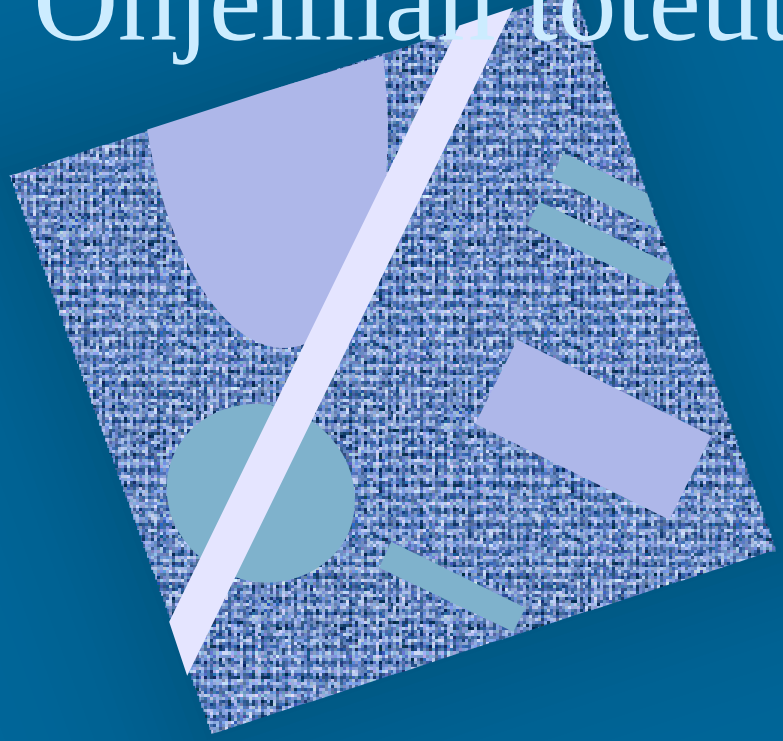
Jakso 8
Ohjelman toteutus järjestelmässä



Prosessi

PCB

I/O:n toteutus

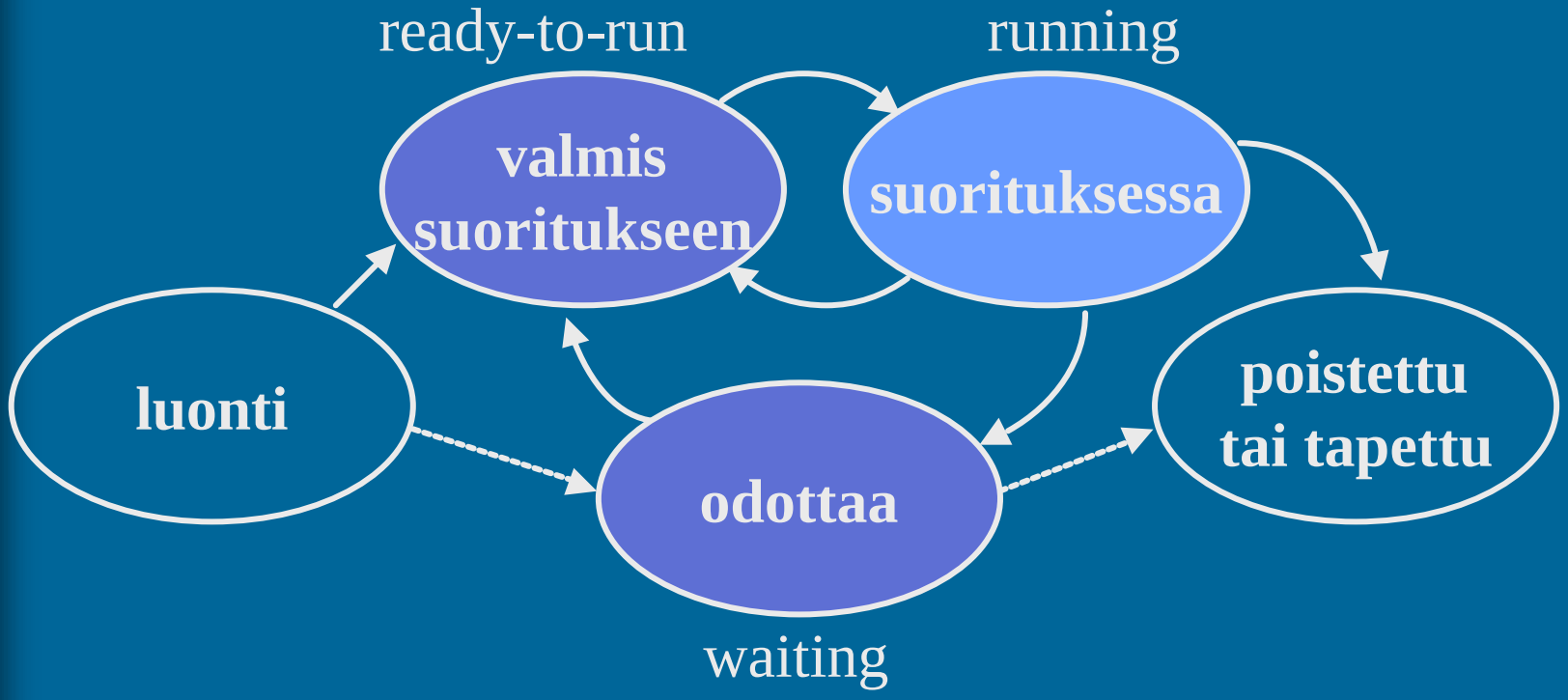
Prosessi (4)

