

Käyttöjärjestelmät II

Prosessit ja säikeet
SMP, Mikroytimet
W2K säikeet
Stallings, Ch 3-4.4

Mitä KJ-I:ssä / KJ-II:ssa?

KJ I + RIO (luvut 3.1-3, 4.1, 5, 6.1-6)

- n **Prosessin tilat, jonot, PCB**
- n **Luonti, prosessinvaihto, prosessi vs. säie, TCB**
- n **User-level vs. kernel-level threads**
- n **Samanaikaisuuden hallinnan perusasiat**

Seuraavaksi KJ-II:ssa (luvut 3-6 loppuun)

- n **UNIX prosessien hallinta**
- n **Moniprosessorijärjestelmä, SMP**
- n **Mikrokernel**
- n **W2K: säikeet, SMP**
- n **Solaris: säikeet, SMP**
- n **Linux: prosessit ja säikeet**
- n **Samanaikaisuuden hallinta eri KJ:ssä**

(luento 3)

(luento 4)

Käyttöjärjestelmät II

UNIX prosessit
Ch 3 [Stal 05]

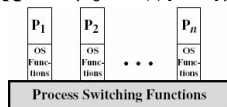
Prosessit yleensä

- n **Käsite**
- n **Tilat** ks. Fig 3.9 [Stal05]
- n **Resurssit** ks. Fig 3.10 [Stal05]
- n **Prosessin tiedot** ks. Tbl 3.4 [Stal05]
 - u kuvaaja PCB ks. Tbl 3.5 [Stal05]

UNIX SVR4 Prosessit

(Fig. 3.15 (b) [Stal05])

- n **Pääosa KJ-työstä tapahtuu käyttäjätason prosessin sisällä**
 - u etuoikeutetussa tilassa (SVC, keskeytys)



- n **Osa KJ-työstä tapahtuu KJ-prosesseissa**
 - u etuoikeutetussa tilassa

- n **Yhdeksän eri prosessin tilaa** ks. Fig. 3.17 [Stal05]
- n **Etuoikeutettu prosessi ei voi menettää suoritinta**
 - u ellei se itse pyydä jotain resurssia, joka ei ole vapaana
- n **Prosessi 0 – boot**
- n **Prosessi 1 – init (mother of all)**

UNIX prosessin kuvaaja

- n **Kääntäjän tietoista: User Level Context**
- n **Suoritinympäristö: Register Context** Tbl 3.10 [Stal05]
- n **KJ:n tiedot: System Level Context**
 - u staattinen
 - F prosessin tilatiedot (Process Table Entry) Tbl 3.11 [Stal05]
 - F prosessin hallintotiedot (U Area) Tbl 3.12 [Stal05]
 - u dynaaminen
 - F muistialueet (Region Table)
 - F oma pino etuoik. tilan koodille (Kernel Stack)

UNIX prosessin luonti

- n **Ytimen systeemikutsu `pid=fork()`**
 - u varaa paikka prosessitaulusta Process Table
 - u anna uniikki ID (nelinumeroinen)
 - u kopioi vanhemman koko prosessinkuvaaja uudelle prosessille (tai luo ihan uudet rakenteet!)
 - ┆ vain linkki yhteiseen muistialueeseen
 - u lisää aukiolevien tiedostojen käyttäjien määrää yhdellä (jos tavallinen "sibling")
 - u siirrä uusi prosessi RR-jonoon
 - u palauttaa lapsen pid:n vanhemmalle ja pid=0 lapselle (tämä on ainoa ero koodinäkökulmasta)
 - ┆ esimerkki seuraavalla kalvolla

UNIX fork()

```
while (TRUE) { /* repeat forever */
    type_prompt(); /* display prompt on the screen */
    read_command(command, params); /* read input line */
    pid = fork(); /* fork off a child process */

    if (pid < 0) {
        printf("Unable to fork0); /* error condition */
        continue; /* repeat the loop */
    }

    if (pid != 0) { /* parent waits for child */
        waitpid(-1, &status, 0);
    } else { /* child does the work */
        execve(command, params, 0);
    }
}
```

Fig. 10-7 [Tane01]. A highly simplified shell.

