” (stored program concept).
- Proses lebih dari “program code yang aktif”:
    - Melacak posisi instruksi (sequential execution): program counter
    - Menyimpan data sementara var., parameter, return value: stack
    - Menyimpan data (initial, global variable dll): data section
    - Menyimpan status proses(contoh, aktif, wait I/O request dll.)

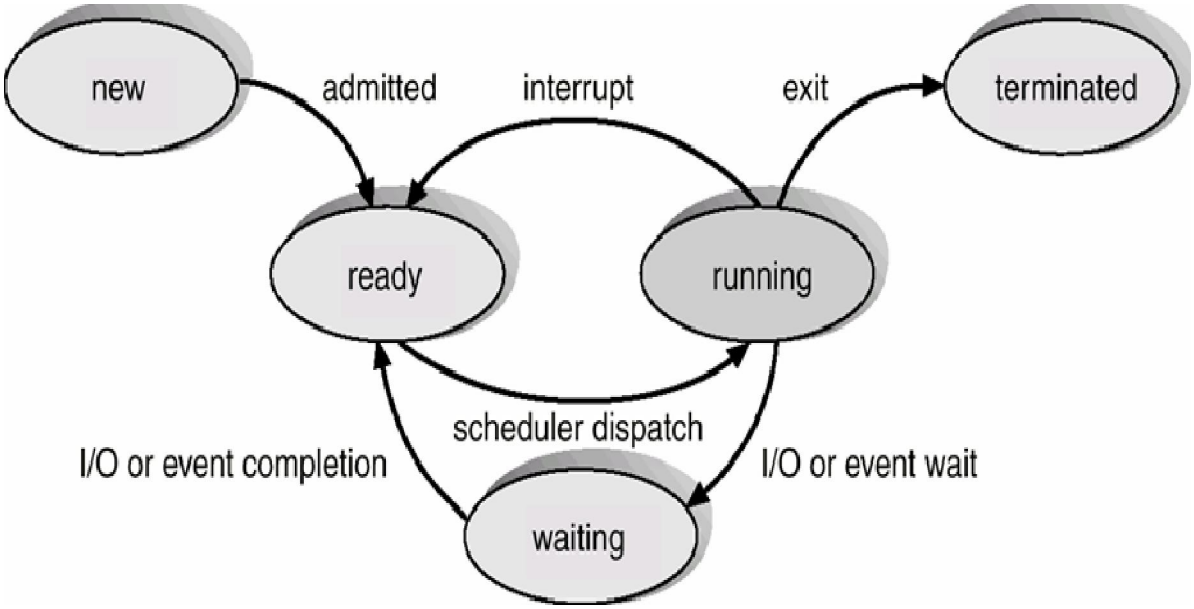
Status Proses

- Saat-saat proses dijalankan (executed) maka status dari proses akan berubah
    - Status proses tidak selamanya aktif menggunakan CPU).

- Sering proses menunggu I/O complete => status wait, sebaiknya CPU diberikan kepada proses yang lain.
- Mendukung multi-tasking –utilisasi CPU dan I/O

- Status proses (antara lain):
  - new: proses dibuat.
  - running: instruksi dieksekusi.
  - waiting: proses menunggu beberapa event yang akan terjadi
  - ready: proses menunggu jatah waktu dari prosessor
  - terminated: proses selesai dieksekusi.

Diagram Status Proses



Informasi Proses

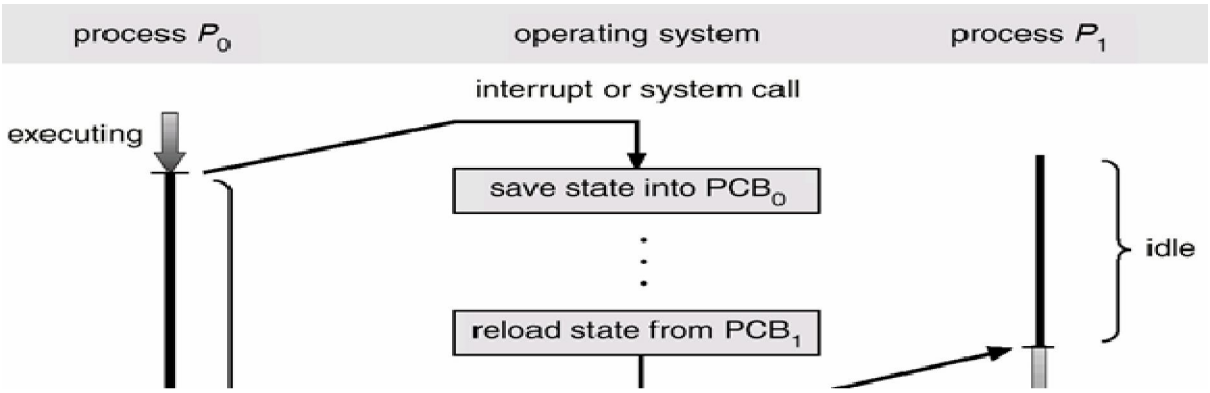
Dimanakah informasi proses disimpan?

