# UTS

26 Oktober 2011
8:13

@Wisnu

**Overview SISOP :** ----------

- Sistem operasi adalah ;
  - Program yang mengatur eksekusi program aplikasi
  - Merupakan interface antara aplikasi dengan perangkat keras
  - Merupakan jembatan penghubung antara user dengan hardware PC

- Kemampuan yang harus dimiliki OS :
  - Sebagai interface user dengan komputer
  - Dapat mengatur resource komputer
  - Dapat dikembangkan atau ditambahkan fungsinya

- Mengapa OS berevolusi :
  - Supaya dapat memanfaatkan kemajuan teknologi
  - Ada penambahan layanan baru
  - Koreksi terhadap kesalahan

- Ciri2 OS yang baik :
  - Tersusun secara modular
  - Interface antara modul terdefinisi dengan baik dan sesederhana mungkin
  - Terdokumentasi dengan baik'

- Evolusi OS :
  - Serial processing
    - User harus mengakses mesing komputer langsung , job dilakukan secara satu per satu
    - Bukan merupakan OS
  - Simple batch System
    - Beberapa job yang akan dikerjakan dikumpulkan menjadi satu sebelum akan dieksekusi
    - Sama dengan uniprogramming
    - Terdapat mode :
      - User mode
        - Kodisi saat program user sedang dieksekusi oleh prosesor
      - Kernel mode
        - Kondisi saat program OS sedang dieksekusi prosesor
    - Kekurangan :
      - Terjadi overhead
        - Sebagian besar memori ditempati oleh OS

  - Multiprogrammed batch System
    - Job yang diproses lebih dari satu diproses bersamaan
    - Sama OS modern = multiprogramming
    - Contoh eksekusi
      - Program A dieksekusi saat program b menunggu event lain
    - Waktu eksekusi Multi 2x lebih cepat dari uni
  - Time-Sharing System
    - Satu komputer diakses oleh banyak orang
    - Multi user
    - Setiap program user selalu diletakkan pada alamat 5000

- Lima hal utama dalam pengembangan OS
  - Proses
  - Manajemen memori

- Proteksi keamanan
- Penjadualan dan manajemen resource
- Struktur OS

- Proses adalah :
  - Program yang dieksekusi/ sedang berjalan di komputer

- OS yang menekankan adanya timing dan sinkronisasi :
  - Multi
  - Time sharing
    - Banyak user mengakses banyak aplikasi
  - Real-time
    - Banyak user 1 aplikasi saja

- Permasalahan pada Proses :
  - Sinkronisasi tidak tepat
  - Kegagalan mutex
  - Eksekusi program tidak terkendali
  - Deadlock

- Management memori
  - Tujuan
    - Agar alokasi memori dapat dilakukan secara terkendali dan efisien
  - Tugas OS dalam menejemen memori
    - Mnegisolasi proses
    - Mengatur dan megalokasikan memori secara dinamis
    - Mendukung pemrograman modular

  - Memori Virtual
    - Terletak pada harddisk
  - Paging
    - Menempatkan program ke memori
    - Nomor page dan offset

- Proteksi dan Keamanan informasi
  - Jaminan informasi selalu tersedia
  - Kerahasiaan informasi
  - Keutuhan data
  - Keaslian data

- Jenis OS modern
  - Monolitic kernel
  - Arsitektur microkernel
  - Multithreading
  - Symmetric multiprocessing (SMP)
  - OS terdistribusi
  - OS model Objek oriented

  - Monolitic kernel
    - OS diimplementasikan sebagain sebuah proses besar
  - Arsitektur microkernel
    - Kernel
      - Penjadwalan dasar
    - Server
      - Pada user mode sama seperti program aplikasi
    - Kelebihan
      - Implementasi lebih sederhana
      - Lebih fleksibel
      - Sangat sesuai dengan lingkukan terdistribusi
  - Multithreading

        - Proses yang sedang dieksekusi dipecah2 menjadi bagian kecil yang berjalan secara konkuren
        - Tread
            - Pecahan dari proses
        - Kelebihan
            - Modularity aplikasi lebih terkontrol
            - Respon aplikasi terhadap suatu event lebih terjamin
            - Overhead lebih kecil
    - Symmetric multiprocessing (SMP)
        - OS yang dijalankan pada komputer dengan lebih dari 1 prosesor
        - Kelebihan
            - Performansi lebih baik
            - Lebih handal
            - dll
    - OS terdistribusi
        - Sejumlah komputer terhubung melalui jaringan membentuk sebuah cluster
    - OS model Objek oriented
        - Modul yang ditambahkan ke kernel dirancang dengan metode OO

- Review OS
    - Windows
        - Hardware Abstraction Layer (HAL)
            - Memungkinkan OS diinstal pada Hardware yang berbeda
    - UNIX

**Deskripsi dan Kontrol Proses :** ----------

- Tugas OS terhadap proses
    - Mengakses banyak proses secara interleave (selang seling)
    - Menyediakan resource bagi tiap proses
    - Mendukung komunikasi antar proses

- Elemen pada proses yang sedang running (ada dalam PCB)
    - Indetifier
        - Identitas unik untuk membedakan suatu proses dengan proses yang lainnya
    - State
        - Status kondisi (ready,running,dll)
    - Prority
        - prioritas
    - Program counter
        - Alamat intruksi berikutnya yang akan dieksekusi
    - Memory Pointer
        - Pointer memori yang menunjuk pada alamat memori kode program dan data yang berhubungan dengan proses
    - Context data
        - Data yang terdapat pada register prosesor
    - I/O status information
    - Accounting information

- Pocess Control Block (PCB)
    - Struktur data yang menyimpan element2 proses
    - Proses = PCB + kode + data