Käyttöjärjestelmät II

Säikeet Symmetric MultiProcessors (SMP) Mikroytimet

Ch 4 [Stal 05]

Prosessit ja säikeet

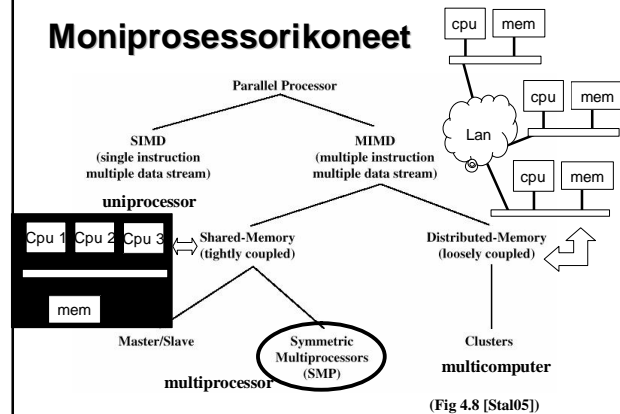
- n **Ennen (kun ei säikeitä)**
 - u prosessi:
 - ┆ resurssien hallinta
 - ┆ suorittava (laskentaa tekevä) yksikkö
- n **Nyt**
 - u prosessi
 - ┆ resurssien hallinta ks. Fig 4.1 [Stal05]
 - u säie
 - ┆ säikeet toteuttava yksikkö ks. Fig 4.2 [Stal05]
 - ┆ samanaikainen suoritus usealla suorittimella? ks. Fig 4.3 [Stal05]
 - ┆ samanaikainen suoritus usealla suorittimella? ks. Fig 4.6 [Stal05]
 - ┆ samanaikainen suoritus usealla suorittimella? ks. Fig 4.7 [Stal05]

Flynnin jaottelu moniprosessorikoneille

Michael J. Flynn, 1966

- n **SISD: Single Instruction - Single Data**
 - u yksi käsky, yksi data
 - u normaali CPU: yksi ALU, yksi CU, yksi MEM
- n **SIMD: Single Instruction - Multiple Data**
 - u yksi käsky, monta data-alkiota
 - u taulukkoprosessori: monta erikoistunutta ALUa
 - u vektorisuoritin: vektorikonekäskyt, vektorirekisterit
- n **MISD: Multiple Instruction - Single Data**
 - u Ei mielekäs
- n **MIMD: Multiple Instruction - Multiple Data**
 - u monta käskyä ja monta dataa
 - u tiukasti kytketty: monta CPU:ta, yhteinen MEM
 - u löyhästi kytketty: monta CPU:ta, kullakin oma MEM

Moniprosessorikoneet



Master-slave vs. SMP

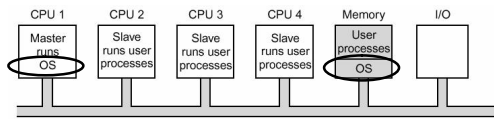


Fig. 8-8. A master-slave multiprocessor model. [Tane 01]

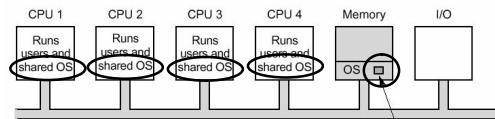


Fig. 8-9. The SMP multiprocessor model. [Tane 01]

