MODUL VI
CCS 210 SISTEM OPERASI

Judul	STRATEGI PENJADWALAN PROSES	
Penyusun	**Distribusi**	**Perkuliahan**
Nixon Erzed	**FASILKOM** **UNIVERSITAS ESA UNGGUL**	Pertemuan – VI ON LINE

Tujuan :

Mahasiswa memahami bagaimana proses terimplementasi dalam sistem operasi dan memahami strategi penjadwalan proses.

Materi:

- Implementasi Proses
  a. Penciptaan Proses
  b. Penghancuran Proses
  c. Pengalihan Proses
  d. Penjadwalan proses
- Strategi Penjadwalan Proses

Referensi :

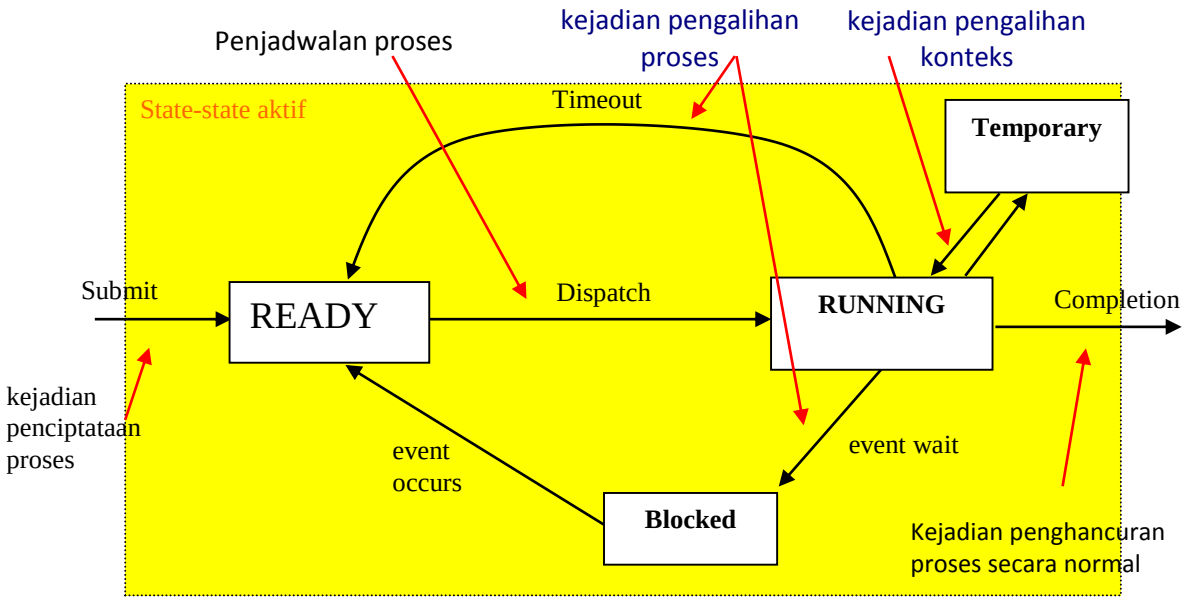
1. Modern Operating System 3th Edition Andrew S Tanembaun 2009
2. Operating System, Internals and design Principles, William Stallings 7th Ed. 2012
3. Operating System Concepts, Abraham Silberschatz, 9th Ed, 2012
4. Sistem Operasi, Bambang Haryanto, Rev.5 2012
5. Arsitektur dan Organisasi Komputer, William Stalling, Prehalindo

IMPLEMENTASI PROSES

Implementasi proses meliputi :

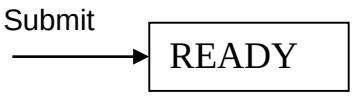
1. Penciptaan Proses
2. Penghancuran Proses
3. Pengalihan Proses
4. Penjadwalan proses

Perhatikan kembali diagram state berikut :



1. PENCIPTAAN PROSES

Penciptaan proses terjadi ketika sistem memberikan tanggapan terhadap datangnya permintaan pengeksekusian suatu program. Dalam diagram state dasar kejadian penciptaan proses adalah aksi event/transisi submit.



