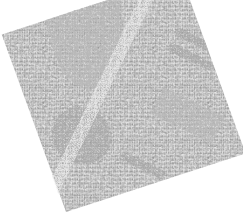


Jakso 8

Ohjelman toteutus järjestelmässä



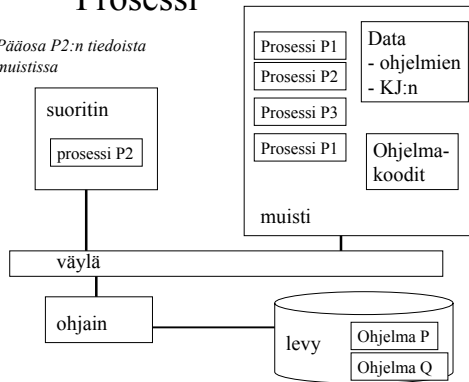
Prosessi
 Prosessin esitysmuoto
 järjestelmässä
 Käyttöjärjestelmä
 KJ-prosessit

Prosessi ⁽⁴⁾

- Järjestelmässä olevan ohjelman esitysmuoto
- Järjestelmässä voi olla ”samalla kertaa” monta prosessia joko samasta tai eri ohjelmasta
 - käyttäjän (ihmisen) näkökulma ja aikaskaala (1 min, 1 sek?)
- suorittimella on yksi prosessi kerrallaan suorituksessa
 - laitteiston näkökulma ja aikaskaala (1 ms, 1 μs, 1 ns?)
- Muut prosessit ovat odottamassa jotakin
 - suoritinta?, I/O:ta?, viestiä toiselta prosessilta?
 - vapaata muistitilaa

Prosessi

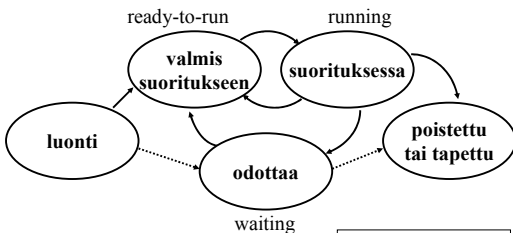
Pääosa P2:n tiedoista muistissa



Prosessin vaihto ⁽⁴⁾

- Suorittimella suoritusvuorossa olevan prosessin vaihtaminen
- Tapahtuu aika usein
 - keskimäärin noin 2000-3000 konekäskyn välein?
 - Esim. 50-500 kertaa sekunnissa?
- Iso operaatio - paljon kopiointia
 - satoja konekäskyjä 50-500?

Prosessin elinkaari ⁽⁹⁾



Prosessin viisi suoritustilaa

- Milloin tilanvaihto tapahtuu
- Mitä tilanvaihdossa tapahtuu?

Mitä suoritin tietää suorituksessa olevasta prosessista?

Prosessin esitysmuoto järjestelmässä ⁽⁴⁾

- PCB - Prosessin kuvaaja eli kontrollilohko (PCB - process control block)
 - isohko tietue, joka sisältää kaiken yhdestä prosessista
 - muistialueet, tiedostot, tiedostojen käsittelykohdat
 - ei suorituksessa oleville myös: suorittimen tila (laiterekisterit, MMU:n rekisterit, kontrollirekisterit)
 - joka prosessista oma PCB
 - luodaan prosessin luonnin yhteydessä ja tuhoaan prosessin päättyessä
 - käyttöjärjestelmäruutiini manipuloivat PCB:tä

Prosessin kuvaajan sisältö (9)

- Prosessin tunniste 14023
- Prioriteetti suorittimen vuoronantoa varten 143
- Prosessin tila ja/tai odottamisen syy R-to-R
- Suoritinympäristö talletettuna odottamisen aikana
 - rekisterit, PC, SP, FP, tilarekisterit
- Seuraavaksi suoritettavan käskyn osoite Main {}
- Poikkeuskäsittelijöiden osoitteet (ellei oletusarv.)
- Aikaviipale
- Käytössä olevat muistialueet, aukiolevat tiedostot
- KJ:n hallintotietoa (kokonaisaika, etc etc)

27/05/2004

Copyright Teemu Kerola, K2003

7

Prosessin tilanvaihdon toteutus (9)

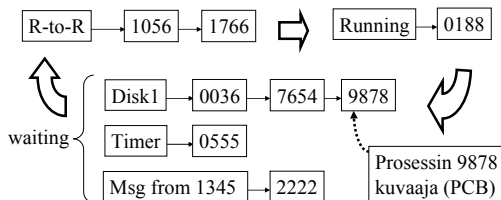
- Prosessin tilanvaihto tapahtuu siirtämällä prosessi (sen PCB) jonosta toiseen
 - ready-to-run -jono (tai jonot)
 - running-jono
 - jota ei oikeastaan ole olemassa
 - waiting-jono
 - joka tyypille oma jononsa
 - esim: laitteen Disk1 I/O:n valmistumista odottavat
 - esim: näppäimistön painallusta odottavat
 - esim: kellolaitekeskeytystä odottavat
 - esim: prosessilta 1345 signaalia odottavat