Toinen jaottelu yhteisen muistin moniprosessorikoneille

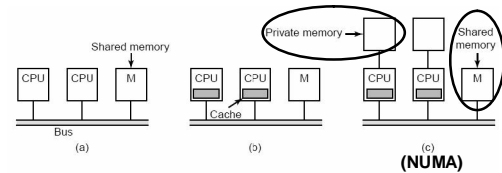
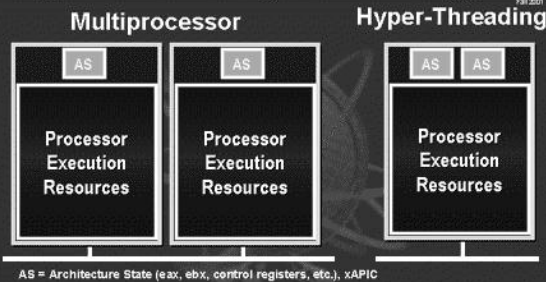


Fig. 8-2. Three bus-based multiprocessors. (a) Without caching. (b) With caching. (c) With caching and private memories.

[Tane 01]

Hyper-Threading Technology

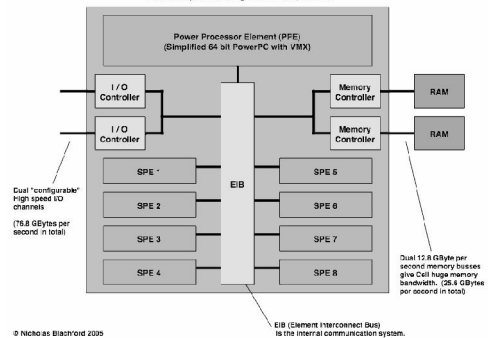


AS = Architecture State (eax, ebx, control registers, etc.), xAPIIC

Hyper-Threading Technology looks like two processors to software

Cell Processor Architecture

This diagram is based on data released by STI. Some acronyms have changed: SPE = APU, PPE = PU



Kolmas jaottelu yhteisen muistin moniprosessorikoneille

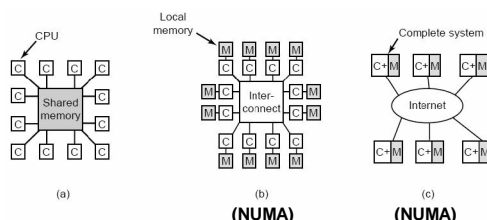
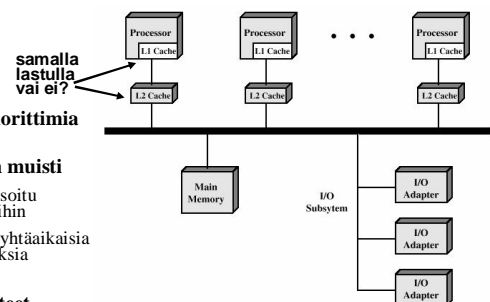


Fig. 8-1. (a) A shared-memory multiprocessor. (b) A message-passing multicomputer. (c) A wide area distributed system.

[Tane 01]

SMP: Symmetric Multiprocessing



- Useita suorittimia (CPU)
- Yhteinen muisti
 - organisoitu lohkoihin
 - useita yhtäaikaista viittauksia
- Yhteiset oheislaitteet
- Välimuistien yhteneväisyys (coherency)?

(Fig. 4.9 [Stal 05])

Huomioitava SMP KJ:ssä

- n **Samanaikaisuuden hallinta** (Ch 5-6, RIO)
 - u KJ voi olla suoritettavana usealla CPU:lla
 - u KJ:n oltava vapaakäyntinen (re-entrant)
 - u KJ:n tietorakenteiden käsittelyssä tarvitaan poissulkemista
- n **Vuorotettavana useita CPU:ita** (Ch 10)
 - u KJ:kin voi olla suorituksessa yhtäaikaan eri CPU:illa
 - u saman prosessin säikeiden vuorottaminen
- n **Synkronointi, lukot** (Ch 5-6, RIO)
 - u poissulkeminen ja tapahtumien järjestys
- n **Muistinhallinta** (Ch 7-8)
- n **Ulkoisten keskeytysten käsittely**
- n **Vikasioitaisuus: joku CPU voi pudota pois**
 - u tästä ei tarvinnut murehtia yhden CPU:n kanssa!
 - u ollaanko vikasioitaisia vai ei? miten?