- 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
  - suorinta, I/O:ta, viestiä toiselta prosessilta

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

Prosessin elinkaari (9)



Prosessin tilat järjestelmässä

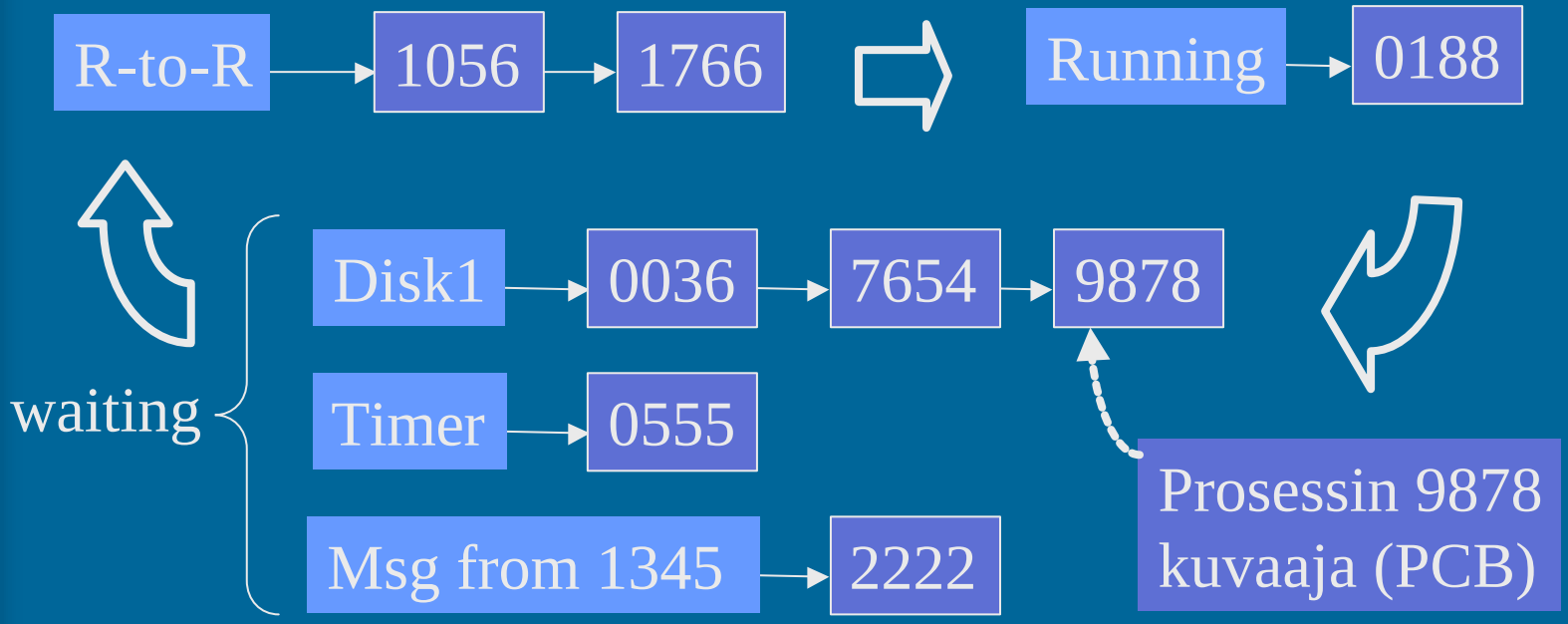
Prosessin esitysmuoto järjestelmässä (4)

- Prosessin kuvaaja tai 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
  - käyttöjärjestelmärutiinit manipuloivat PCB:tä

Prosessin tilanvaihdon toteutus (9)