- Status Proses
    - Trace proses
        - Adalah daftar urutan alamat memori sutu proses yang telah dieksekusi
    - Program dispatcher
        - Bbagian dari sistem operasi yang mengatur giliran pemanfaatan prosesor

- Pembentukan proses

    - Dibentuki oleh OS
    - Penyebab terbentuknya proses baru
        - Masuk job baru
        - Log on user
        - Dibentuk oleh proses lain
    - Penyebab terminasi proses
        - Telah selesai secara norma;
        - Time out
        - Memori tidak tersedia

- Model Proses
    - 2 status
        - Running
        - Not running
    - 5 status
        - New
        - Ready
        - Running
        - Blocked
            - Berhenti karena menunggu even lain
        - Exit
    - 6 status
        - 5
        - Suspend -> HD
            - Memori terbatas
    - 7 status
        - 5
        - Ready-suspend
            - Event yang ditunggu akan datang namun masih belom terdapat memori atau ada prioritas yang lebih besar masuk
        - Blocked-suspend
            - Memori kurang dibawa ke HD

- Struktur kontrol sistem Operasi
    - Informasi setiap resource disimpan ke dalam struktur tabel yang terdiri sbb:
        - Tabel memori
            - Digunakan untuk menyimpan data pengunan memori real maupun virtual
        - Tabel I/O
            - Digunakan untuk menyimpan data pengguna I/O
        - Tabel file
            - Info file
        - Tabel proses
            - Info proses

- Struktur kontrol Proses

    - Process Control Block (PCB) terdiri dari
        - Process Identification (PID)
            - Merupakan identitas numerik
        - Processor State Information (PSI)
            - Informasi status yang ada di prosesor
        - Process Control Information (PCI)
            - Informasi yang digunak untuk penjadwalan oleh OS

- Kontrol Proses

    - Model Eksekusi Proses
        - User mode
        - Kernel mode
    - Pembentukan proses

        - Beri identitas unik
        - Alokasi ruang memori
        - Inisialisai PCB
        - Set linkages yang sesuai
        - Bentuk atau perluas struktur data yang lain

    - Switching Proses
        - Hal yang memicu terjadinya pergantian proses
            - Interrupt
                - Pergantian proses disebabkan adanya event eksternal dan tidak ada hubungannya dengan proses yang sedang running
                - Contoh : I/O interrupt, memory fault
            - Trap
                - Pergantian proses terjadi karena kesalah yang ditimbulkan oleh proses yang sedang dieksekusi
            - Supervisor call
                - Pergantian proses disebabkan oleh proses itu sendiri
                - Misal : minta operasi I/O

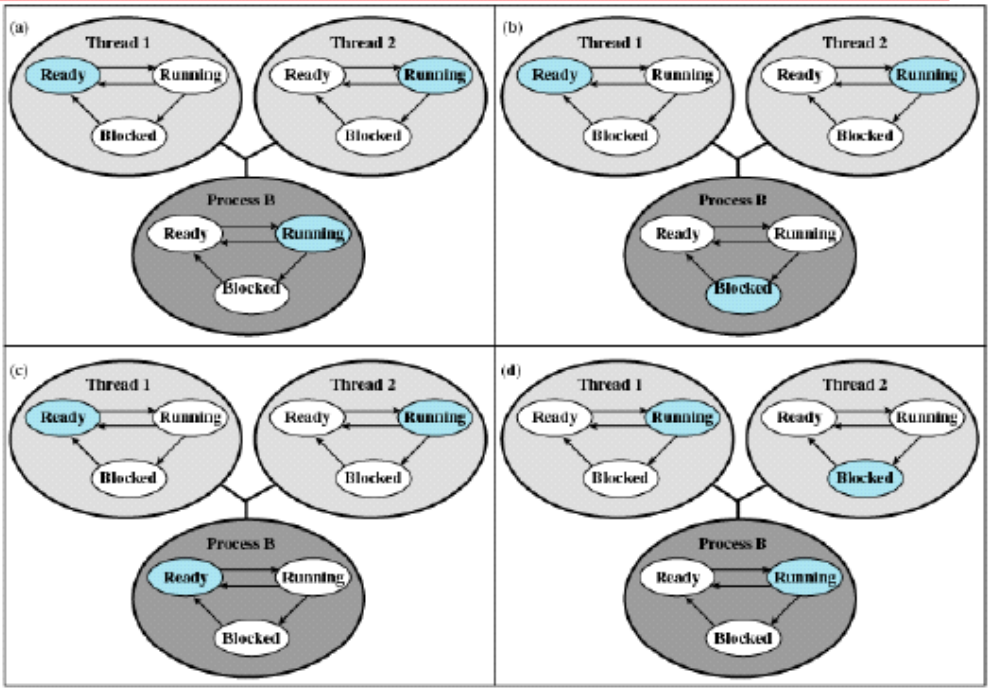
- Status Pada Proses UNIX
    - User Running
        - Dalam mode user
    - Kernel Running
        - Dalam mode kernel
    - Ready to Run, in memory
        - Dapam status ready
    - Asleep in memory
        - Seperti status Blocked
    - Ready-Swapped
        - Seperti ready suspend
    - Sleeping, swapped
        - Seperti block suspend
    - Preempted
        - Prose ingin berubah mode
    - Created
        - Proses baru saja dibuat dan belom siap running
    - Zombie
        - Proses sudah selesai, namun proses tersebut meninggalkan record untuk proses lainnya.