Aktivitas :

- Menamai/id proses
- menyisipkan id proses pada senarai proses atau tabel proses
  antara lain : menempatkan id proses dalam tabel antrian Ready
- menciptakan PCB
  Ketika proses diciptakan, sistem operasi membangun struktur data untuk mengelola dan alokasi ruang alamat proses tersebut.
- menentukan prioritas awal dan inisialisasi proses → store to PCB

- mengalokasikan sumber daya awal bagi proses (misal ruang memory yang dilanjutkan dengan loading program)

Pemicu penciptaan proses

1. Pada lingkungan batch system, sebagai tanggapan terhadap pemberian suatu kerja/job (datangnya waktu pengeksekusian tumpukan proses)
2. Permintaan log-on proses (submit) oleh user, pada lingkungan interaktif
3. Sebagai tanggapan terhadap permintaan layanan sistem operasi oleh suatu proses pemakai → rutin-rutin sistem operasi
4. Proses menciptakan proses anak (dalam pengertian umum)

2. PENGHANCURAN PROSES

Ketika sebuah proses menyelesaikan pengeksekusian intruksinya yang terakhir (completion), maka proses akan meninggalkan sistem dan harus membebaskan semua sumber daya yang digenggamnya. Kejadian ini disebut sebagai penghancuran proses.

Dalam pengertian yang lebih luas, terdapat dua keadaan proses yang harus dihancurkan, yaitu :

a. Penghancuran Normal
b. Penghancuran Tidak Normal

Normal :
Dalam keadaan yang normal penghancuran proses terjadi karena proses yang berada pada state running telah menyelesaikan semua aktivitasnya atau proses mencapai kejadian completion.

Tidak normal :
Selain proses completion, proses yang sedang aktif dapat mengalami kejadian penghancuran karena suatu sebab. Dalam kasus ini sistem operasi harus dapat membebaskan (dealokasi) sumber-sumber daya yang dikuasai proses tersebut.

Penyebab penghancuran tidak normal antara lain :

- run time error,
- penghentian paksa (abort atau end task) untuk state mana saja

Aktivitas :

- Sumber daya-sumber daya yang dipakai dikembalikan kepada pengelola proses
- proses dihancurkan dari senarai proses atau tabel-tabel sistem
- PCB dihapus

Sesuai dengan kebutuhan interleave, ketika sebuah proses dihancurkan maka sistem operasi segera mengaktifkan proses yang berada pada antrian terdepan di state ready (dispatch)

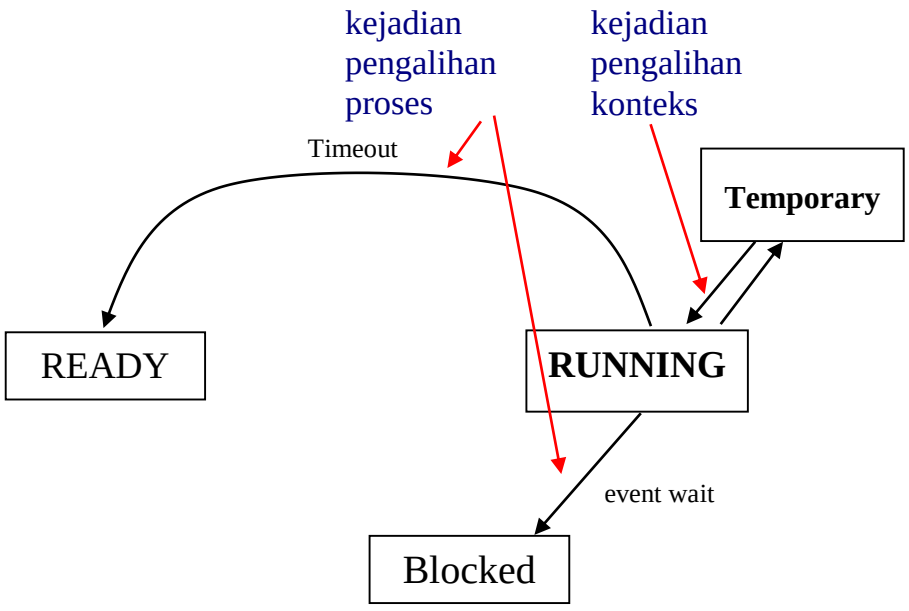
Kerumitan penghancuran proses dapat terjadi jika proses yang dihancurkan telah menciptakan proses lain (proses anak). Terdapat dua pendekatan yang dapat diterapkan :