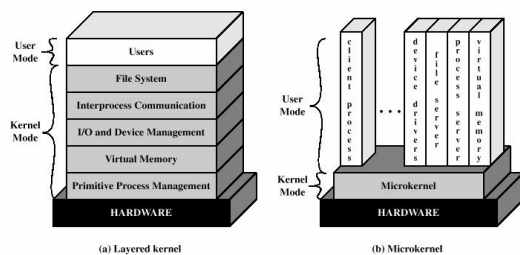
Käyttöjärjestelmät II

Mikroytimet

Ch 4.3 [Stal05]

KJ:n ydin

(Fig 4.10 [Stal 05])



Monoliittinen ydin

- n "**Kokoelma KJ:hin kuuluvia funktioita**", jotka käännetään yhdeksi binäärikoodiksi
 - u koko KJ etuoikeutetussa tilassa
 - u funktio voi kutsua suoraan toista funktiota

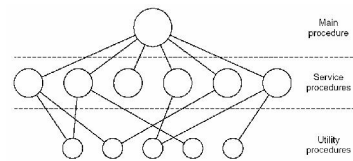


Fig. 1-24. A simple structuring model for a monolithic system.

[Tane 01]

Monoliittinen ydin

- n **Periaatteessa kaikki KJ:n osat pääsevät käsiksi kaikkiin tietorakenteisiin**
 - u globaalien muuttujien käyttö
 - F ei edes parametrien kopiointi hidastamassa
 - u kopioi itse tieto suoraan paikalleen
 - F ei sanomanvälitystä
 - F ei *etuoikeutettu tila* ↔ *käyttäjätila* -vaihtoja
- n **Sovellus käyttää palvelua palvelupyynnöllä**
 - u parametrit esim. pinon, keskeytys
 - u 'vastaus' suoraan annettuihin muuttujiin tai pinossa
- n **Nopeus, tehokkuus, yksinkertaisuus**
- n **Ongelmat?**

Mikroydin (mikrokernel)

- n **Etuoikeutetussa tilassa toimiva pieni KJ:n ydin (core functions)**
 - u tärkeät laitetason liittymät
 - u minimijoukko muita perustoimintoja
- n **Muut alijärjestelmät käyttäjätilassa**
 - u laiteajurit
 - u tiedostojärjestelmät
 - u virtuaalimuistin hallinta
 - u ikkunoitu käyttöliittymä
 - u turvallisuusosia, mm. käyttäjien tunnistus
- n **Alijärjestelmät eivät pääse suoraan käsiksi laitteistoon tai toisiinsa**
 - u käyttö sanomanvälityksen kautta

Mikroytimen kehuja

- n **Yhtenäinen palvelujen käyttötapa**
 - u sama mekanismi etuoikeutetussa ja käyttäjätilassa
 - u sanomanvälitys: send ja receive
- n **Laajennettavuus**
 - u helppo lisätä uusia palveluja (KJ:kin kehitty...)
 - u helppo lisätä uusia liittyviä vanhoihin moduuleihin
- n **Joustavuus**
 - u piirteitä poistamalla saadaan räätälöity versio
 - u paikkopaketit voidaan ottaa mukaan boottaamalla konetta
 - u esim: sulautettu järjestelmä - minimijoukko toiminnallisuutta
 - F kännykkä, kämmenmikro, videonauhuri, tv, ...
- n **Sirrettävyys**
 - u laitesidonnaiset osat rajattu mikroytimeen
 - u uusi laitteisto vaatii vain mikroytimen ronkkimista