- Data struktur dari OS dalam bentuk table :
  - Satu entry table/linked list => struktur data untuk menampung informasi satu proses (array of structure).
  - Setiap entry pada tabel proses menyimpan satu proses. Contoh: MINIX (src/kernel/proc.h) => structproc { …};
- Informasi yang disimpan:

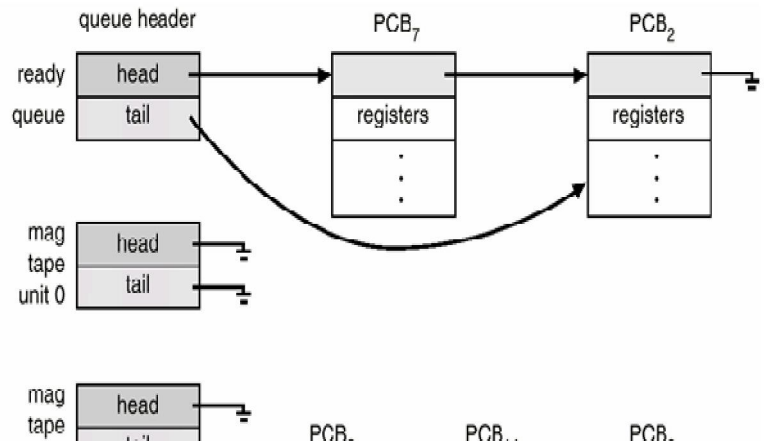
- Informasi internal CPU: isi register-register, program counter, status CPU dll (umumnya dalam bentuk stack frame).
- Identifikasi proses: nama proses, proses number/index, prosesid.
- Identifikasi proses: nama proses, proses number/index, prosesid.
- Accounting dan timer: user time, system time, alarm etc.
- Resources: memory & file management.

Process Control Block (PCB)

CPU Switch Dari Satu Proses ke Proses Lainnya


process P_0
operating system
process P_1
interrupt or system call
executing
save state into PCB_0
reload state from PCB_1
idle

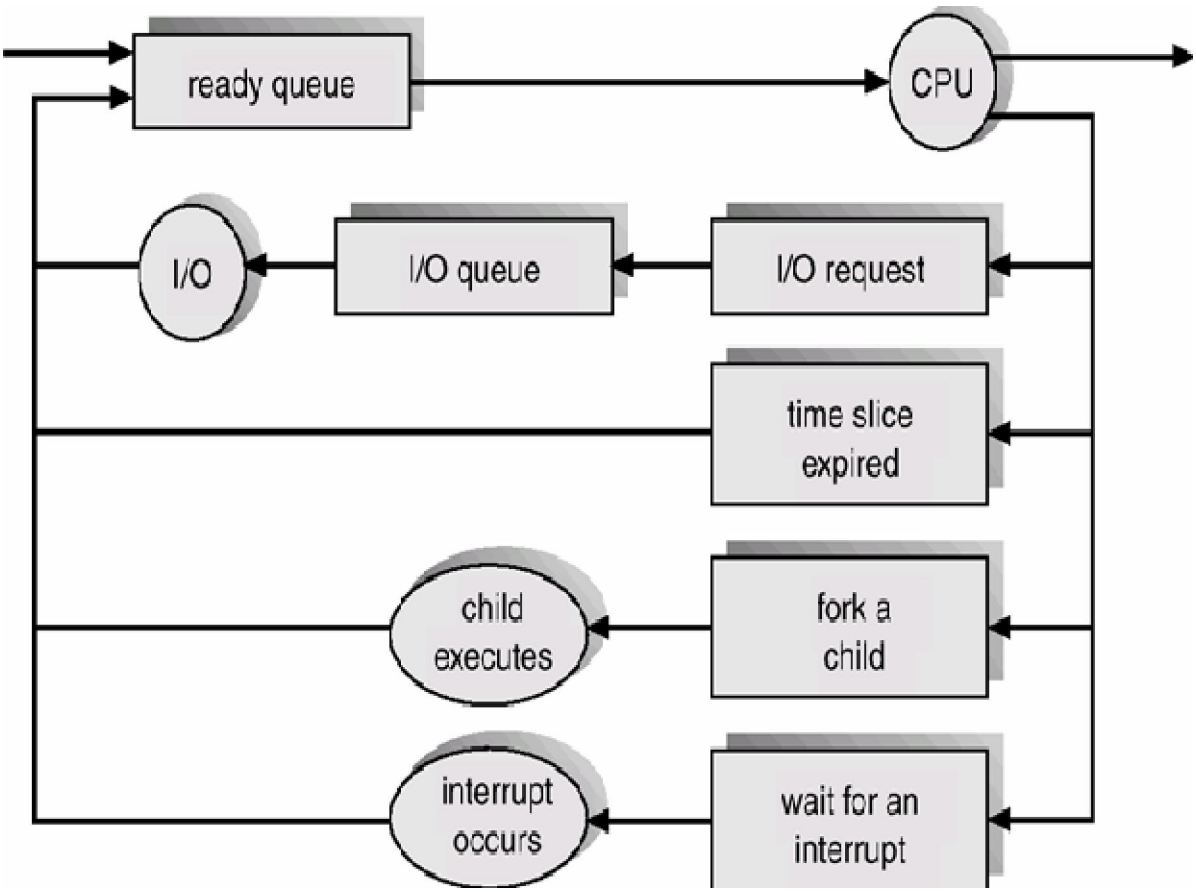
]