27/05/2004

Copyright Teemu Kerola, K2003

8

Prosessit jonoissa (1)



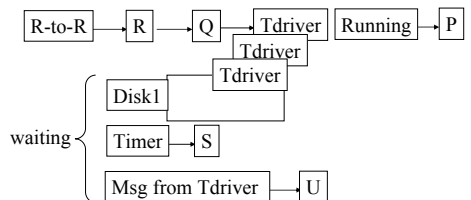
Vuoronanto:
valitse seuraava prosessi Ready-to-Run -jonosta ja
siirrä se suoritukseen CPU:lle
 (kopioi tämän prosessin **suorittimen tila** suorittimelle)

27/05/2004

Copyright Teemu Kerola, K2003

9

KJ esimerkki: I/O keskeytys (5)



I/O keskeytys laitteelta Disk1 prosessille Tdriver?

Suoritin havaitsee keskeytyssignaalin ja suorittaa I/O keskeytyksikäsitteilyrutiinin (P:n ympäristössä)

Tdriver siirretään R-to-R jonoon

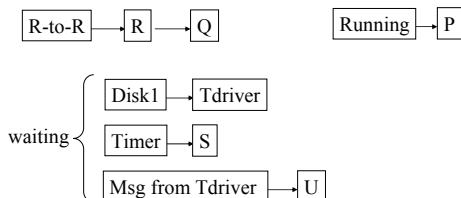
P:n suoritus jatkuu vai jatkuuko?

27/05/2004

Copyright Teemu Kerola, K2003

10

KJ esimerkki: I/O keskeytys (ei anim)



I/O keskeytys laitteelta Disk1 prosessille Tdriver?
 Suoritin havaitsee keskeytyssignaalin ja suorittaa I/O keskeytyksikäsitteilyrutiinin (P:n ympäristössä)

Tdriver siirretään R-to-R jonoon

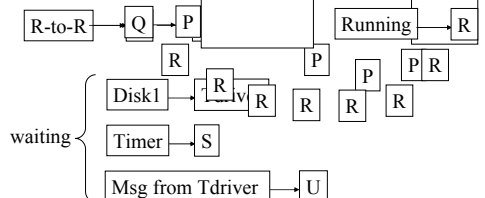
P:n suoritus jatkuu vai jatkuuko?

27/05/2004

Copyright Teemu Kerola, K2003

11

KJ esim: aikaviipalekeskeytys (7)



P saa aikaviipalekeskeytyksen?

P siirretään takaisin R-to-R jonoon

Seuraava prosessi R saa suoritusvuoron

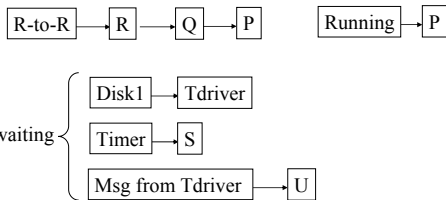
Entä jos P olisi pyytänyt levy I/O:ta Disk1:ltä?

27/05/2004

Copyright Teemu Kerola, K2003

12

KJ esim: aikaviipalekesk. (ei anim)



P saa aikaviipalekeskeytyksen?

P siirretään takaisin R-to-R jonoon

Seuraava prosessi R saa suoritusvuoron

Entä jos P olisi pyytänyt levy I/O:ta Disk I:ltä?

27/05/2004

Copyright Teemu Kerola, K2003

13

Prosessin vaihto (4)

- Vaihdon tekee KJ rutiini sillä hetkellä suorittavan prosessin ympäristössä
- Talleta vanhan prosessin suoritusympäristö suorittimelta omalle talletusalueelle muistiin
 - talleta kaikki suorittimella olevat tiedot muistiin
- Kopio uuden prosessin suoritusympäristö omalta talletusalueeltaan suorittimelle
 - lataa kaikki suorittimen rekisterit (myös PC!)
- Uuden prosessin suoritus jatkuu täsmälleen siitä mihin viime kerralla jäätin
 - sama konekäsky, käytännössä sama suoritusympäristö!
 - usein keskellä prosessin vaihtoa suorittavaa KJ- rutiinia

27/05/2004

Copyright Teemu Kerola, K2003

14

Prosessin prioriteetti (3)

