Nama : ADITO EFRI

NIM : 14121004

PRODI : SISTEM INFORMASI

KELAS : 21(Pagi)

ARTIKEL TENTANG THREAD

Definisi Thread

Thread adalah sebuah pengontrol aliran program pelaksanaan program dengan menggunakan kendali tunggal. Operasi yang paling Modern saat ini adalah sistem yang banyak sekali menyediakan berbagai cara, dan memungkinkan suatu proses terkendali dengan baik.

Pendekatan tradisional sebuah thread eksekusi per-proses, dimana konsep thread tidak dikenal.

Multithreaded Process

Benefits/manfaat
· Kemampuan reaksi
· Sumber daya berbagi
· Ekonomi
· Scalabilas

Thread bermanfaat untuk Multithreading yang berguna untuk Multiprocessor dan Singleprocessor.
Kegunaan untuk system Multiprocessor, adalah :
a) Sebagai unit pararel atau tingkat granularitas pararelisme.
b) Peningkatan kinerja disbanding berbasis proses.
Kegunaan Multithreading pada singleprocessor, adalah :
a) Kerja foreground dan background sekaligus di satu aplikasi.
b) Penanganan asynchronous processing menjadi lebih baik.
c) Mempercepat eksekusi program.
d) Pengorganisasian program menjadi lebih baik.

Multicore Programming
Multicore systems mendesak/memaksa para programmer untuk melewati tantangan yang meliputi :
a) pembagian aktivitas
b) Saldo/Timbangan
c) Data yang telah hancur

d) Ketergantungan Data
e) Pengujian dan debugging

Arsitektur Server Multithreaded

ketika client mengajukan suatu permintaan, pada saat itu juga server akan menuliskan suatu thread yang baru untuk pelayanan atas permintaan yang diajukan oleh client. Selain itu server juga menyimpulkan atau mendengarkan atas permintaan client sehingga permintaan client dapat terpenuhi.

Pelaksanaan Eksekusi Pada Saat Yang Bersaman Pada Suatu Sistem Berinti Tunggal
Single CoreGambar dibawah ini merupakan suatu gambar yang menunjukkan sebuah program yang melakukan dua remote procedure calls (RPC) ke dua host yang berbeda untuk memperoleh hasil gabungannya.

Pada sebuah program single-core, untuk memperoleh suatu hasil proses dilakukan secara berurutan. Penulisan ulang program dengan menggunakan thread-thread yang terpisah bagi setiap RPC-nya menghasilkan kecepatan yang cukup berarti. Apabila program ini beroperasi pada sebuah uniprosesor, maka request harus dibuat secara seri dan hasil beroperasi secara seri, namun program akan menunggu dua jawaban pada waktu yang bersamaan.
Pelaksanaan Paralel Pada Suatu Multicore Sistem

User Threads
1. Pelaksanaan manajemen thread yang dilakukan oleh user-level thread library. Terdapat dua kelompok besar implementasi thread, yaitu user-level thread dan kernel-level thread. Didalam fasilitas user-level thread yang murni, semua tugas manajemen thread dilakukan oleh aplikasi dan kernel tidak mengetahui keberadaan thread.
2. Tiga kunci thread libraries :
· POSIX Pthread
· Win32 thread
· Java thread

Kernel Threads
1. Suatu proses thread Yyang didukung oleh kernel. Untuk memberitahu kejadian kernel, kernel menciptakan scheduler activatioan baru, memberikan ke pemroses dan melakukan upcall ke ruang pemakai.
2. Contoh :
· Windows XP/2000
· Solaris
· Linux
· Tru64 UNIX
· Mac OS X

Multithreading Models

1. Many-to-One : banyaknya User-Level thread yang dipetakan ke kernel thread tunggal, akan tetapi dari beberapa user thread dapat menggunakan satu kernel thread saja.

Contoh :

· Solaries Green Thread
· GNU Portable Thread

2. One-to-One : setiap user-level thread memetakan ke kernel thread, akan tetapi user thread hanya dapat menggunakan satu kernel thread.

Contoh :

· Windows NT/XP/2000
· Linux
· Solaris 9 and later

3. Many-to-Many
· Mengijinkan beberapa user-level thread memakai beberapa kernel thread.
· Mengijinkan system operasi untuk menciptakan beberapa kernel thread.

Thread Libraries

1. Thread libraries menyediakan pemrogram dengan API untuk menciptakan dan memanage thread.
2. Dalam pengimplementasiannya ada dua cara, diantaranya :
· Keseluruhan library pada ruang pemakai.
· Kernel-support library yang didukung dengan OS.

Pthreads

1. Dapat dianggap sebagai user-level atau kernel-level.
2. Suatu POSIX standar (IEEE 1003.1c) API untuk menciptakan thread dan sinkronisasi.
3. API menetapkan sifat thread library, implementasinya adalah pengembangan library.