queue header
PCB_7
PCB_2
ready
queue
head
tail
registers
registers
mag
tape
unit 0
head
tail
mag
tape
head

Penjadualan Proses

- Proses dapat berubah status dan berpindah dari satu antrian ke antrian yang lain
    - Proses dengan status “ready” berada di Ready Queue
        - Menunggu giliran/dipilih oleh scheduler => menggunakan CPU
    - Selama eksekusi (status “run”) events yang dapat terjadi:
        - I/O request => I/O wait berada pada Device Queue
        - Create “child”proses=> Jalankan proses “child”, tunggu sampai proses selesai (wait)
        - Time slice expired => Waktu pemakaian CPU habis, interrupt oleh scheduler, proses akan berpindah ke Ready Queue

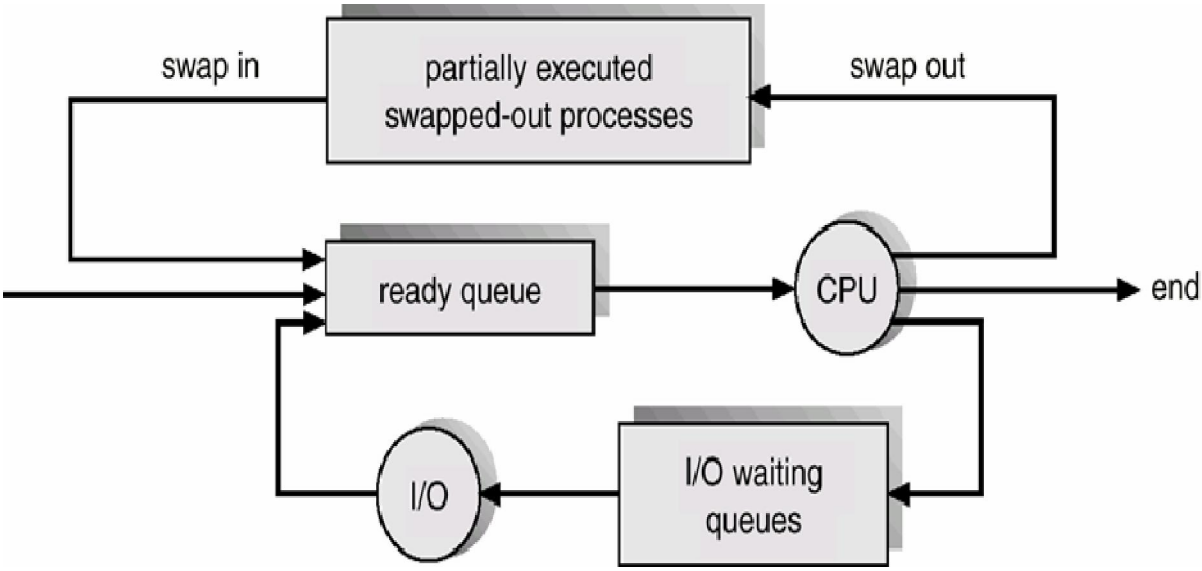
Representasi Penjadualan Proses



Penjadual / Schedulers

- Bagaimana schedulers memilih proses atau program (decision)?
    - Lebih dari satu proses atau program yang akan dijalankan?
- Long-term scheduler (or job scheduler) –memilih proses/program yang manayang akan diload dan berada di ready queue.
    - Kemungkinan terdapat proses atau job baru.
    - Kemungkinan proses dipindah kan dari memori ke disk (swap out).
- Short-term scheduler (or CPU scheduler) –memilih proses yang manayang berada di ready queue akan“run” (mendapatkan jatah CPU).