**Tread, SMP, Microkernel :** ----------

- Prose dan Thread
    - OS traditional
        - Satu proses satu thread
            - MS-DOS
        - OS-Modern
            - Banyak proses dan tiap proses terdiri dari bnyak thread
    - Multithreading
        - Merupakan kemampuan OS dalam mengeksekusi banyak Thread yang berasal dari sebuah proses
    - Kelebihan Thread
        - Pembentukan thread jauh lebih cepat dari pada proses
        - Terminasinya lebih cepat
        - Perpindahan eksekusi anatar thread lebih cepat
        - Komunikasi antar thread tanpa melibatkan kernel

    - Pengaturan Thread
        - Level thread
            - Eksekusi tiap thread diatur dalam penjadawalan

        - Level proses
            - Jika proses suspend maka semua threadnya suspend juga
            - Jika proses diterminasi maka semua thread juga diterminasi
- Jenis Thread
    - User Level Thread (ULT)
        - Semua manajemen thread dilakukan oleh program aplikasi
        - Tidak ada kode program untuk mengatir thread pada program kernel
        - Contoh :
            - POSIX, Solaris 2
        - Contoh hubungan status ULT




Screen clipping taken: 26/10/2011 11:06

        - Kelebihan ULT dari KLT :
            - Pergantian thread tidak melibatkan kernel, sehingga overhead akibat perubahan 2 mode bisa dihindari
            - Algoritma penjadwalan bisa dibua berbeda pada tiap aplikasi
            - Dapat dijalankan pada OS beda asal sama ULT
        - Kekurangan :
            - Jika ada thread yang melakukan System call akan menyebabkan thread yg lain terhenti
            - Penggunaan multiprosesor tidak bisa dimaanfaatka secara maksimal
    - Kernel Level Thread (KLT)
        - Semua manajemen thread dilakukan oleh thread
        - Tidak ada kode program untuk mengantur thread pada program aplikasi
        - Contoh OS : windows NT, 2000
        - Kelebihan KLT dari ULT ;
            - Thread dari satu proses dapat dieksekusi di prosesor lain
            - Thread yang melakukan service call tidak menyebabkan proses terblock
            - Routine kernel dapat terdiri dari banyak thread
        - Kekurangan KLT
            - Dapat menyebabkan overhead
            - Algoritma penjadwalan tidak fleksibel
            - Tidak dapat dijalankan pada OS yang beda

    - Kombinasi ULT dan KLT
        - Tentu lebih bagus

- Sysmmetric Multriprocessing (SMP)
    - Arsitektur Master/Slave

        - Kernel sistem selalu dijalankan pada sastu prosesor
        - Proses master bertanggung jawab mengatur proses thread
        - Kelebihan :
            - Implementasi sederhana
            - Rebutan resource dapat diatasi
        - Kerugian
            - Bila prosesor master gagal, maka seluruh sistem gagal
            - Dapat terjadi bottle neck jika prosesor master kelebihan beban

    - SMP
        - Kernel sistem dapat dijalankan pada sembarang prosesor
        - Kelebihan :
            - Sistem masih dapat berjalan meski ada prosesor yang gagal;
            - Bottle neck dapat dihindari
        - Kerugian
            - Implementasi komplek
            - Dapat terjadi rebutan resource

    - Microkernel
        - Microkernel adalah core/inti OS yang berukuran kecil yang hanya berisi fungsi2 utama
        - Dibuat karena model OS sudah semakin besar dan semakin kompleks

        - Arsitektur Microkernel
            - Model lapis horisontal
            - Merupakan arsitektur client/server
            - Tugas microkernel
                - Mengatur pertukaran pesan
                - Menjamin keamanan pesan

        - Kelebihan Microkernel
            - Interface seragam
                - Tidak membedakan layanan user-level dengan kernel-level
                - Semua layanan dilakukan dengan message passing
            - Dapat dikembangkan
            - Fleksibel
            - Portable
            - Handal
            - Dapat digunakan pada sistem terdistribusi
            - Termasuk OS OO

        - Performansi Microkernel
            - Kekurangan
                - Waktu membangun dan mengirim pesan relatif lebih besar daripada service call
            - Modifikasi pda microkernel
                - Ukuran MK yang semakin kecil

        - Perancangan Microkernel
            - Fungsi dalam Microkernel berhubungan langsung dengan hardware
            - Fungsi yang harus ada dalam Microkernel
                - Manajemen low level meory
                - Komnikasi antar proses
                - Manajemen I/O dan interrupt

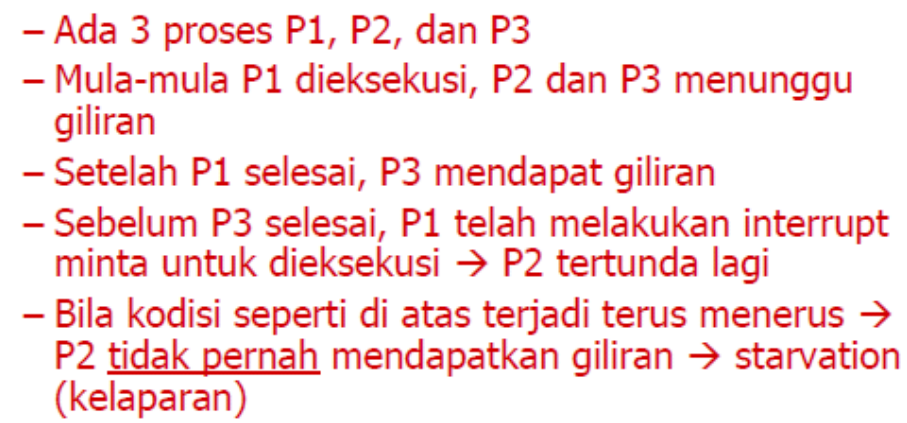
**Mutual Exclusion dan Sinkronisasi : ----------**

- Ruang lingkup concurrency
    - Komunikasi antar proses
    - Sharing dan kompetisi penggunaan resource
    - Singkronisasi antar berbagai proses