- proses-proses anak dihancurkan ketika proses induk dihancurkan
- proses-proses anak dianggap independen terhadap proses induk, sehingga ketika proses induk dihancurkan maka proses anak tidak secara otomatis dihancurkan.

3. PENGALIHAN PROSES

Pengalihan proses adalah → pengalihan penguasaan prosesor dari suatu proses ke proses lainnya.

Pengalihan proses terjadi ketika sebuah proses yag sedang running, harus meninggalkan state running sebelum menyelesaikan prosesnya, dan sistem operasi menempatkan proses lain yang berada pada antrian terdepan di state ready pada state running, atau menempatkan rutin kernel (proses sistem operasi) yang menginterupsi pada state running.



Aktivitas :

- Pengendali menyimpan konteks proses yang running ke stack, dan membackup semua status sistem ke PCB dan fasilitas lainnya
- Menginisialisasi semua register/flag dengan data proses baru
- Pengendali men-set register pencacah program (PC) ke alamat awal program yang di*dispatch.*
- Perbarui PCB proses yang running dan yang kena pengalihan.

- Perbarui struktur data proses-proses (tabel-tabel proses)

Pengalihan proses dapat terjadi akibat :

A. Process Switching :

- proses kehabisan jatah waktu prosesnya sehingga terkena *time out*
- proses sedang menunggu kejadian untuk melengkapi jobnya sehingga kena blocked

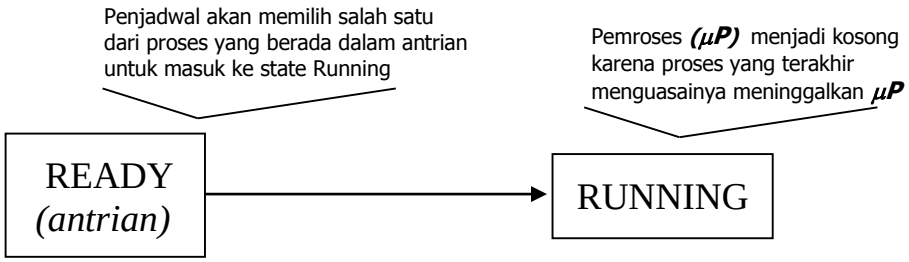
B. Context switching

- terjadi intrupsi oleh proses dengan kewenangan lebih tinggi.

Khususnya pengalihan proses yang terjadi karena interupsi, begitu proses yang mengiterupsi meninggalkan state running, maka proses tersebut segera kembali menjadi running tanpa harus melalui penjadwalan kembali. Pengalihan proses tersebut dikenal sebagai pengalihan konteks (context switching).

4. MENJADWALKAN PROSES

Penjadwalan dapat diartikan sebagai cara menentukan/memilih satu proses dari sekumpulan proses berstate ready untuk running ketika proses terakhir yang running melepaskan penguasaannya terhadap prosesor.