Penjadualan Jangka Menengah



Penjadual / Schedulers (Cont.)

- Long-term scheduler tidak sering (proses baru) (seconds, minutes) => (may be slow).
    - The long-term scheduler controls the degree of multiprogramming => berapa banyak proses yang dapat aktif (berada dimemori)
- Short-term scheduler dijalankan sangat sering (milliseconds) => giliran pemakaian CPU dari proses-proses yang siap
    - Pada saat terjadi penggantian alokasi CPU dari satu proses ke proses lain:

        - § Menyimpan informasi internal CPU dari proses yang akan digantikan (SAVE).
        - § Meload kembali informasi internal CPU dari proses yang akan menggantikan.
- Dikenal dengan istilah: context switch proses.

Alih Konteks / Context Switch

- Jika Scheduler switch keproses lain, maka sistim harus menyimpan “informasi” proses sekarang (supaya dapat dijalankan kembali)
- Load “informasi”dari proses baru yang berada di PCB
- Waktu Context-switch adalah overhead; sistem tidak melakukan pekerjaan saat terjadi switch.
    - o Sangat tergantung pada waktu di hardware
    - o OS modern mencari solusi untuk mengurangi overhead waktu switch proses

Pembuatan Proses

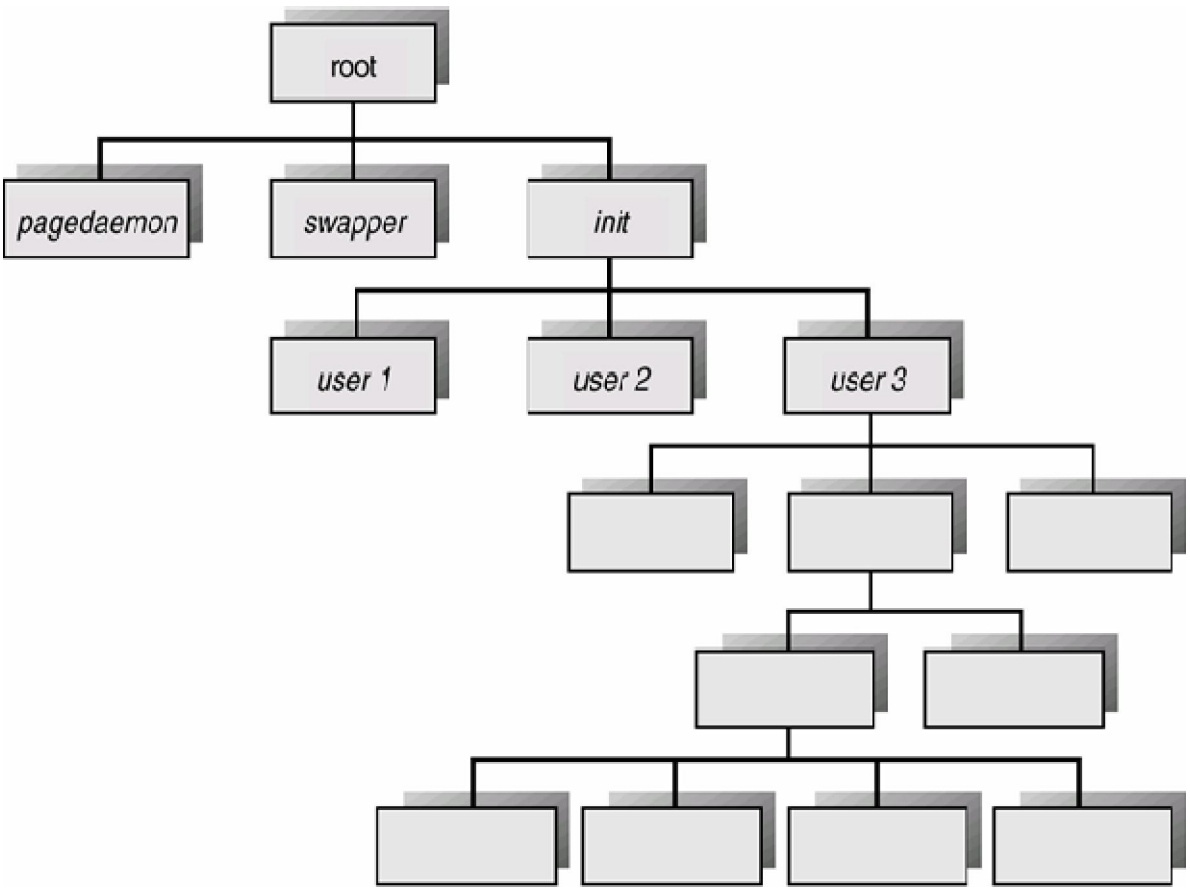
- Umumnya proses dapat membuat proses baru (child process).
    - o Child process dapat membuat proses baru.
    - o Terbentuk “tree” dari proses.
- Pilihan hubungan antara parent dan child proses:
    - o Resource sharing
        - § Parent dan child berbagi resource
        - § Children berbagi subset dari resource milik parents.
        - § Parent dan child tidak berbagi resource.
    - o Execution
        - § Parent dan children melakukan eksekusi secara serempak.
        - § Parent menunggu hingga children selesai.