    - Pengalokasian waktu prosesor untuk setiap proses
    - Dimana concurrancy diperlukan :
        - Multiprograming
            - Banyak proses - satu prosesor
        - Multiprocesing
            - Banyak proses - bnyak prosesor
        - Distributed procesing
            - Bnayak proses banyak prosesor
- Istilah - istilah yang berkaitan dengan concurranancy
    - Critical section
        - Resource yang dalam satu waktu hnya dapat diakses oleh satu proses saja
        - Contoh l : printer, file, dll
    - Deadlock
        - Keadaan dimana 2 proses atau lebih tidak dapat meneruskan eksekusi akibat saling menunggu aksi/data
    - Livelock
        - Keadaan dimana dua proses atau lebih saling mengubah status sebagai respon terhadap perubahan status proses lain
    - Mutual exclusion
        - Syarat atau kondisi yang harus dipenuhi untuk mencegah terjadinya pengaksesan critical section oleh lebih dari satu proses dama satu saat.
    - Race condition
        - Keadaan dimana terdaoat banyak thread atau proses mengakses data bersamaan yang menyebabkan hasil akir sulit dipastikan
    - Starvation
        - Keaddan diman suatu proses yang siap dieksekusi terus menerus tidak diberikan kesempatan untuk melakukan aksinya

- Contoh kasus perlunya concurrency
    - Sebuah prosedure digunakan 2 buah proses dalam iniprosesor

- Jenis interaksi antar proses
    - Kompetisi antar proses
        - Mutex
            - 2 buah proses ingin mengakses printer
        - Deadlock
            - Dua buah proses P1 dan P2 sama2 mmbutuhkan resource R1 dan R2
            - Os telah memberikan R2 pada P1 dan R1 pada P2
            - Namun P1 dan P2 sama2 tidak mau melepas resource yang sedang digunakan.
        - Starvation




Screen clipping taken: 26/10/2011 12:43

- Mekanisme mutual exclusion
    - Syarat MUTEX

        - Dalam satu waktu hanya satu proses saja yang dapat mengakases critical section
        - Proses yang berjalan diluar critical section harus dapat melakukan aktifitas lain yang tidak menggangu proses lain
        - Tidak boleh terjadi deadlock dan stravation
        - Dalam mengakses critical section tidak bole ada delay bila tidak ada yang sedang mengakses critical section tersebut
        - Tidak boleh terjadi reace condition
        - Sebuah proses ada dalam critical section dalam waktu yang terbatas
- Implementasi mutual exclusion
    - Mutex dengan Enable-disable interrupt
    - Mutex dengan intruksi atomik
        - Intruksi test dan set
            - Memeriksa nilai satu variable dan mengubah nilainya
        - Intruksi exchange
            - Menukarkan data dari variable memory ke variable register dan sebaliknya
        - Kelebihan :
            - Dapat diterapkan pada sistem dengan jumlah proses berbda2
            - Bisa di unipro maupun multipro
            - Sederhana
        - Kekurangan
            - Dapat terjadi busy-wait
            - Dapat terjadi statvation
            - Dapat terjadi deadlock

- Semaphore
    - Mengunakan variable yang disebut semaphore yang digunakan sebagai tanda
    - Prosedure yang digunakan
        - semSignal(s)
            - Untuk mengirim signal semaphore
            - Increment
        - semWait(s)
            - Untuk menerima signal semaphore
            - Test
    - Primitif semaphore
        - General semaphore
        - Counting semaphore

    - Semaphore Primitif
        - Ketentuan
            - Inisialisasi variable semaphore tidak boleh negatif
            - Prosedure semWait
                - Akan mengurangi niali variable semaphore
                - Jika nilai variable negatif
                    - Proses yang mengeksekusi semwait akan diblok
                - Jika tidak
                    - Proses tersebut akan dilayani
            - Prosedur semSignal
                - Akan menambah nilai variable semaphore
                - Jika niali variable menjadi <=0
                    - Sebuah proses yang diblok oleh semWait akan dibebaskan
            - Pengaruh varible s.
                - s>=0
                    - Merupakan jumlah proses yang dapat mengeksekusi semWait tanpa penundaan
                - s<0
                    - Repakan jumlah proses yang diblok dan berada di dalam antrian

    - Semaphore Biner
        - Ketentuan
            - Inisialisasi variable semaphore hanya bernilai 0 dan 1

        - Prosedure semWaitB
            - Akan memeriksa nilai variable semaphore
            - Jika nilai variable = 1 maka idubah menjadi 0
            - Jika 0 maka proses tersebut di-block dan dimasukkan ke dalam antrian
        - Prosedure semSignalB
            - Akan memeriksa jumlah proses dalam antrian dengan fungsi is_emty()
            - Jika tidak ada proses dalam antrian maka nilai variable akan menjadi 1
            - Jika ada proses maka
                - Sebuah proses dipindah dari antrian ke status ready
                - Nilai variable tetap 0

    - Strong dan weak semaphore
        - Strong s
            - Adalah semaphore yang menentukan urutan proses yang akan dikeluarkan dari antrian
            - Dapat mencegah terjadinya stravation
        - Weak semaphore
            - Semaphore yang tidak menentukan urautan proses
            - Dapat terjadi starvation

    - Kelebihan semaphore
        - Dapat digunakan untuk membentuk mutex
        - Merupakan tool yang serba guna
    - Kekurangan semaphore
        - Tidak mudah membuat program dengan semaphore
        - Penangan mutes dan sinkronisasi sepenuhnya menjadi tanggung jawa programer