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

25

Mikroytimen kehuja

- n **Luotettavuus**
 - u modulaarinen rakenne helpompi toteuttaa ja testata
 - u jos joku palveluosa kaatuu, muut voivat silti toimia
- n **Sopii hajautettuun järjestelmään**
 - u paikallisen ja etäpalvelun käyttö samanlaista
 - u 'single microkernel image': jos palvelut ja prosessit nimetty kaikissa koneissa yksikäsitteisesti
 - F eri koneissa sama palveluiden nimentä ja osoitteet
- n **Sopii oliopohjaisen KJ:n toteutukseen**
 - u mikroydin yksi olio muiden (palvelija)olioiden joukossa
 - u kommunikointi vain tunnettujen rajapintojen kautta
 - F private data, public interfaces

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

26

Mikroytimen heikkouksia

- n **Tehokkuus**
 - u yhteiskäytössä oleva vs. paikallinen muisti
 - u sanomanvälitys vs. suora funktiokutsu
 - u paljon **user mode** ↔ **kernel mode** -vaihtoja
 - F viestien käyttö vs. suorat funktiokutsut
- n **Apu: enemmän toimintoja takaisin suoritettavaksi etuoikeutettuun tilaan**
 - u esim. Mach, Chorus, W2K
 - u laiteajureita takaisin etuoikeutettuun tilaan
 - u muita paljon käytettyjä palveluja
 - u kompromissi - mitä menetetään?

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

27

Mikroytimen peruspalvelut

- n **Mitkä?**
 - u ei täysin yleispäteviä sääntöjä
 - u ainakin laitteistoa suoraan ronkkivat osat
 - u tuki käyttäjätilan palveluille ja sovelluksille
 - F kommunikointi
- n **Minimi:**
 - u muistinhallinnan alimmat tasot (low-level MM)
 - u prosessien välinen kommunikointi (IPC)
 - u siirrännän laitesidonnaiset osat
 - u keskeytysten käsittely
 - u prosessin vaihto (basic scheduling)

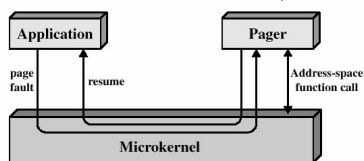
KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

28

Mikroytimen peruspalvelut

- n **Muistinhallinnan alimmat tasot**
 - u virtuaalisivujen kuvaus fyysisiksi sivuiksi ytimessä
 - F sivutaulu ja MMU:n asetukset
 - u sivutus ja virtuaalimuistin hallinta mikroytimen ulkopuolella
 - F sivupuutoksen käsittely
 - F sivujen poisto- ja sijoitusalgoritmit



(Fig. 4.11 [Stal 05])

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

29

Mikroytimen peruspalvelut (jatkuu)

- n **Prosessien välinen kommunikointi (IPC)**
 - u Perusmekanismina sanomanvälitys
 - F pyyntö / vastaus
 - F jos prosesseilla ei yhteiskäyttöistä muistia, sanomanvälitys muistista muistiin kopiointia (etuoik. tilassa)
 - u Sanoma: (keneltä, kenelle, data)
 - F data = esim. palvelun nimi ('portti') ja sen parametrit
 - F datan paikalla voi olla osoite, josta data löytyy
 - u Palveluun voi liittyä oikeuksien tarkistus KJ:ssä
 - F kirjanpito 'porteista' ja pääsyoikeuksista
 - F prosessi voi muuttaa pääsyoikeuksia
 - u Poissulkemisen ja synkronoinnin primitiivit

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

30

Mikroytimen peruspalvelut (jatkuu)

- n **Siirräntä**
 - u laiteohjaimen rekistereiden käyttäminen, DMA
 - u I/O-porttien sijoittaminen prosessin osoiteavaruuteen
- n **Keskeytysten käsittely**
 - u mikroytimen tunnistaa keskeytykset, mutta erillinen käyttäjätason palvelu käsittelee ne
 - F keskeytysvektori: mikroytimen pitää kirjata keskeytysnumeroista ja niihin liittyvistä palvelijoista
 - F generoi ja lähettää sanoma käsittelijälle (esim. ajurille)
 - välit tarvittaessa ohjaimen rekistereiden arvot
 - F kuittaa keskeytys huomatuksi
 - F lyhyt ja nopea?

