Thread

Threads

- Fasilitas tread adalah mekanisme yang berguna yang diadopsi pada beberapa OS baru untuk mendukung resource sharing dan akses konkuren
- Penggunaan thread sepadan dengan penggunaan system call “fork” dihasilkan dengan program counter baru, atau thread control dan mengeksekusinya dalam ruang alamat yang sama

Struktur Threads - 1

- Thread (Lightweight Process – LWP) adalah suatu unit dasar dari CPU utilization yang berisi program counter, kumpulan register dan ruang stack
- Thread bekerjasama dengan thread yang lain dalam penggunaan bagian kode, bagian data, resource; seperti open file dan sinyal secara kolektif yang sering disebut task
- Proses tradisional (heavy-weight) sama dengan task dengan satu thread
- Thread harus tepat satu task dan task tidak berarti jika tidak ada thread di dalamnya
- Dengan menggunakan fasilitas thread, CPU dapat secara ekstensif membagi diantara peer thread tanpa menggunakan manajemen memori
- Meskipun context switch pada thread masih memerlukan register set switch, tetapi pembuatan thread lebih murah dibandingkan dengan context switch diantara proses heavy-weight

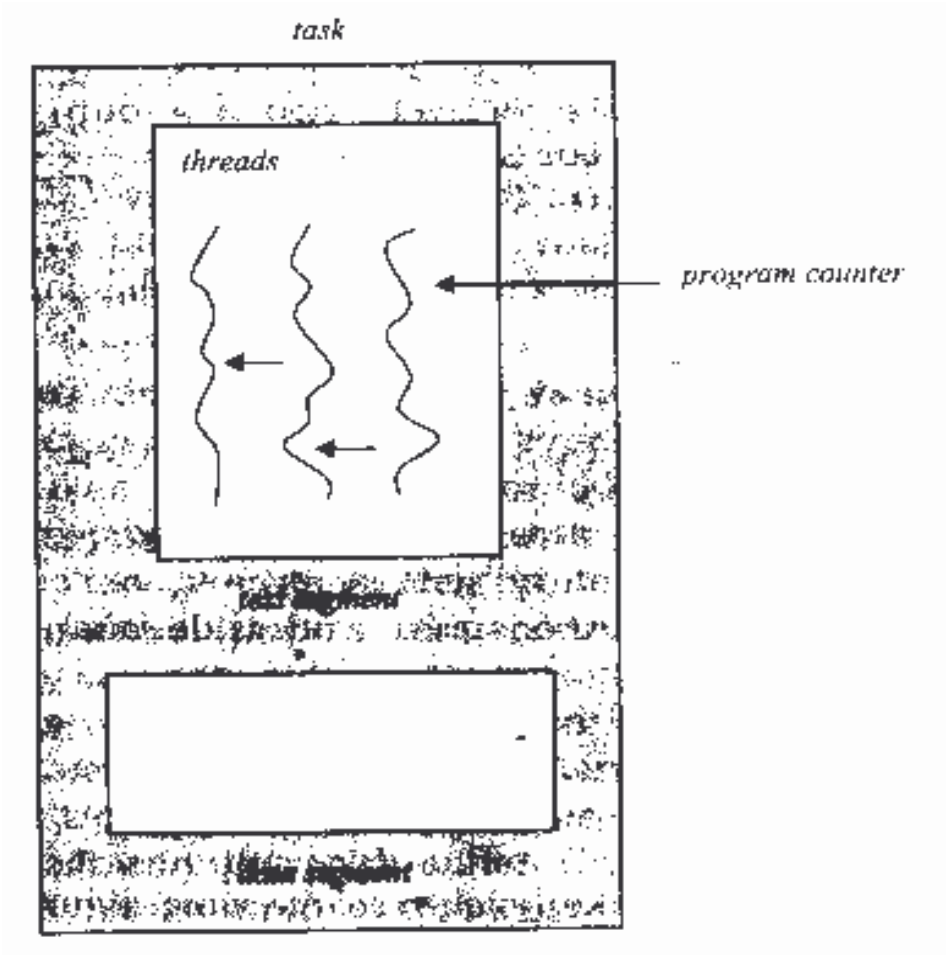
Struktur Threads - 2

- Beberapa sistem mengimplementasikan "user-level threads" pada user-level library daripada melalui system call
  - Sehingga switching thread tidak perlu memanggil OS dan menyebabkan interrupt ke kernel
  - Switching antara user-level thread dapat dilakukan secara independent dari OS (dapat dilakukan lebih cepat)
- Bloking thread dan switching ke thread lain adalah solusi yang dapat diterima untuk permasalahan bagaimana sebuah server dapat menangani beberapa permintaan secara efisien, contoh :
  - Pada OS single-CPU, file server mungkin harus melakukan blok pada saat menunggu akses disk. Tanpa fasilitas thread, tidak ada proses server lain yang dapat dieksekusi samapi proses pertama di-unblok.
  - Pada task yang berisi multiple thread, pada saat satu thread serber di-blok dan menunggu, thread kedua pada task yang sama dapat dijalankan.
- Contoh lain :
  - Permasalahan producer-consumer membutuhkan sharing buffer dan keduanya dapat dipandang sebagai thread pada suatu task. Sehingga, switch hanya membutuhkan ongkos murah dan pada sistem multi-prosessor keduanya dapat dieksekusi secara paralel pada 2 prosessor untuk efisiensi maksimum

Struktur Threads - 3

- Dua contoh alternatif implementasi dari thread :
  1. Thread dapat didukung oleh kernel (pada Mach dan OS/2). Dalam hal ini, kumpulan system call yang setara untuk proses diperlukan
  2. Thread dapat didukung diatas kernel, melalui sekumpulan library call pada level user (pada Project Andrew from CMU)
- Thread level user tidak melibatkan kernel, sehingga lebih cepat melakukan switch daripada thread yang didukung kernel
- Sebaliknya, beberapa call ke OS menyebabkan keseluruhan proses menunggu, karena penjadwalan kernel hanya proses (tidak tahu mengenai threads) dan sebuah proses yang menunggu tidak mendapatkan jatah waktu CPU

Threads - 4



Contoh : Solaris 2

- Solaris 2, versi UNIX, sampai 1992 didukung hanya proses tradisional heavy-weight
- Kemudian ditransformasikan ke modern OS yang mendukung thread baik pada kernel maupun lever user (hybrid), symmetric multi-processing dan penjadwalan real time
- Karakteristik utama :
  1. Thread kernel hanya mempunyai sebuah struktur data yang kecil dan sebuah stack. Switching antara thread kernel tidak memerlukan mengubah informasi memory access, sehingga relatif cepat
  2. LWP berisi PCB dengan register data, informasi akutansi dan informasi memori. Switching antara LWP membutuhkan sedikit pekerjaan tambahan dan relatif lambat
  3. Thread level user hanya memerlukan sebuah stack dan program counter : tanpa membutuhkan resource kernel. Kernel tidak dilibatkan dalam menjadwalkan thread level user sehingga switching relatif cepat
- Meskipun banyak thread level user, tetapi semua kernel selalu melihat LWP pada proses yang mendukung thread level user

Thread pada Solaris 2