Java Threads

1. Java thread diatur oleh JVM.
2. Secara khusus diterapkan untuk menggunakan thread model yang disediakan dengan dasar OS.
3. Java thread diciptakan oleh :
· Memperpanjang thread class.
· Menerapkan runnable interface.

Threading Issues

1. Semantics of fork() and exec() system call.
2. Thread cancellation of target thread
3. Asynchronous atau deffered (penundaan atau ketidak serempakan)
4. Signal handling (isyarat menangani)
5. Thread pools : menciptakan sejumlah thread dalam suatu kolom dimana mereka saling menunggu pekerjaan. Keuntungannya yaitu :
· Pada umumnya sedikit lebih cepat untuk melayani suatu permintaan dengan suatu thread yang sudah ada dibandingkan menciptakan suatu thread baru.
· Mengijinkan sejumlah thread dalam suatu aplikasi menjadi seukuran pool.
6. Thread-specific data

7. Scheduler activation

Thread Cancellation
1. pengakhiran suatu thread sebelum selesai.
2. Dua pendekatan umum :
· Cancellationterminate yang tidak serempak dengan penyusupan target.
· Penundaan Cancellationterminate dengan penyusupan target pada waktu tertentu.

Signal Handling
1. Isyarat digunakan Sistem UNIX untuk memberitahu suatu proses bahwa peristiwa tertentu telah terjadi.
2. Suatu signal handler digunakan untuk proses signal :
· Signal yang dihasilkan oleh peristiwa tertentu.
· Signal yang dikirim untuk suatu proses.
· Signal adalah handler.
· Linux mengacu pada thread sebagai tasksrather disbanding thread.
· Thread muncul ketika dilaksanakannya melalui clone() system call.
· Clone()allows merupakan sebuah child untuk membagi dalam pengalokasian ruang alamat yang menjadi tugas parent pada proses.

Windows XP Threads
Pengimplementasian one-to-one menentukan, kernel-level.
Setiap threads meliputi :
· Id thread : suatu nilai unik yang mengindentifikasikan sebuah thread apabila thread itu memanggil server.
· Register set : nilai-nilai register tingkat pengguna yang disimpan.
· Separate user dan kernel stack.
· Private data storage area.
Dari hasil pemahaman diatas, secara garis besar dapat diketahui bahwa…
Thread sering disebut Light Weight Process (LWP), yaitu unit dasar utilisasi pemroses dan berisi program counter, register set dan stack space. Thread-thread di satu proses berbagi (memekai bersama) bagian code, data dan sumber daya system operasi seperti file dan signal. Pemakaian ektensif menyebabkan alih pemroses antara thread-thread di satu proses tidak mahal disbanding alih konteks antar proses. Meski alih thread masih memerlukan alih himpunan register, namun tidak ada keterlibatan manajemen memori.
Multithreading merupakan upaya untuk menigkatkan kinerja system computer, disebabkan :
1. Penciptaan thread baru lebih cepat dibandingkan penciptaan proses baru.
2. Terminasi thread lebih cepat dibandingkan pengakhiran proses.
3. Alih ke thread lain di suatu proses lebih cepat disbanding dari satu proses ke proses lain.
4. Thread-thred di satu proses dapat berbagi kode, data dan sumber daya lain secara nyaman dan efisiensi disbanding proses-proses terpisah.

Kegunaan Thread
Multithreading berguna untuk multiprocessor dan singleprocessor.
Kegunaan untuk system multiprocessor adalah :
· Sebagai unit pararel atau tingkat granularitas pararelisme.

· Peningkatan kinerja disbanding berbasis proses.
Kegunaan multithreading pada singleprocessor, adalah :
· Kerja foreground dan background sekaligus di satu aplikasi.
· Penanganan asynchronous proseccing menjadi baik.
· Mempercepat eksekusi program.
· Pengorganisasian program menjadi lebih baik.
Manfaat utama banyak thread di satu proses adalah memaksimumkan derajat kongkurensi antara operasi-operasi yang terkait erat. Aplikasi jauh lebih efisien dikerjakan sebagai sekumpulan thread disbanding sekumpulan proses.

Karakteristik Thread
Proses merupakan lingkungan eksekusi bagi thread-thread yang dimilikinya. Thread-thread di satu proses memakai bersama sumber daya yang dimiliki proses, yaitu :
· Ruang alamat.
· Himpunan berkas yang dibuka.
· Proses-proses anak.
· Timer-timer.
· Snyal-sinyal.
· Sumber daya-sumber daya lain milik proses.
Tiap thread mempunyai property independen berikut seperti :
· Keadaan (state) eksekusi thread (running, ready dan sebagainya).
· Konteks pemroses. Thread dapat dipandang sebagai satu PC (program counter) tersendiri independen di satu proses.
· Beberapa penyimpan static per-thread untuk variable-variabel local.