- Monitor
    - Monitor adalah Bagian bahasa pemrogramaan yang mempunyai fungsional seperti semaphore tetapi lebih mudah pengontrolannya
    - Karakteristik monitor
        - Variable lokal hanya dapat diakses oleh prosedur yang ada didalam modul monitor
        - Sebuah proses dapat masuk ke dalam monitor dengan cara minta salah satu prosedure yang ada di monitor
        - Dalam satu saat hanya satu proses yang dapat diproses dimonitor
        - Data variable global dapat dilindungi bila ditaruh di dalam monitor.
    - Fungsi untuk sinkronisasi
        - Cwait(c)
            - Akan menunda eksekusi proses yang memanggil prosedur di dalam monitor sampai variable kondisi c terpenuhi
        - Csignal(c)
            - Akan mengaktifkan eksekusi proses yang tertunda oleh fungsi cwait(c) dengan mengirimkan signal variable kondisi c
            - Signal ini akan hilang sendirinya jika tidak ada program yg membutuhkan
    - Mekanisme monitor
        - Dalam satu saat hanya satu proses saja
        - Proses yg sedang aktif dimonitor dapat terblok jika memenuhi kondeisi tertentu
        - Proses akan keluar dari monitor setelah menjalankan fungsi csignal
        - Jika sampai akhir prosedure fungsi csignal tidak tereksekusi makan proses tersebut di-blok ke antrian urgent, sehingga monitor dapat digunakna oleh proses lain
    - Kelebihan monitor
        - Dapat menangani sinkronisasi ( tidak perlu melibatkan programmer)
        - Pengecekan masalh yg berhubungan dengan mutex dapat terpusat hanya pada modul monitor, tidak tersebar diberbagai lokasi
        - Sekali program minitor telah benar makan akses terhadap critical resource oleh berbagai proses akan selalu benar
    - Kelemahan
        - Bila signal csignal terus hilang  maka proses yang ada dalam antrian akan selalu terblok
        - Proses yang megeluarkan csignal harus segera keluar

        - Jika proses tersebut terblok diperlukan w tahapan switching tambahan.
        - Memerlukakn mekanisme penjadualan proses yang harus benar2 handal
- Message passing
    - Adalah fasilitas untuk mendukung singkronisasi dan komuniksi antar proses
    - Dimana digunakan?
        - Sistem terdistribusi
        - Shared memory pada uni/multiprocesor
    - Primitif yg digunakan
        - Send(destination, message)
            - Untuk mengirim informasi dalam bentuk pesan ke proses tujuan
        - Receive(source, message)
            - Untuk menerima informasi dalam bentuk message dari proses asal
    - Sinkronisasi
        - Sinkronisasi diperlukan untuk mengatur komunikasi antar proses
        - Apa yang mungkin terjadi bila suatu proses mengekseskusi send()?
            - Dapat ter-blok hingga pesan telah diterima (blocking send) atau
            - Proses dapat melakkan proses yang lainc (nonblocking send)
        - Prose mengeksekusi receive()
            - Bila pesa telah dikirim oleh proses lain
                - Terima pesan dan lanjutkan eksekusi
            - Bila pesan belom dikirim
                - Proses terblock hingga pesan ada (blocking receive)
                - Lanjut eksekusi, abaikan pesan (nonblocking receive)
        - Model sinkronisasi yang dapat digunakan
            - Blocking send, blocking receive
            - Nonblocking sen, blocking receive
            - Nonblocking send, nonblocking receive
- Beberapa contoh kasus

**Konsep manajemen memori :** ----------

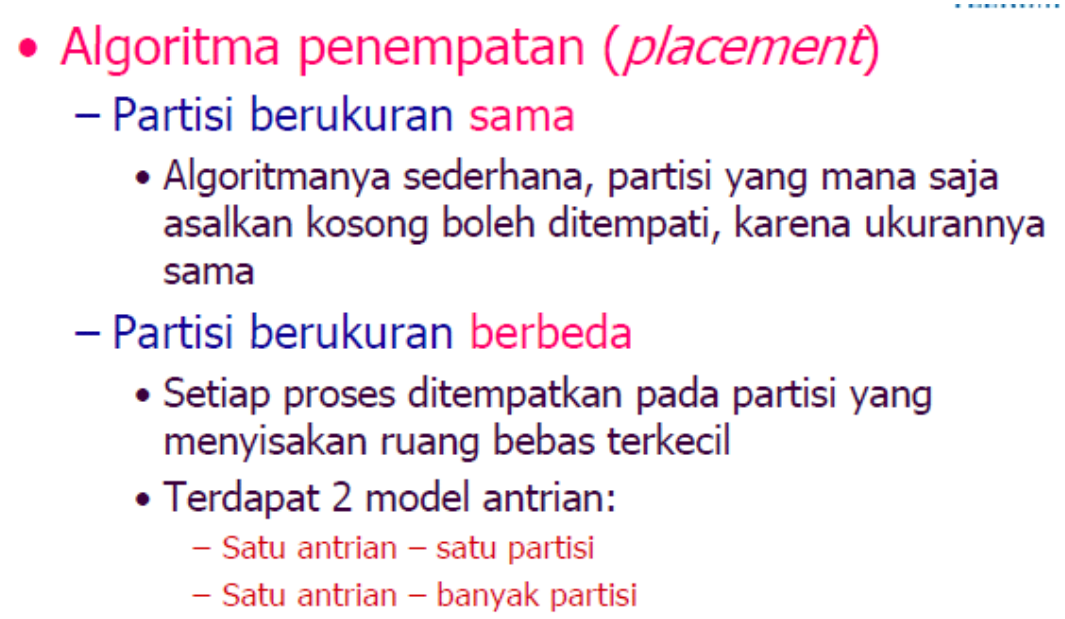
- Manajemen Memori
    - Dialkukan dengan cara membagi-bagi memori untuk mengakomodasi banyak proses
    - Untuk menjamin agar setiap proses yang ready dapat segera memanfaatkan procesor time
- Terdapat 5 requermnet manajemen memori
    - Relocation
        - Alamat harus yang akan ditempati harus ditemukan
        - Jenis2 alamat
            - Alamat logical
            - Alamat relatif
            - Alamat fisik
        - Jenis2 register
            - Base register
                - Alamat awal suatu proses
            - Bound register
                - Alamat akhir suatu proses
    - Protection
    - Sharing
    - Logical organization
    - Physical organization