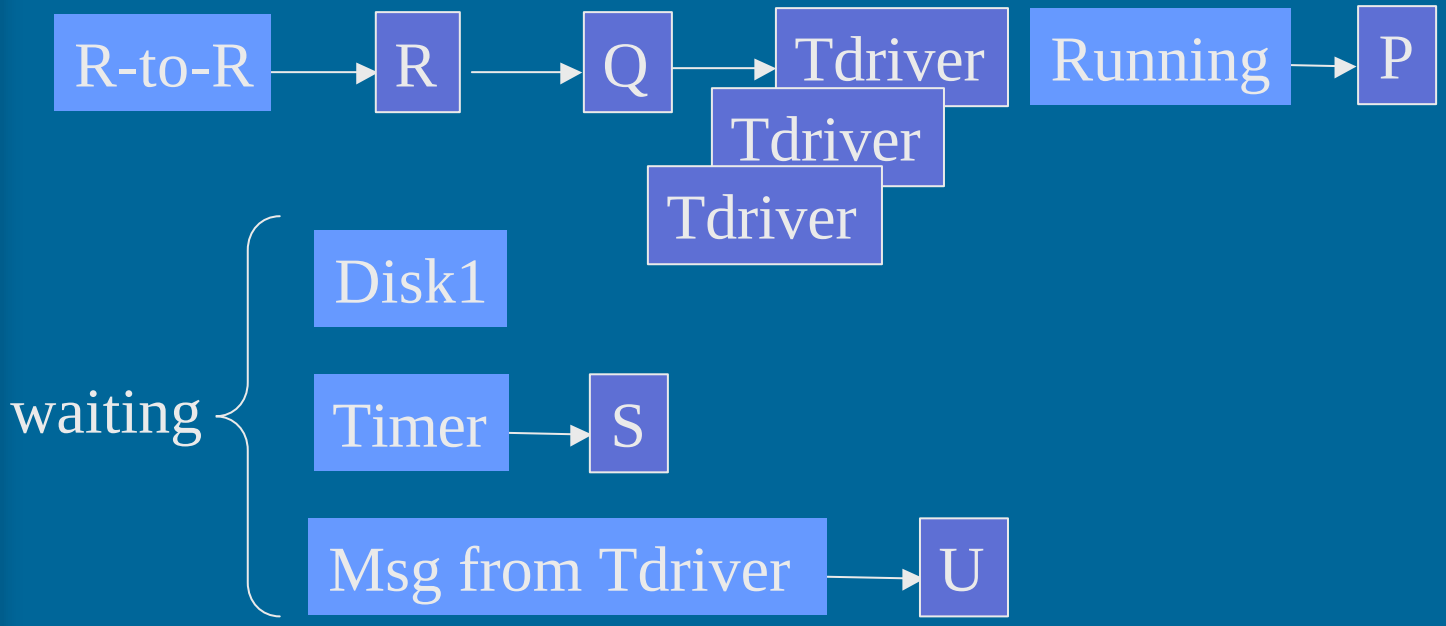
- Prosessin tilanvaihto tapahtuu siirtämällä prosessi (sen PCB) jonosta toiseen
  - ready-to-run jono (tai jonot)
  - running jono
    - jota ei oikeastaan ole olemassa
  - waiting jono
    - joka tyypille oma jononsa
    - esim: laitteen Disk1 I/O:n valmistumista odottavat
    - esim: näppäimistön painallusta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

Prosessit jonoissa (1)



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

KJ esimerkki: I/O keskeytys (5)



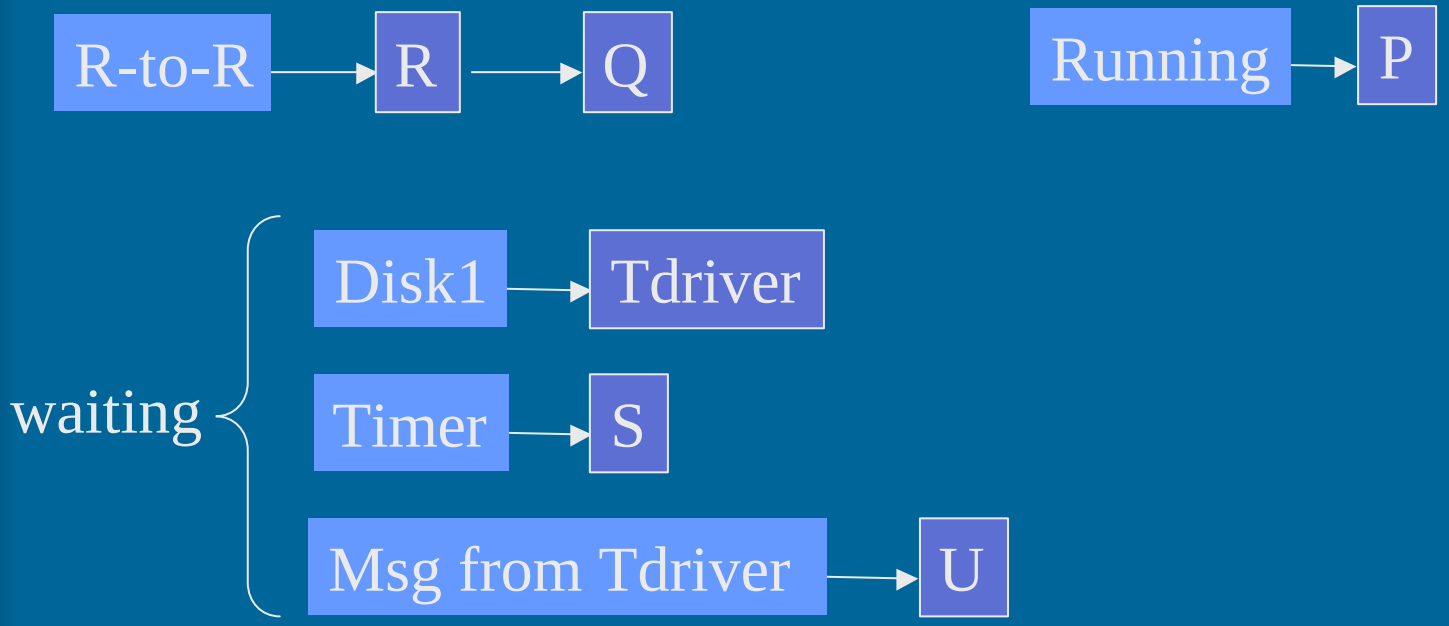
I/O keskeytys laitteelta Disk1 prosessille Tdriver?