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

31

Laiteajuri

- n **Käyttäjätalassa toimiva prosessi / säie**
 - u saa pyyntöjä sovelluksilta ja laitteistolta

```

while (true) do {
    waitFor (msg, sender);
    if (sender == my_sw_client) {
        read/write I/O-ports
        ...
    }
    if (sender == my_hw_interrupt) {
        read/write I/O-ports
        reset hardware interrupt
    }
}
    
```

- u I/O-portit ajurin osoiteavaruudessa (muuten tiedot sanomina)

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

32

Käyttöjärjestelmät II

Windows 2000 (eli W2K)

Säikeet ja SMP

Ch 4.4 [Stal 05]



KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

33

W2K resurssien ja suorituksen hallinta

- n **Työ (job)**
 - u prosessien kokoelma, jolla yhteiset rajoitukset
 - u max prosessien lkm, max CPU aika per prosessi,
- n **Prosessi (process)**
 - u omistaa resurssit
 - u koostuu useasta säikeestä
- n **Säie (thread)**
 - u oma access token (saatu esim. asiakkaalta)
 - u suorittava vuorotuksen yksikkö
 - u kaksi pinnoa, oma user ja kernel tiloissa suoritusta varten
- n **Kuitu (fiber) – "kevytsäie" (nopea vuorotus)**
 - u ydin ei tiedä kuiduista mitään Win32 API hoitaa kaiken
 - F hyvin nopea vuorotus user tilassa (Win32 API hoitaa)
 - F jos kuitu blokkaa, niin toinen saman säikeen kuitu vuorotetaan automaattisesti (säie ei siis blokkau)

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

34

W2K: Säikeet

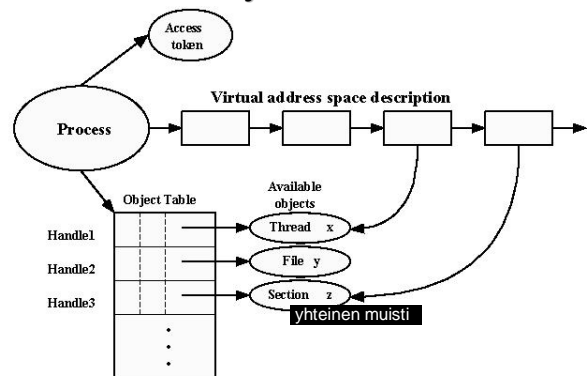
- n **W2K tukee useita erilaisia ajoympäristöjä (Win16, Win32, Posix?, OS/2?), eroja:**
 - u prosessien nimeäminen
 - u säikeitä vai ei
 - u prosessin kuvaaminen
 - u prosessin resurssien suojaus
 - u prosessien välinen kommunikointi
 - u prosessien sukulaisuussuhteet
- n **Perusta**
 - u prosessit ja säikeet toteutettu olioina
 - u prosessi = yksi tai useampi säie
- n **Sekä prosessi- että säieoliolla synkronointiprimitiivejä**

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

35

W2K: Prosessi ja sen resurssit



KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

(Fig 4.12 [Stal05])