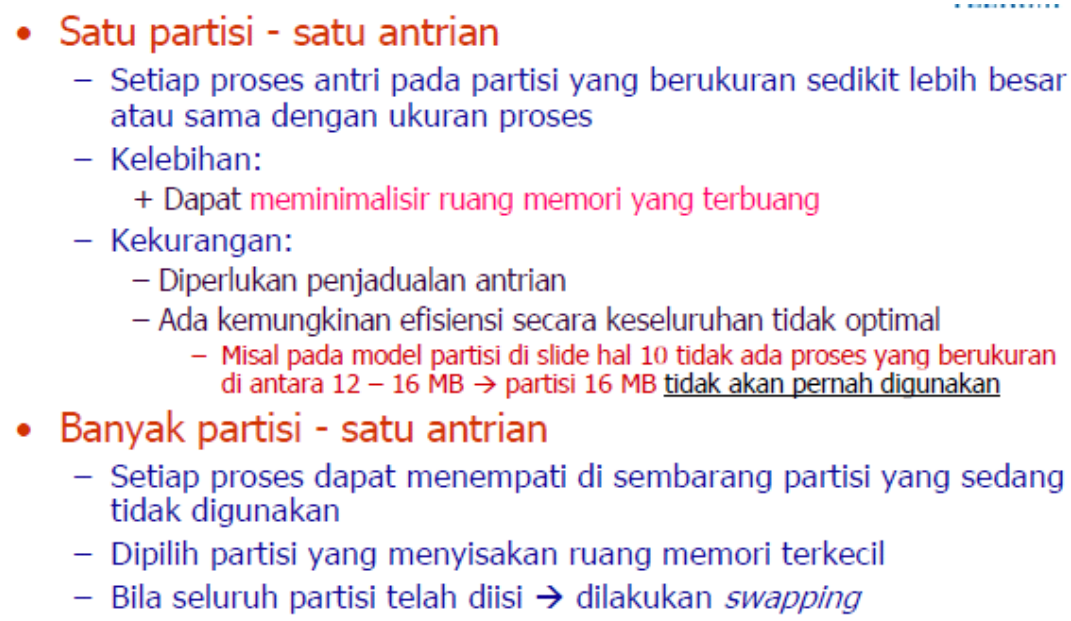
- **Teknik Manajemen Memori**
    - Partisi
        - Partisi tetap (fixed)
            - Partisi berukuran sama
            - Ukuran partisi berbeda2
        - Partisi dinamis
    - Paging sederhana

- Segmentasi sederhana
- Virtual memory
  - Virtual memory paging
  - Virtual memory segmentation

- Penjelasan :-)

  - **Partisi**
    - **Partisi tetap (fixed)**
      - Sebelum digunakan, memori dipartisi dulu
      - Partisi berukuran sama
        - Setiap proses yang ukurannya <= partisi dapat menempati partisi tsb.
        - Jika partisi full, maka os akan melakukan swap terhadap proses yang tidak aktif
        - Penggunaan memori tidak effesian
          - Misal memori 8mb, program 3mb, maka adakan ada 6mb memori yg terbuang
      - Ukuran partisi berbeda2
        - Lebih baik dari berukuran sama
          - Penggunaan memori lebih efesien
          - Tidak perlu overlay
      - Algoritma penempatan