Pembuatan Proses (Cont.)

- Address space
    - o Child menduplikasi parent.
    - o Child memiliki program yang diload kedalamnya.
- Contoh UNIX :
    - o Fork system call membuat proses baru

- Execve (EXEC) :
  - Menjalankan program spesifik yang lain
  - Nama program tersebut menjadi parameter dari system call
  - EXEC (sering diload sesudah menjalankan fork).
- Tahapan pembuatan proses baru:
  - Periksa apakah masih terdapat ruang pada PCB.
  - Mencoba mengalokasikan memori untuk proses baru.
  - Mengisi informasi untuk proses baru: nama proses, id, copy data dari parent dll.
  - Mencantumkan informasi proses ke kernel OS.

Proses Tree pada Sistem UNIX



Terminasi Proses

- Proses dapat berakhir:
  - Eksekusi instruksi terakhir (atau keluar: exit system call).

    - OS yang akan melakukan dealokasi (memory, file resources).
- UNIX (MINIX):
    - Output signal dari child keparent
    - Jika parent tidak menunggu (via wait system call), proses akan terminate tapi belum direlease dari PCB (status: ZOMBIE).
    - Proses dengan status ZOMBIE (parent telah terminate), akan menjadi child dari proses “init”.
- Parent dapat menghentikan eksekusi proses child secara paksa.
    - Parent dapat mengirim signal (abort, kill system call).

Kerjasama Proses

- Proses independent tidak mempengaruhi eksekusi proses yang lain
- Kerjasama proses dapat mempengaruhi atau dipengaruhi oleh eksekusi proses yang lain
- Keuntungan kerjasama proses:
    - Sharing informasi
    - Meningkatkan kecepatan komputasi
    - Modularitas
    - Kemudahan

Masalah Producer - Consumer

- Paradigma kerjasama proses–proses Producer menghasilkan informasi yang akan dikonsumsi oleh proses Consumer
    - Unbounded-buffer – tidak menggunakanbatasan ukurandibuffer.
        - § Consumer selalu dapat meminta item baru dan Producer selalu dapat menghasilkan item-item baru.
    - Bounded-buffer – menggunakan buffer dengan ukuran tertentu
        - § Consumer harus menunggu jika buffer kosong dan Producer harus menunggu jika buffer penuh

Bounded-Buffer – Solusi dari Shared Memory

- Shared data

```
#define BUFFER_SIZE 10
Typedef struct {
. . .
```

```
} item;
 item buffer[BUFFER_SIZE];
intin = 0;
intout = 0;
```

- Solution is correct, but can only use BUFFER_SIZE-1 elements

Bounded-Buffer – Proses Producer

```
item nextProduced;
while (1) {
while (((in + 1) % BUFFER_SIZE) == out) ;
/* do nothing */
buffer[in] = nextProduced;
in = (in + 1) % BUFFER_SIZE;
}
```

Bounded-Buffer – Proses Consumer

```
item nextConsumed;
while (1) {
while (in == out) ;
 /* do nothing */
nextConsumed= buffer[out];
out = (out + 1) % BUFFER_SIZE;
}
```

Interprocess Communication (IPC)

- Mekanisme proses untuk komunikasi dan sinkronisasi aksi
- Sistem Pesan–komunikasi proses satu dengan yang lain dapat dilakukan tanpa perlu pembagian data.
- IPC menyediakan dua operasi:
    - Send (message) – pesan berukuran pasti atau variabel
    - Receive (message)
- Jika P dan Q melakukan komunikasi, maka keduanya memerlukan:
    - Membangun jalur komunikasi diantara keduanya
    - Melakukan pertukaran pesan melaui send / receive
- Implementasi jalur komunikasi
    - physical (shared memory, hardware bus)