Dalam pengertian yang lebih luas penjadwalan meliputi :

- Penundaan proses (suspend a process)
  Me-non aktifkan sementara suatu proses untuk memperbaiki kinerja sistem. Proses non aktif akan di-swap out ke *extended area.*
- Pelanjutan kembali proses (resume a process)
  Memulihkan kembali proses yang non aktif. Resume sebuah proses dilakukan dengan cara men-swap in dari *extended area* kembali ke memory
- Pengubahan prioritas proses
- Mem-block proses

STRATEGI PENJADWALAN PROSES

1. PENGERTIAN PENJADWALAN

Pada implementasi multitasking, terdapat lebih dari 1 proses yang aktif dan membutuhkan layanan sistem. Sementara itu secara nyata, hanya satu proses yang dilayani oleh prosesor pada satu saat, dan proses yang lain akan berada dalam antrian ready. Saat prosesor ditinggalkan oleh proses yang running terakhir sekali, maka sistem operasi (kernel) harus memilih salah satu proses yang berada dalam antrian ready.

Penjadwalan dapat diartikan sebagai cara menentukan/memilih satu proses dari sekumpulan proses berstate ready untuk running ketika proses terakhir yang running melepaskan penguasaannya terhadap prosesor.

Dalam sistem operasi sebuah Penjadwal merupakan kumpulan kebijakan dan mekanisme di sistem operasi yang berkaitan dengan urutan kerja (urutan eksekusi proses-proses) yang diimplementasikan dalam sistem komputer.

Perhatikan data contoh berikut :

Id proses	Waktu Kedatangan (LT/ logon time)	Estimasi Waktu Proses (RTE/Run Time Estimation)
P_x	0	16
P_z	2	8
P_y	3	12
P_w	12	7
P_t	18	10

Pada waktu ke-0 (relatif terhadap pengamatan) terdapat hanya satu proses yang sudah datang, sehingga dengan mengasumsikan bahwa prosesor saat itu idle dan belum ada proses lain dalam antrian ready, maka proses Px akan segera menjadi running.

Jika Px kemudian diberikan jatah waktu menuntaskan eksekusi seluruh instruksi-nya, maka Px akan meninggalkan proses pada waktu ke-16. Pada saat Px running, didalam antrian ready berturut-turut datang proses Pz, Py, dan Pw.

Dalam pemahaman umum yang menggunakan logika FIFO, proses berikutnya yang akan running adalah Pz.

Namun cara penentuan/pemilihan proses yang akan running tidak harus selalu mengacu kepada waktu/urutan kedatangan, dapat juga mengacu pada suatu skala prioritas atau bisan juga berdasarkan ukuran proses. Begitu pula halnya dengan penetapan berapa lama prosesor akan dialokasikan pada proses yang running, juga dapat dibatasi.

Algroitma penjadwalan akan mengimplementasikan pola logika penentuan/ pemilihan proses dan penetapan berapa lama proses akan berjalan.

Atau dengan kalimat lain penjadwal adalah → implementasi kebijakan tentang urutan eksekusi proses-proses.

2. KRITERIA PENJADWALAN

Dalam perancangan algoritma penjadwalan proses terdapat beberapa sasaran yang mesti dioptimalkan, yaitu :

1. Adil (fairness)
2. Efisiensi
3. Response time → minimal
4. Turn Around Time → minimal
5. Throughput → maksimal

Adil (fairness)

Proses-proses diperlakukan sama dan mendapat jatah waktu pemroses. Tidak ada proses yang tidak kebagian waktu pemroses sehingga mengalami starvation (dikucilkan).

Pada penjadwalan yang didasarkan pada waktu kedatangan proses (FIFO), akan timbul masalah jika terdapat proses kecil (misalnya yang hanya memerlukan 5 siklus mesin) datang dibelakang proses besar yang membutuhkan 1 milyar siklus mesin, sehingga proses kecil tersebut terpaksa harus menunggu hingga proses besar yang ada didepannya menyelesaikan eksekusi seluruh rangkaian proses (1 milyar siklus mesin) hanya untuk mendapat alokasi 5 siklus mesin.