- Prosessin tärkeysjärjestys suorittimella
 - esim. pieni numero \Rightarrow iso prioriteetti
- Joka prioriteetti (luokalle) oma R-to-R jononsa
 - KJ prosesseilla parempi prioriteetti kuin käyttäjätason prosesseilla
 - tosiaikasovelluksen prosesseilla parempi prioriteetti kuin KJ prosesseilla
 - muistakaa antaa KJ:lle aikaa aina joskus ... !
- Prioriteetti voi vaihdella prosessin elinaikana
 - paljon suoritinaikaa \Rightarrow huonompi prioriteetti
 - kauan R-to-R jonossa \Rightarrow parempi prioriteetti (prosessi siirretään korkeamman prioriteetin R-to-R jonoon)

27/05/2004

Copyright Teemu Kerola, K2003

15

Käyttäjän näkökulma (käyttö)järjestelmään

- Miten järjestelmä toimii minun ohjelmani kanssa?
- Onko järjestelmä riittävän nopea pelaamaan suosikkipelejäni isolla näytöllä?
- Onko minun helppo asentaa uusi ohjelma koneelle?
- Onko minun helppo muuntaa (portata) ohjelmani tähän käyttöjärjestelmään?
- Miten muistin lisääminen vaikuttaisi minun ohjelmani nopeuteen?

27/05/2004

Copyright Teemu Kerola, K2003

16

Käyttöjärjestelmän näkökulma (käyttö)järjestelmään

- Ovatko kaikki systeemin resurssit mahdollisimman hyvässä käytössä?
- Mikä on keskimääräinen jonon pituus (prosessien lukumäärä) suorittimella?
- Minkä osan ajasta suoritin odottaa järkevää työtä?
- Minkä osan ajasta kovalevyn hakuvarsi on liikkeessä?
- Miten usein datamuistiviitteet löytyvät välimuistista ?
- Miten muistin lisääminen vaikuttaisi nopeuteen?

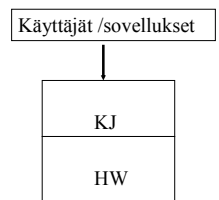
27/05/2004

Copyright Teemu Kerola, K2003

17

Käyttöjärjestelmä käyttöliittymänä laitteistoon

- Loppukäyttäjälle (ihmiselle)
- Sovellusohjelmalle
- Piilottaa laitteiston erityispiirteet käyttäjiltä
 - käskykanta
 - konekäskyn rakenne
 - suorittimen toteutus ja suorittimien lukumäärä
 - I/O:n toteutus
 - I/O-laitteiden sijainti



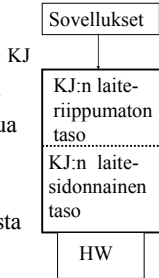
27/05/2004

Copyright Teemu Kerola, K2003

18

Käyttöjärjestelmän tavoitteet

- Laiteriippumaton (HW-riippumaton) käyttöliittymä laitteistoon
 - järjestelmää on helppo käyttää
 - järjestelmä antaa reilua palvelua kaikille
 - sovellukset on helppo tehdä
 - sovellukset helppo siirtää muista järjestelmistä



27/05/2004

Copyright Teemu Kerola, K2003

19

Käyttöjärjestelmän tavoitteet (jatk)

- Järjestelmän resurssien tehokas hallinta
 - kaikista resursseista saatava maksimihyöty
 - joustava resurssien yhteiskäyttö
 - tiukka tietosuoja

27/05/2004

Copyright Teemu Kerola, K2003

20

Käyttöjärjestelmä resurssien vartijana

- Suoritinaikaa reilusti kaikille
 - kukaan ei odota suoritinta ikuisesti
 - kriittiset prosessit saavat ajoissa suoritinaikaa
- Tiedostojen (koodi, data) tehokas käyttö
 - laitteistosta ja sijainnista riippumaton käyttö
 - helppo yhteiskäyttö ja samalla tietojen suojaus
- Tietoliikenneverkkojen käyttö
 - laiteriippumaton käyttö
 - yhteiskäyttö ja tiedon suojaus
- Hallintokirjanpito

27/05/2004

Copyright Teemu Kerola, K2003

21

KJ järjestelmän eheyden turvaajana

- Varauduttu kaikkiin mahdollisiin virheisiin
- Sovellusohjelmat eivät voi häiritä KJ:tä tai muita prosesseja
 - tahallaan (esim. Tietokoneviirukset)
 - vahingossa (yleisin tapa)
- Järjestelmä ei lukkiudu tai 'kaadu'
 - KJ:n omat tietorakenteet aina eheitä
 - sovellusohjelmat eivät koske KJ:n tietorakenteisiin
 - sovellusohjelmat aina lopulta antavat vuoron KJ:lle