- logical (logical properties)

Komunikasi Langsung

- Proses harus diberi nama secara jelas:
  - Send (P, message) –kirim pesan ke proses P
  - Receive (Q, message) – terima pesan dari proses Q
- Properti jalur komunikasi
  - Jalur dibangun secara otomatis
  - Setiap jalur memiliki pasangan masing-masing dalam proses komunikasi
  - Jalur komunikasi tersebut biasanya directional

Komunikasi Tidak Langsung

- Pesan dikirim dan diterima melalui mailboxes (yang ditunjuk sebagai port)
  - Proses
  - Processes can communicate only if they share a mailbox.
- Properti jalur komunikasi
  - Jalur komunikasi hanya dibangun jika proses di-share dalam mailbox
  - Jalur merupakan gabungan beberapa proses
  - Setiap pasangan proses dibagi kedalam beberapa jalur komunikasi.

Komunikasi Tidak Langsung

- Operasi
  - Membuat mailbox baru
  - Mengirim dan menerima pesan melalui mailbox
  - Menghapus / memusnahkan mailbox
- Primitive didefinisikan: send (A, message) – kirim pesan ke mailbox A receive (A, message) – terima pesan dari mailbox A

Komunikasi Tidak Langsung

- Mailbox sharing
  - P1, P2,dan P3 berbagi (share) mailbox A.
  - P1, send; P2 and P3 receive.
  - Siapa yang mendapat pesan?
- Solusi
  - Memperbolehkan suatu jalur yang merupakan gabungan lebih dari dua proses

- Hanya meperbolehkan satu proses pada suatu waktu untuk mengeksekusi operasi receive .
- Memperbolehkan sistem untuk memilih receiver. Sender diberitahu siapa yang menjadi receiver.

Sinkronisasi

- Pesan yang disampaikan dapat diblok atau tidak (non-blocking)
- Blocking dikenal dengan synchronous.
- Non-blocking dikenal dengan asynchronous

Buffering

- Antrian pesan yang dihubungkan dalam suatu jalur, diimplementasikan dengan tiga jalan:
    1. Zero capacity –tidak ada pesan -Sender harus menunggu receiver (rendezvous).
    2. Bounded capacity –memiliki panjang yang terbatas (finite length) dari pesan. -Sender menunggu pada saat jalur penuh.
    3. Unbounded capacity –memiliki panjang tidak terbatas(infinite length) - Sender tidak pernah menunggu.

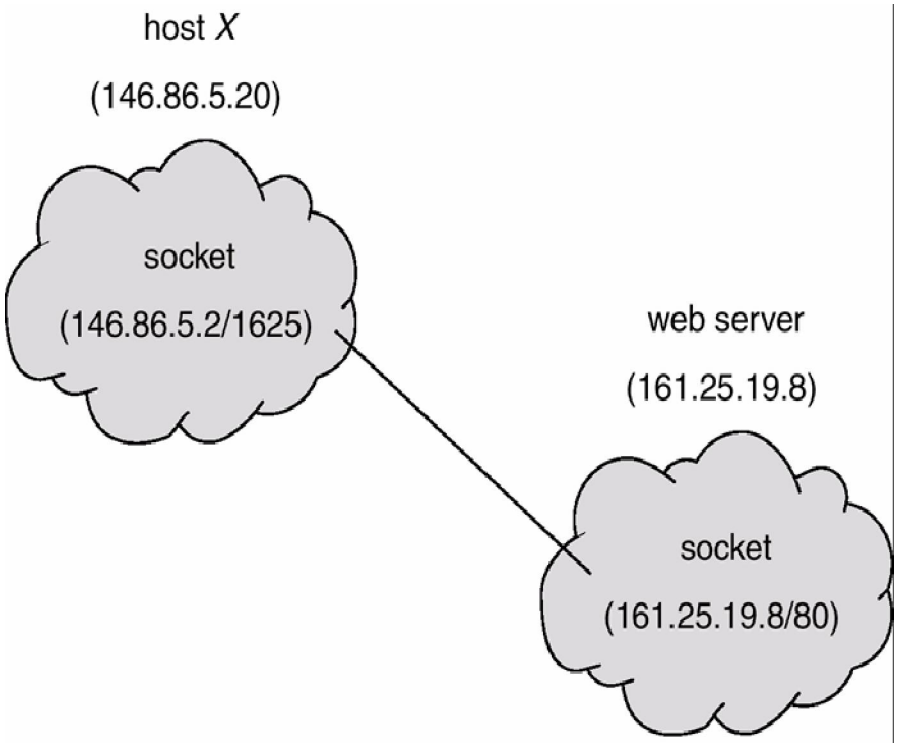
KomunikasiClient-Server

- Sockets
- Remote Procedure Calls (RPC)
- Remote Method Invocation (Java)