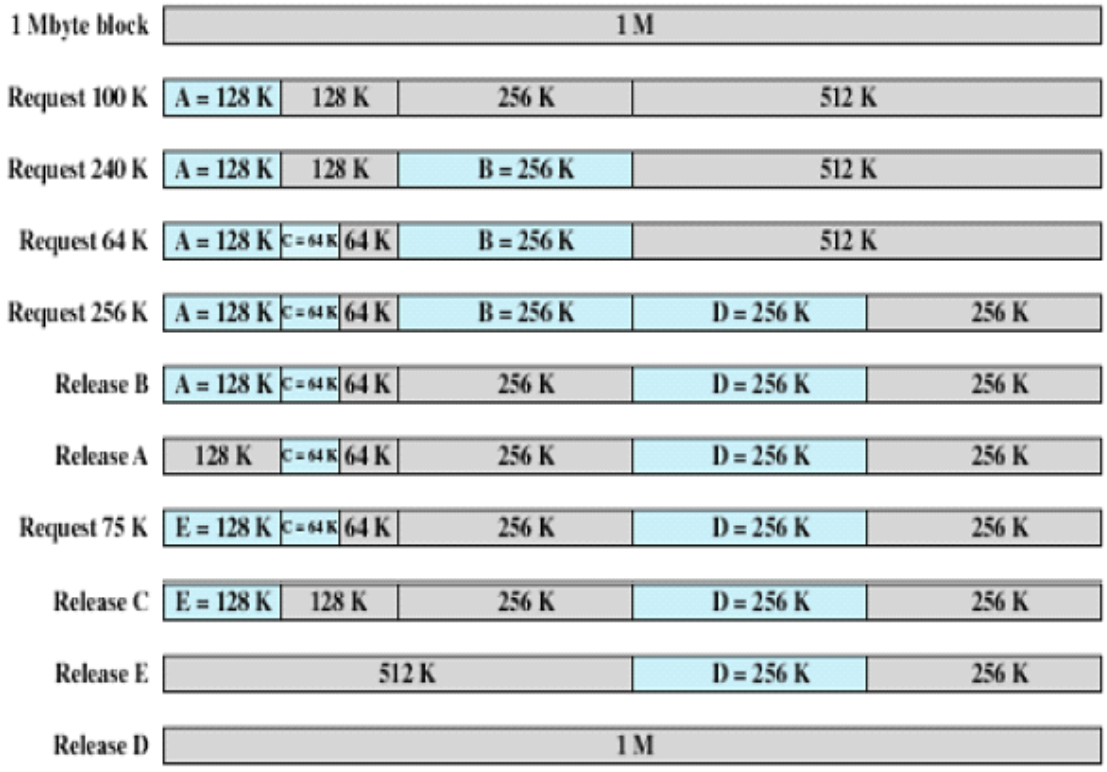

Screen clipping taken: 26/10/2011 16:01

- Kelebihan :
  - Mudah diimplementasikan
  - Overhead OS hanya sedikit
- Kekurangan
  - Tidak efesian dalam penggunaan memori
  - Dapat terjadi fragmentasi internal : sisa yang terjadi jika program < partisi

- **Partisi dinamis**
  - Jumlah dan ukuran memori tidak tetap
  - Ukuran partisi sama dengan ukuran proses yang akan menempatinya -> tidak terjadi fragmentasi internal
  - Dapat terjadi fragmentasi eksternal : sisa ruang yang terjadi jika ukuran proses lebih kecil dari ruang memori yang disediakan
  - **Algoritma penempatan**
    - **Best-fit**
      - Memilih block memori yang paling sedikit menyisakan ruang memori
      - Performasi jelek
        - Pecarian tempat yg lama
    - **First-fit**
      - Mencari blok memori kosong dimulai dari awal dan yg dipilh adalah yang pertama ukurannya sesuai
      - Algoritma Paling cepat
    - **Next-fit**
      - Pencarian memori kosong dimulai dari lokasi penempatan memori terakir
      - Lebih jelek dari first-fit
  - **Kelebihan :**
    - Tidak terjadi fragmentasi internal
    - Penggunaan memori lebih efisien
    - Jumlah proses aktif lebih fleksible
  - Kekurangan
    - Implementsi lebih susah
    - Dapat terjadi fragmentasi eksternal
    - Terjadi overhead penggunaan procesor

- **Buddy System**
  - Merupakan gabungandari teknik partisi tetap dengan partisi dinamis
  - Ruang memory yang tersedia selalu berukuran 2^k

- Ukuran program yang akan menempati partisi memori minimal setengah dari partisi.
- Contoh :




Screen clipping taken: 26/10/2011 16:23

- **Paging sederhana**
  - Dilakukan dengan cara
    - Membagi2 memori menjadi bagian2 kecil yang bersifat tetap dan ukuranya sama (mirip partisi tetap) dan selanjutnya disebut dengan frame
    - Membagi2 proses menjadi bagian2 kecil yang ukurannya sama dengan bagian2 memori yang selanjutnya disebuat dengan page
  - OS mengunakan page table untuk mencatat alokasi memori
  - Setiap prose mempunyai table sendiri
  - Page table = komor page + nomor frame
  - Perbedaan antara pagging sederhana dengan partisi tetap
    - Ukuran partisi lebih kecil
    - Program boleh menempati lebih dari satu partisi
    - Letak program dalam memori boleh tidak berurutan
    - Fragmentasi internal yang terjadi lebih kecil

  - Alamat :
    - Alamat lojik = nomor page + offset
    - Satu offset = satu alamat
    - Cara mentranslasikan alamat relatif atau alamat lojik ke alamat absolut :
      - Pisahkan bit2 nomor page
      - Gunakn nomor page tersebut sebagai indeks untuk mengetahui nomor frame k pada page table
      - Alamat awal dari alamat fisik adalah k x 2^m
      - Alamat fisik dapat diperoleh dengan cara menambahkan alamat awal dengan offset
      - Cara lain tambahkan bit2 nomor frame dengan bit2 offset
    - Contoh :

- Contoh *paging* sederhana:
  - Sebuah memori menggunakan pengalamatan 16 bit dan dipartisi dengan model *paging* sederhana dimana ukuran setiap *page* adalah 1 kB (1024 byte). Berapakah alamat absolut untuk alamat relatif 1502 atau alamat lojik dengan *page#* = 1 dan *offset* = 478 dimana page tersebut ditaruh pada frame nomor 6 ?
  - Jawaban:
    - Ukuran 1 *page* = 1024 byte → diperlukan 10 bit
    - Ukuran *offset* maksimum adalah sebesar ukuran satu *page (1024)*
    - Maka:
      - Jumlah bit untuk **page** = jumlah bit untuk *offset* = 10 bit
      - Jumlah bit untuk **nomor page** adalah 16-10 = 6 bit → jumlah *page* maksimum = $2^6$ = 64 *page* masing-masing berukuran 1 KB
      - Berapa kapasitas memori maksimum yang tersedia ???
    - Alamat relatif 1502 kalau ditulis dalam biner adalah 0000010111011110
    - Alamat tersebut sama dengan alamat lojik sbb:
      - Nomor *page#* = 1 (000001)
      - Nilai *offset* = 478 (0111011110)
    - Jika nomor *frame* untuk *page#* 1 (000001) pada *page table* adalah 6 (000110), maka:
      - Alamat fisik dari alamat relatif 1502 adalah 000110 (nomor frame) digabung dengan 0111011110 (offset) sehingga menjadi **0001100111011110 atau 6622**

- Kelebihan :
  - Model alamat lojik mudah diartikan
  - Mudah diimplementasikan dengan hardware