27/05/2004

Copyright Teemu Kerola, K2003

22

Käyttöjärjestelmän toteutus ⁽⁵⁾

- Joukko prosesseja ja proseduureja
 - prosessit elävät omaa elämäänsä (etuoikeutetussa tilassa eli root'ina?) koko ajan
 - swapper (Unix) - muistinhallintaprosessi
 - init prosessi (Unix) - kaikkien käyttäjätason prosessien "äiti"
 - laiteajurit
 - proseduurit suoritetaan sen hetkisen prosessin ympäristössä (etuoikeutetussa tilassa?)
 - keskeytyskäsitteijät
 - Saavat kontrollin aina tarvittaessa
 - keskeytykset, SVC, ajastimet

27/05/2004

Copyright Teemu Kerola, K2003

23

KJ-palvelun kontrollin palautus

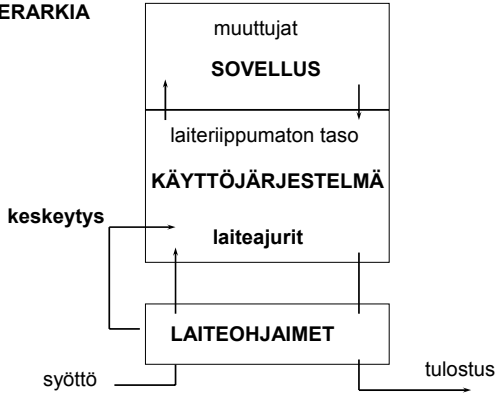
- Aliohjelmakutsut
 - CALL — EXIT
- SVC
 - SVC — IRET
- Viestit
 - viesti — vastausviesti
 - lähettäjä odottaa vastausta RECEIVE:ssä
- Ajastimet ja muut keskeytykset
 - keskeytykset — IRET

27/05/2004

Copyright Teemu Kerola, K2003

24

SIIRRÄNNÄN HIERARKIA



KJ-esimerkki: laiteajuri

- Aliohjelmuna (eli proseduurina)
 - laiteajuri suoritetaan KJ-rutiinina tavallisen SVC-kutsun kautta
 - Vain yksi kutsu kerrallaan suorituksessa? Miksi?
 - Miten voidaan valvoa?



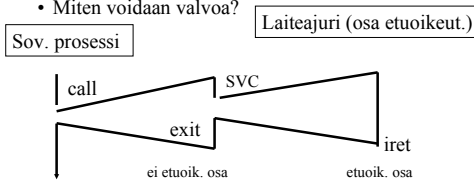
27/05/2004

Copyright Teemu Kerola, K2003

26

KJ-esimerkki: laiteajuri

- Aliohjelmuna (eli proseduurina)
 - laiteajuri suoritetaan KJ-rutiinina tavallisen aliohjelmakutsun tai SVC-kutsun kautta
 - osa tai kaikki koodista voi olla etuoikeutettua
 - Vain yksi kutsu kerrallaan suorituksessa? Miksi?
 - Miten voidaan valvoa?



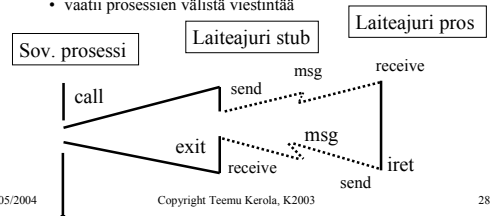
27/05/2004

Copyright Teemu Kerola, K2003

27

KJ-esimerkki: laiteajuri

- Prosessina
 - proseduurina kutsuttu laiteajurin tynkä (stub) lähettää I/O-pyyynnön viestinä laiteajuriprosessille ja odottaa vastausta
 - tynkä voi olla käyttäjätilainen
 - ajuriprosessi voi olla (joskus) etuoikeutettu
 - vaatii prosessien välistä viestintää



27/05/2004

Copyright Teemu Kerola, K2003

28

-- Jakson 8 loppu --

[Tane99]

```
// Open files for input and output.
inhandle = CreateFile("data", GENERIC_READ, 0, NULL, OPEN_EXISTING, 0, NULL);
outhandle = CreateFile("newf", GENERIC_WRITE, 0, NULL, CREATE_ALWAYS,
FILE_ATTRIBUTE_NORMAL, NULL);
```

```
// Copy the file.
do {
    s = ReadFile(inhandle, buffer, BUF_SIZE, &count, NULL);
    if (s > 0 && count > 0) WriteFile(outhandle, buffer, count, &ocnt, NULL);
    while (s > 0 && count > 0);
```

```
// Close the files.
CloseHandle(inhandle);
CloseHandle(outhandle);
```

Figure 6-40. A program fragment for copying a file using the Windows NT API functions. This fragment is in C because Java hides the low-level system calls and we are trying to expose them.

Lisää tietoa? KJ kurssit, RIO, HajJärj

27/05/2004

Copyright Teemu Kerola, K2003

29