Paket Bahasan Perancangan Paket Thread
Paket thread adalah sekumpulan primitive (misalnya library calls) untuk pemrogram berhubungan dengan thread di program aplikasi. Pertimbangan penting pembuatab paket thread adalah :
· Waktu penciptaan thread.
· Penanganan critical region di tingkat thread.
· Penanganan private global variables.
· Implementasi paket thread.

Jenis-jenis Thread Berdasarkan Waktu Penciptaannya
Kategori thread berdasarkan waktu penciptaan :
1. Static threads
Jumlah thread yang akan dibuat ditentukan saat penulisan dan kompilasi program. Tiap thread langsung dialokasikan stack tetap.
Keunggulan à sederhana.
Kelemahan à tidak fleksibel.
2. Dynamic threads
Penciptaan dan penghancuran thread “on-the-fly” saat eksekusi. Penciptaan thread biasanya menspesifikasikan fungsi utama thread (seperti pointer ke procedure) dan ukuran stack, dapat juga ditambah parameter-parameter lan seperti prioritas panjadwalan.
Keunggulan à fleksibel.
Kelemahan à lebih rumit.

Implementasi Paket Thread
Implementasi paket thread :
1. Thread level kernel.
> Keunggulan :
· Memudahkan koordinasi multithread seperti proses server.
· Tidak seboros kumpulan proses tradisional.
> Kelemahan :
· Operasi manajemen thread sangat lebih boros.
· Kernel harus menyediakan semua feature.
2. Thread level pemakai.
> Keunggulan :
Kinerja luar biasa bagus disbanding thread level kernel.
· Tidak diperlukan modifikasi kernel.
· Fleksibelitas tinggi.
> Kelemahan :
· Tidak manfaatkan multiprocessor.
· Untuk aplikasi dengan keterlibatan kernel yang kecil.
· Mengharuskan nonblocking system call.

Struktur Sistem Operasi Multiprocessor
System operasi serupa multiprogrammed uniprocessor. System operasi multiprocessor lebih kompleks karena terdapat banyak pemroses yang mengeksekusi task-task secara kongkuren. System operasi multiprocessor harus dapat mendukung eksekusi banyak task dan mengeksploitasi banyak pemroses untuk meningkatkan kinerja.
Struktur system operasi multiprocessing, antara lain :
1. Separate supervisor
Semua pemroses mempunyai kopian kernel, supervisor dan struktur data sendiri, menanggapi interupsi-interupsi pemakai yang berjalan di pemroses itu. Terdapat struktur data bersama untuk interaksi di antara pemroses-pemroses. Pengaksesan diproteksi menggunakan mekanisme sinkronisasi :
· Tiap pemroses mempunyai perangkat I/O dan system file sendiri.
· Terdapat sedikit coupling diantara pemroses-pemroses.
· Tiap pemroses bertindak sebagai system otonom dan independen.
> Keunggulan :
· Dapat degradasi secara perlahan dalam menghadapi kegagalan pemroses karena hanya terdapat sedikit coupling diantara pemroses-pemroses.
> Kelemahan :
· System sulit melakukan eksekusi pararel satu task tunggal (yang dipecah menjadi subtask-subtask dan menjadwalkan subtask-subtask di banyak pemroses secra kongkuren).
· Tidak efisien karena supervisor, kernel dan struktur data direplikasi di tiap pemroses.
2. Master-slave
Satu pemroses disebut master, bertugas memonitor status dan memberikan kerja ke pemroses-pemroses lain, slave. Slave-slave dipandang sebagai pool sumber daya yang dijadwalkan master. Pemroses-pemrose slave mengeksekusi program-program aplikasi.
> Keungulan :

· Efisien.
· Implementasi (sinkronisasi pengaksesan variable, dan sebagainya) mudah karena system operasi dieksekusi di satu pemroses tunggal.
> Kelemahan :
· Sangat bergantung pada pemroses master.
· Master dapat mengalami botlencek dan berkonsekuensi tidak menggunakan pemroses-pemroses slave secara penuh.
3. Symmetric
Semua proses otonom dan dipandang identik. Terdapt satu kopian supervisor atau kernel yang dapat dieksekusi semua pemroses secara kongkuren.
> Masalah :
· Pengaksesan kongkuren struktur data bersama perlu dikendalikan agar terjaga intergritasnya.
> Solusi :
· Cara paling sederhana adalah memandang seluruh system operasi sebagai satu critical section dan hanya memungkinkan satu pemroses mengeksekusi system operasi di satu waktu. Metode ini disebut floating master karena dapat dipandang sebagai konfigurasi master-slave dimana master di-apungkan/diangkat dari satu pemroses ke pemroses lain.
> Keunggulan :
· Paling fleksibel dan berdaya tahan tinggi.
· Memungkinkan eksekusi satu task tunggal secara pararel.
· Dapat mendegradasi secara perlahan dalam menghadapi kegagalan.
· Penggunaan sumberdaya sangat efisien.
> Kelemahan :
· Perancangan dan implementasi paling sulit.