Efisiensi

Ukuran efisiensi adalah utilisasi pemroses. Sebuah algoritma penjadwalan harus mengupayakan perbaikan efisiensi relatif terhadap algoritma sebelumnya.

Utilitas → yang diartikan sebagai perbadingan/rasio waktu sibuk pemroses terhadap waktu aktif sistem.

→ Sehingga agar efisien, diupayakan pemroses selalu sibuk, atau meminimalkan delay antar proses (meningkatkan interleave).

$$\text{utilasi pemroses} \quad U = \frac{\text{waktu pemroses terpakai/sibuk melayani proses pemakai}}{\text{Waktu aktif}} \times 100\%$$

Response time

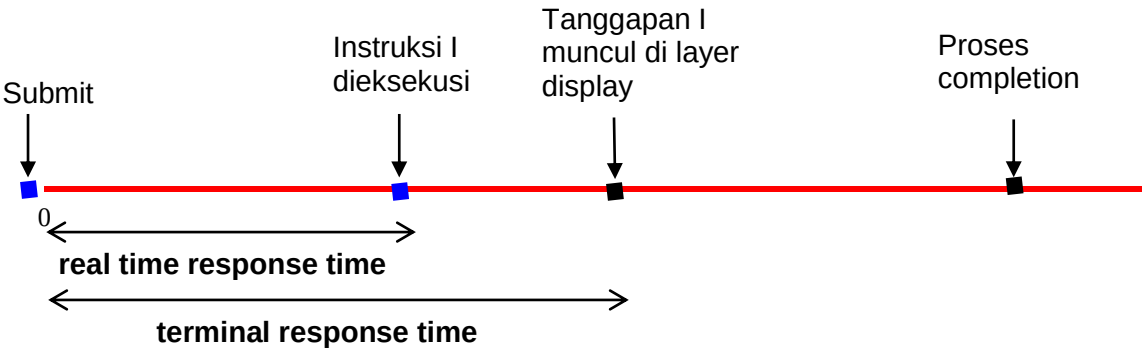
Response time adalah waktu menunggu yang diperlukan mulai dari ketika proses disubmit hingga mendapat layanan pemroses.

Terdapat dua macam response time :

- terminal response time (interactive system response time):
  waktu tanggap berdasarkan pandangan user, yang dihitung dari mulai proses disubmit hingga hasil pertama muncul dimonitor.
- event response time (Real Time Respons Time):

  waktu tanggap berdasarkan sistem waktu nyata, yaitu dihitung dari mulai proses disubmit hingga instruksi yang pertama dieksekusi

Perhatikan gambar berikut ini :

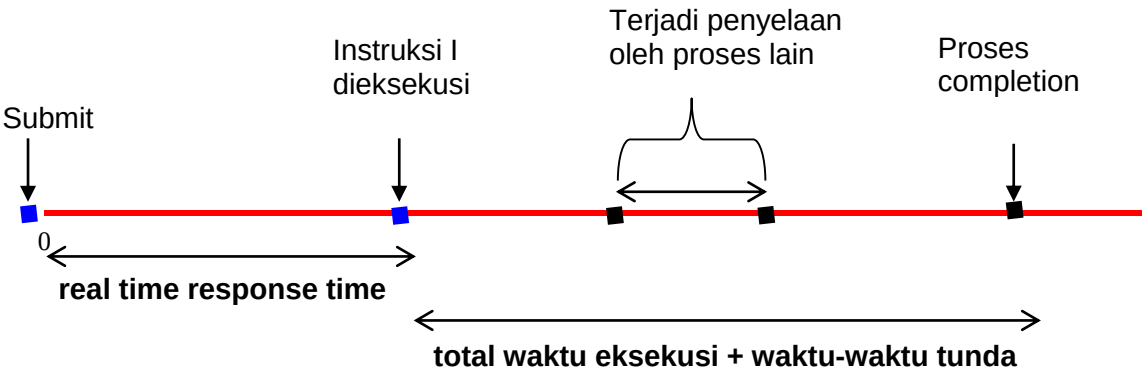


Catatan : *Terminal response time ≥ real time response time*

Turn Arround Time (TAT)