- **Segmentasi sederhana**
  - Program dan data dibagi2 dalam sejumlah offset
  - Ukuran setiap segment boleh berbeda2
  - Panjang segment memilik batasan maksimum
  - Format alamat => nomor segement + offset
  - Segment table terdiri dari
    - Nomor segment
    - Panjang segement
    - Awal alamat fisik
  - Segmentasi sederhana identik dengan partisi dinamis, karena ukuran segment berbeda-beda
  - Perbedaan segementasi sederhana dengan partisi dinamis
    - Program boleh menempati lebih dari satu partisi
    - Letak program dalam partisi boleh tidak terurut
    - Ukuran fragmentsi lebih kecil
  - Cara mentranslasikan alamat relatif ke alamat absolut
    - Pisahkan bit segment dengan offset
    - Gunakan nomor segment tesebut untuk mengetahui alwamt awal

- Bandingkan nilai offset dengan panjang segment
  - Jika offset lebih besar sama dengan panjang segment makan alamat tersebut tidak valid
- Alamat fisik = awal alamat fisik + offset
- Contoh :

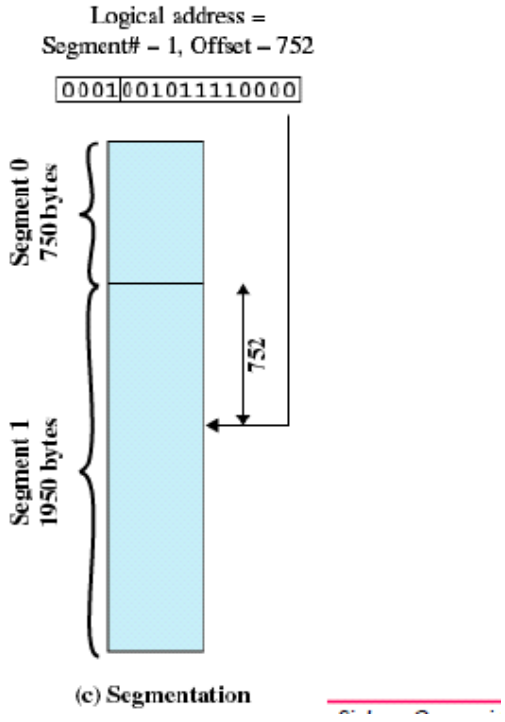
– Sebuah memori menggunakan pengalamatan 16 bit dan dipartisi dengan model segmentasi sederhana dimana jumlah bit untuk nomor segment sebesar 4 bit. Berapakah alamat absolut untuk alamat lojik dengan *segment#* = 1 dan *offset* = 752 ?

– Jawaban:

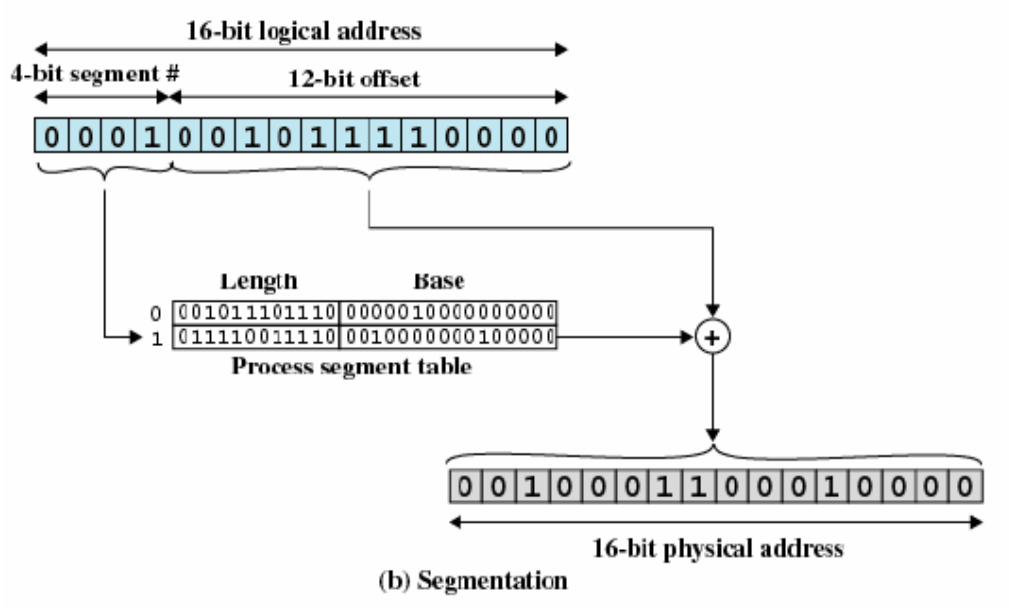
- Nomor *segment* terdiri dari 4 bit, maka:
  – Jumlah segment maksimum = $2^4$ = 16 segment
  – Jumlah bit offset = 16-4 = 12 bit → ukuran segment maksimum adalah $2^{12}$ = 4096

Screen clipping taken: 26/10/2011 16:46

Contoh segmentasi sederhana: (cont'd)




Screen clipping taken: 26/10/2011 16:47




Screen clipping taken: 26/10/2011 16:47

- Alamat lojik dengan *segment#* = 1 dan *offset* = 752 dalam biner adalah, maka:
  - *Segment#* = 1 → 0001
  - *Offset* = 752 → 001011110000
  - Alamat dalam biner adalah 0001001011110000
- Jika awal alamat fisik pada *segment table* untuk alamat lojik tersebut adalah adalah 0010000000100000, maka:
  - Alamat fisik dari alamat lojik tersebut adalah penjumlahan awal alamat fisik dengan *offset* = 0010000000100000 + 001011110000 sehingga menjadi **0010001100010000 atau 8976**

Screen clipping taken: 26/10/2011 16:47

- Kelebihan segmentasi sederhana :
  - Untuk tujuan modularitas, programer dapat membagi-bagi programnya dan ditempatkan dalam segment2 berbeda dalam memori-> lebih fleksibel
- Kelemahan
  - Programer harus tau ukuran maksimum dari segment
  - Implementasi translasi ke hardware lebih kompleks