36

W2K: Prosessi ja sen resurssit

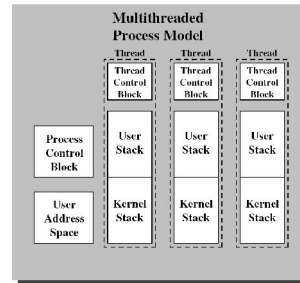
- n **Oliotaulukko (object table)**
 - u pääsy olioiden tietorakenteisiin
 - u prosessiin liittyvät säikeet
- n **Pääsypoletti (pääsylippu?) istuntoa käynnistettäessä**
 - u määrää oletusoikeudet luotaville olioille access token
 - F esim. säikeille
 - F myöhemmin säie voi saada esim. asiakkaan pääsypoletin, jolloin lisää oikeuksia
 - u rajoittaa / sallii olioiden käyttöä
 - F kuka saa käyttää, kuka ei
 - F käyttötapa: read, write, change, ...
- n **Osa tietorakenteista vain KJ:n hallussa**
 - u sovelluksella ei suoraa kahvaa (esim. prosessin oma pääsypoletti)

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

37

W2K Prosessi ja säikeet (Fig 4.2 [Stal 05])



Prosessi:
yhteisten resurssien kirjanpito, passiivinen

Säie:
omien resurssien kirjanpito, vuorottamisen yksikkö, aktiivinen

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

38

W2K Säikeiden yhteistyö

- n **Yhteiskäyttöinen muisti**
 - u saman prosessin säikeet voivat aina käyttää
 - u järjestyy myös eri prosessien säikeiden välille
- n **Sanomanvälitys**
 - u kuka, kenelle, data
- n **Semaforit**
 - u poissulkeminen ja synkronointi

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

39

W2K Prosessi- ja säieoliot

- n **prosessiolio**
 - u attribuutit: I/O counters, quota limits, debug ports, ...
 - u palvelut
 - F create process
 - F terminate process
- n **säieolio**
 - u attribuutit: processor affinity, base priority, ...
 - u palvelut
 - F create thread
 - F suspend, resume
 - F get context

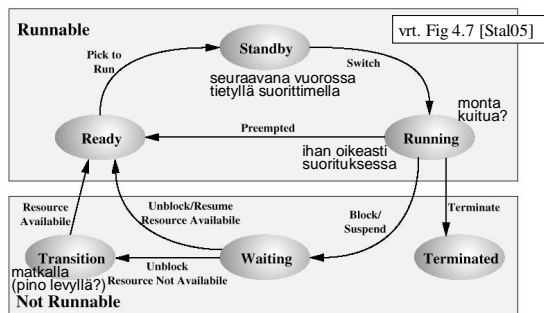
Fig 4.13 [Stal05]

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

40

W2K Säietilat



(Fig 4.14 [Stal05])

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

41

W2K: SMP

- n **Prosessin säikeet voivat olla yhtäaikaan eri prosessoreilla**
 - u myös W2K Executive
- n **Prosessi/säie voidaan sitoa tietyille prosessorijoukkoille (affinity)**
 - u tieto kirjattu sekä PCB:hen että TCB:hen
 - u vuorotetaan aina samoille prosessoreille
 - u thread affinity \subset process affinity
 - u pehmeä sidonta (soft affinity) – yritä edes (välimuistin vuoksi)
 - u kova sidonta (hard affinity) – muu ei kelpaa
- n **Pyritään vuorottamaan uudelleen samalle CPU:lle**
 - u välimuisti

Affinity: kinship by marriage or adoption, the force attracting atoms to each other
[hyperdictionary]

KJ-II K2006 / Auvo Häkkinen - Teemu Kerola

13.3.2006

42

Kertauskysymyksiä

- ⁂ Miksi Unix prosesseilla on tila zombie?
Mitä kyseinen prosessi tekee enää järjestelmässä?
- ⁂ Mitkä ovat suurimmat erot mikroydin KJ:llä ja monoliittisella KJ:llä?
- ⁂ Milloin W2K sovellus olisi parempi toteuttaa usealla kuidulla yhdessä säikeessä kuin usealla säikeellä? Miksi?
- ⁂ Mitä hyötyä/haittaa on siitä, että W2K prosessit ja säikeet ovat olioita?
- ⁂ Mitä eroa on Linux'illa ja W2K'illa prosessien ja säikeiden suhteen?