Sockets

- Suatu socket didefinisikan sebagai titik akhir (endpoint) komunikasi
- A socket is defined as an endpoint for communication.
- Gabungan IP address dan port
- Socket 161.25.19.8:1625 mengacu pada port 1625 pada host 161.25.19.8
- Komunikasi berada diantara pasangan socket

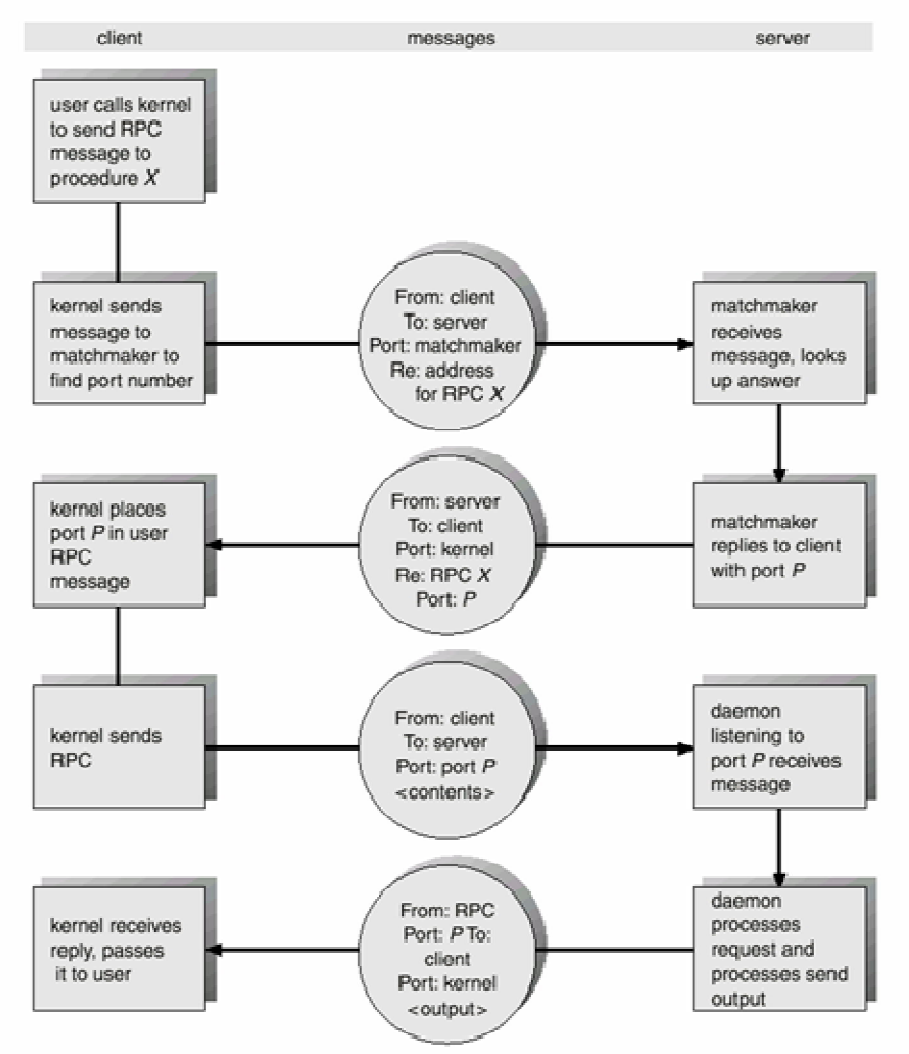
Komunikasi Socket



Remote Procedure Calls (RPC)

- Remote Procedure Call (RPC) adalah abstraksi pemanggilan prosedur diantara proses pada sistem jaringan
- Stubs–proxy sisi client untuk prosedur aktual pada server
- Stub sisi client ditempatkan diserver dengan parameter marshalls.
- Stub sisi server menerima pesan, membongkarnya dengan parameter marshall dan menjalankan prosedur pada server.