Turn arround time adalah waktu yang diperlukan mulai dari proses disubmit hingga eksekusi perintah yang terakhir diselesaikan (complention). Atau Turn Arround Time adalah jumlah waktu menunggu real time (RT response time), total waktu ekseskusi dan waktu-waktu tunda yang terjadi akibat proses disela/diinterupsi oleh proses lain.

$$TAT = RT\ response\ time + waktu\ eksekusi + waktu\ sela$$



Throughput

Throughput merupakan jumlah proses/thread yang dapat diselesaikan dalam waktu tertentu. Semakin banyak proses yang dapat diselesaikan dalam suatu rentang waktu berarti berkurang beban sistem.

→ Algoritma penjadwalan harus dapat memaksimumkan throughput.

Lima sasaran penjadwalan tersebut kadang memiliki dampak yang saling belawanan, sehingga mesti diatur sedemikian rupa agar menghasilkan kondisi yang paling optimal.

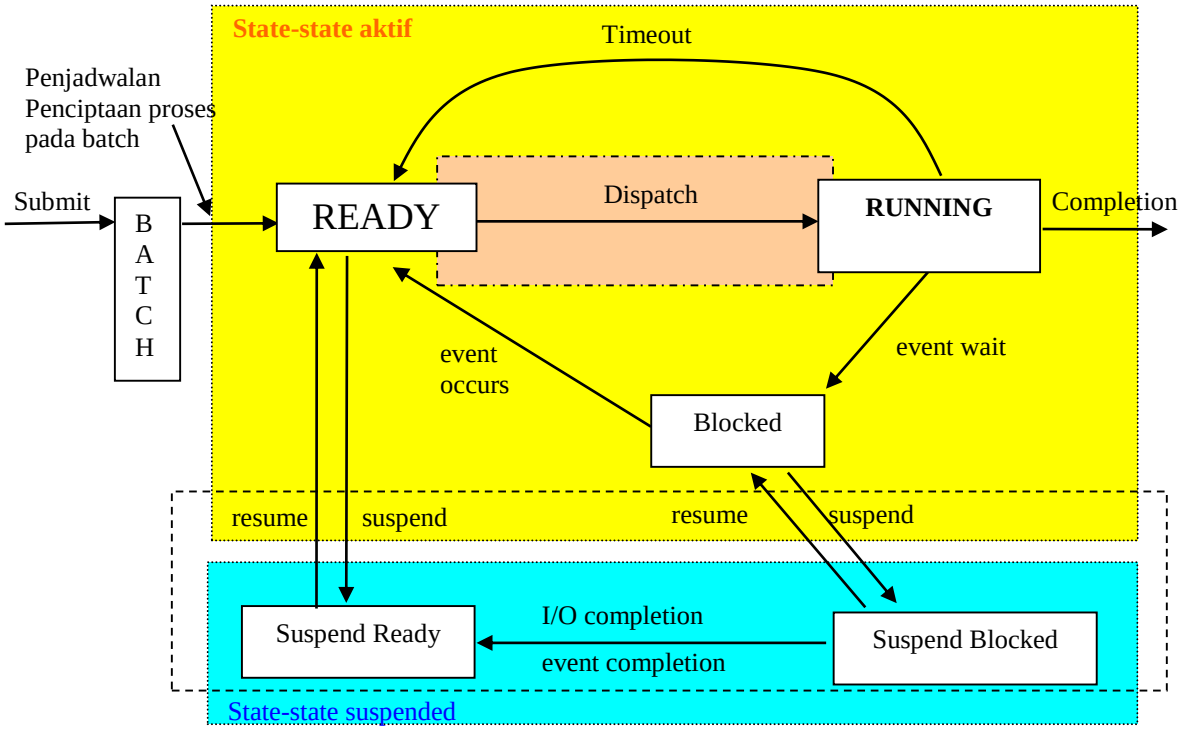
Misalnya untuk meningkatkan response time, maka dibuat kwanta waktu yang lebih kecil. Dari segi response time memendekkan kwanta waktu akan mempersingkat waktu tunggu, tapi dari sisi turn arround time memedekan kwanta waktu mengakibatkan bertambahnya waktu tunggu (sela) dan waktu eksekusi proses.

