COS3 lektion 4

Nis Sarup

28. september 2010

2 System Structures

2.1 Operating System Services

- OS har forskellige services.

2.2 User OS Interface

- 2 slags shells
  - Indbyggede kommandoer
  - Kalder eksterne programmer (Bedre portabilitet)
- GUI = Graphical User Interface

2.3 System Calls

- Systemkald oftest brugt gennem API: WIN32, POSIX, JAVA.
- Forskellige systemer kan have samme API.

2.4 Types of System Calls

- Systemkaldstyper:
  - Processkontrol
  - Fil
  - Device
  - Information
  - Kommunikation
  - Sikkerhed

2.5 System Programer

- Systemprogrammer kan være små og enkle, evt. kun et kald til API, eller mere komplekse.

2.6 OS Design

- At splitte policy (why) fra mechanism (how) er godt: Mere fleksibelt.
- Implementation i high-level sprog = større portabilitet og bedre hastighed med bedre compilere.

2.7 Operating System Structure

- Ny hardware tillader nye OS.
- Microkernel, layered vs. monolithic structure
- Loadable modules.

3 Process Management

3.1 Process Concept

- Process = program in execution = aktivt vs. det inaktive program på disken.
- Processer har states
  - New
  - Running
  - Waiting
  - Ready
  - Terminated
- Process Control Block (PCB) for hver enkelt process indeholder:
  - State
  - PID
  - Program Counter
  - Registre
  - SPU scheduling
  - Memory Management
  - Accounting
  - I/O status
- Threads skifter CPU'en mellem flere processer.

3.2 Process Scheduling

- Queues for forskellige resourcer styrer hvornår en process får adgang til den enkelte resource. F. eks. CPU, Disk, Netværk, etc.
- Short-term schedule vælger processer der er ready in-memory og sender dem til CPU'en.
- Long-term scheduler vælger processer fra disken og lægger dem i ready-køen.
- Medium-term scheduler står for at swappe processor ind og ud.
- (Hvad er forskellen på Medium og Long?)
- Context Switch: State save of current process og state restore of new process.

3.3 Operations in Processes

- Process tree: Parent/children processes.
- PID = Process IDentifier.
- Parent process kan eksekvere sideløbende eller vente på at child-processen terminerer.
- Child processer kan være en kopi af parent eller loades fra et nyt program.
- En process terminere med exit(); kommandoen og returner typisk en integer.

3.4 Interprocess Communication

1. Shared Memory
    - Fælles hukkommelsesområde begge processer kan læse/skrive fra/til.
2. Message Passing
    - Beskeder sendes mellem processer.
    - Processer behøver ikke nødvendigvis at være på samme computer.
    - Mailboxes ejes enten af en process (server) og kan sendes til af flere klienter (men kun læses fra af serveren) eller den kan ejes af sytemet hvor efter processen der lavede mailboxen kan give flere processer læse-adgang.