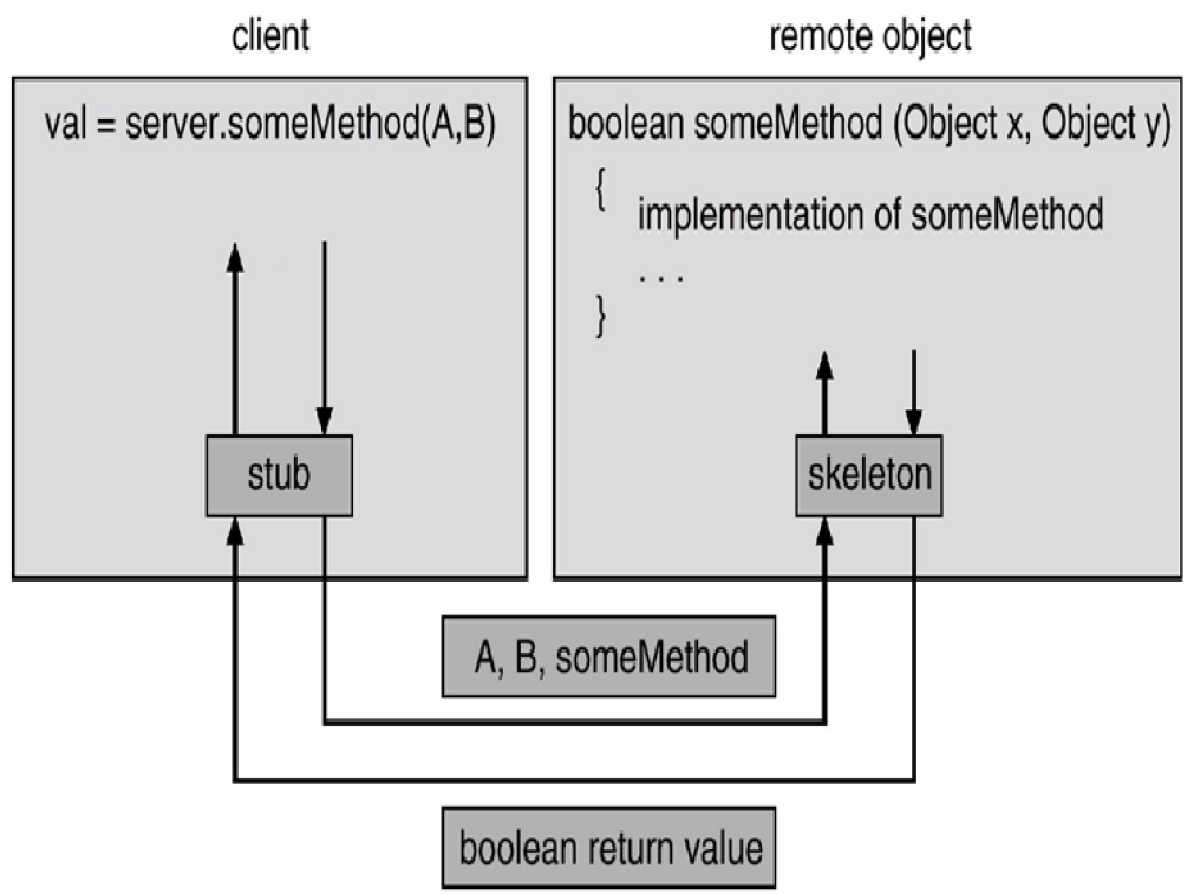
Eksekusi RPC



Remote Method Invocation (RMI)

- Remote Method Invocation (RMI) adalah mekanisme pada JAVA yang hampir sama dengan RPC
- RMI membolehkan program JAVA pada satu mesin untuk menggunakan metode untuk melakukan remote objek.

]



C. SOAL LATIHAN/TUGAS

1. Jelaskan pengertian proses?
2. Jelaskan status proses?
3. Jelaskan penjadwalan proses?

D. DAFTAR PUSTAKA

Buku

Bambang Hariyanto. 1997. Sistem Operasi, Bandung:Informatika Bandung.

Dali S. Naga. 1992. Teori dan Soal Sistem Operasi Komputer,Jakarta: Gunadarma.

Operating System Concepts (6th or 7th Edition). Silberschatz, Galvin, Gagne, ISBN: 0-471-25060-0. Wiley

Silberschatz Galvin. 1995. 4 Edition Operating System Concepts: Addison Wesley.

Sri Kusumadewi. 2000. Sistem Operasi. Yogyakarta: J&J Learning.

Tanenbaum, A.1992. Modern Operating Systems.New York: Prentice Hall

Link and Sites:

http://www.ilmukomputer.com
http://vlsm.bebas.org
http://www.wikipedia.com