3. JENIS-JENIS PENJADWAL

Dalam suatu sistem operasi terdapat beberapa jenis penjadwal yang dikelompokkan berdasarkan seberapa dekat kejadian penjadwalan tersebut dengan state running. Jadi dalam hal ini, penjadwal sesungguhnya tidak hanya untuk menetapkan/memilih proses pada antrian ready yang akan running ketika prosesor ditinggalkan oleh proses yang terakhir menguasainya.

Pengelompokkan penjadwal berdasarkan area implementasinya adalah:

- **Penjadwal jangka pendek**
  Termasuk dalam penjadwal jangka pendek adalah penjadwal untuk memutuskan proses yang akan running ketika prosesor menjadi kosong → dispatcher. Penjadwal ini dimasukkan kedalam jenis penjadwal jangka pendek karena merupakan kejadian yang diimplementasikan pada area yang paling dekat dengan status running.
- **Penjadwal jangka menengah**
  Proses yang terkena suspended, dalam batasa waktu yang tertentu harus dipulihkan (resume) kembali. Penjadwal untuk memutuskan kapan suatu proses yang suspended akan dipulihkan kembali dikelompokkan sebagai penjadwal jangka menengah
- **Penjadwal jangka panjang**
  Penjadwal jangka pendek dan jangka menengah memiliki kepastian dalam implementasinya, karena penjadwal tersebut terjadi pada proses yang sudah diciptakan. Sedangkan penjadwal jangka panjang berlaku bagi kejadian yang tidak memiliki kepastian kapan akan dilakukan, yang meliputi :
  - Penjadwal pada *batch processing system* untuk memutuskan kapan suatu bakal-proses (dari tumpukan) akan dilaksanakan.
  - Penjadwal untuk mengintervensi modul penjadwal pada state Ready (penjadwal jangka pendek) katika terdapat proses yang terkucil



4. STRATEGI PENJADWALAN

Strategi penjadwalan berkaitan dengan diizinkan atau tidaknya interupsi terhadap proses yang sedang running. Terdapat dua strategi penjadwalan, yaitu :

1. Penjadwalan non preemptive
2. Penjadwalan preemptive

Dalam implementasi sistem operasi diterapkan gabungan/kombinasi strategi preemptive dan non preemptive

a. Penjadwalan non preemptive :

Saat proses diberikan jatah waktu pemroses maka pemroses tidak dapat diambil alih oleh proses lain sampai proses itu selesai.

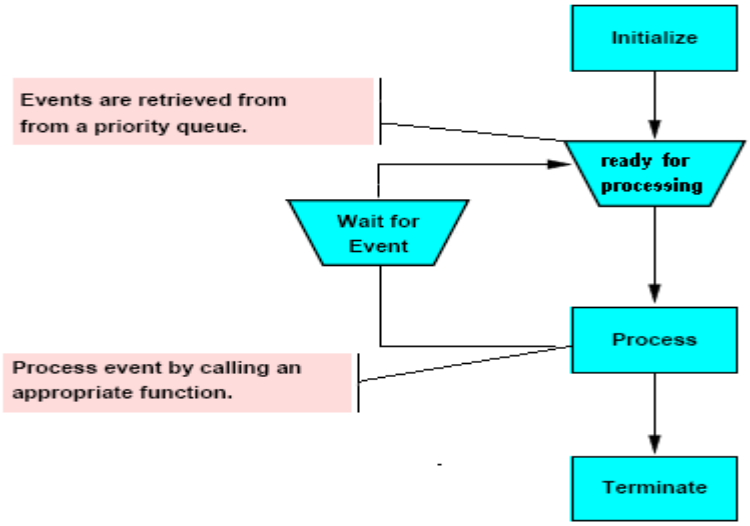
Berikut adalah sifat penjadwalan non preemptive

→ run to completion *(akan running sampai selesai)*

→ tidak mengenal time-out

→ kecuali untuk proses yang harus menunggu suatu event atau tersedianya suatu sumber daya (event wait → blocked).

Perhatikan gambar di bawah ini.

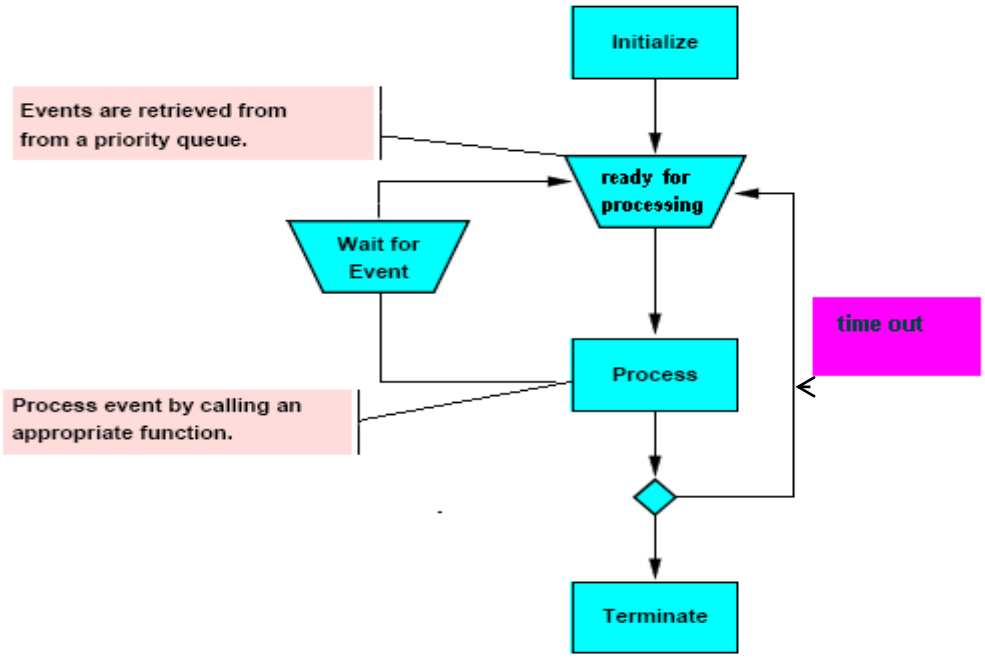


Proses yang mendapat alokasi waktu pemroses akan dikerjakan sampai tuntas, kecuali jika terdapat kejadian proses tersebut harus menunggu suatu event atau tersedianya sumber daya (proses Blocked) .

b. Penjadwalan preemptive

Saat proses diberi jatah waktu pemroses → pemroses dapat diambil alih proses lain → proses disela sebelum selesai dan harus dilanjutkan/menunggu jatah waktu pemroses tiba kembali pada proses itu → proses akan masuk state ready.

Penjadwalan preemptive berguna pada sistem dimana proses-proses memerlukan pemroses secara tanggapan cepat (real time).



c. Strategi kombinasi

Pada implementasi sistem operasi untuk *general purpose computer* sulit jika diterapkan hanya salah satu strategi saja, karena terdapat berbagai karakteristik proses yang mungkin memiliki prioritas yang khusus. Pendekatannya :

- pada kondisi umum menggunakan strategi non preemptive, pada keadaan tertentu mengizinkan preemptive
- pada kondisi umum menggunakan preemptive, tapi untuk proses dengan kriteria tertentu diberlakukan non preemptive.