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

Tdriver siirretään R-to-R jonoon

P:n suoritus jatkuu vai jatkuuko?

KJ esimerkki: I/O keskeytys (ei anim)



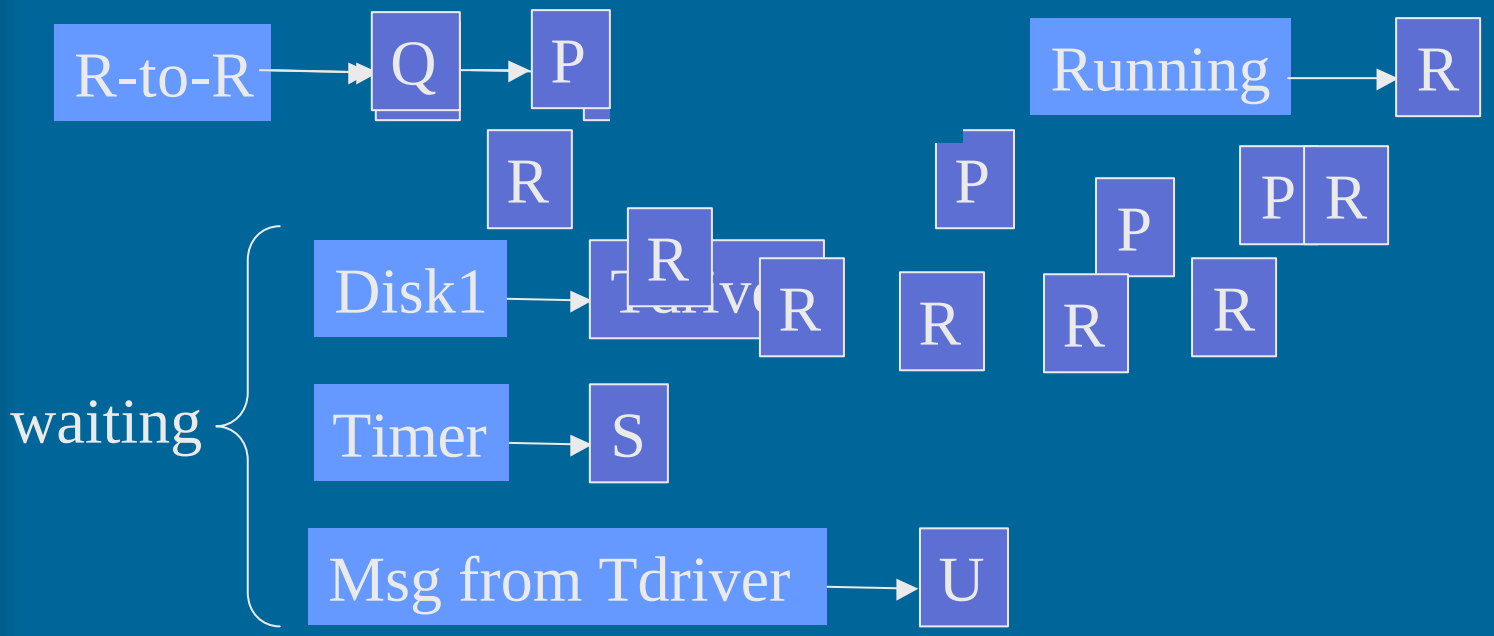
I/O keskeytys laitteelta Disk1 prosessille Tdriver?

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

Tdriver siirretään R-to-R jonoon

P:n suoritus jatkuu vai jatkuuko?

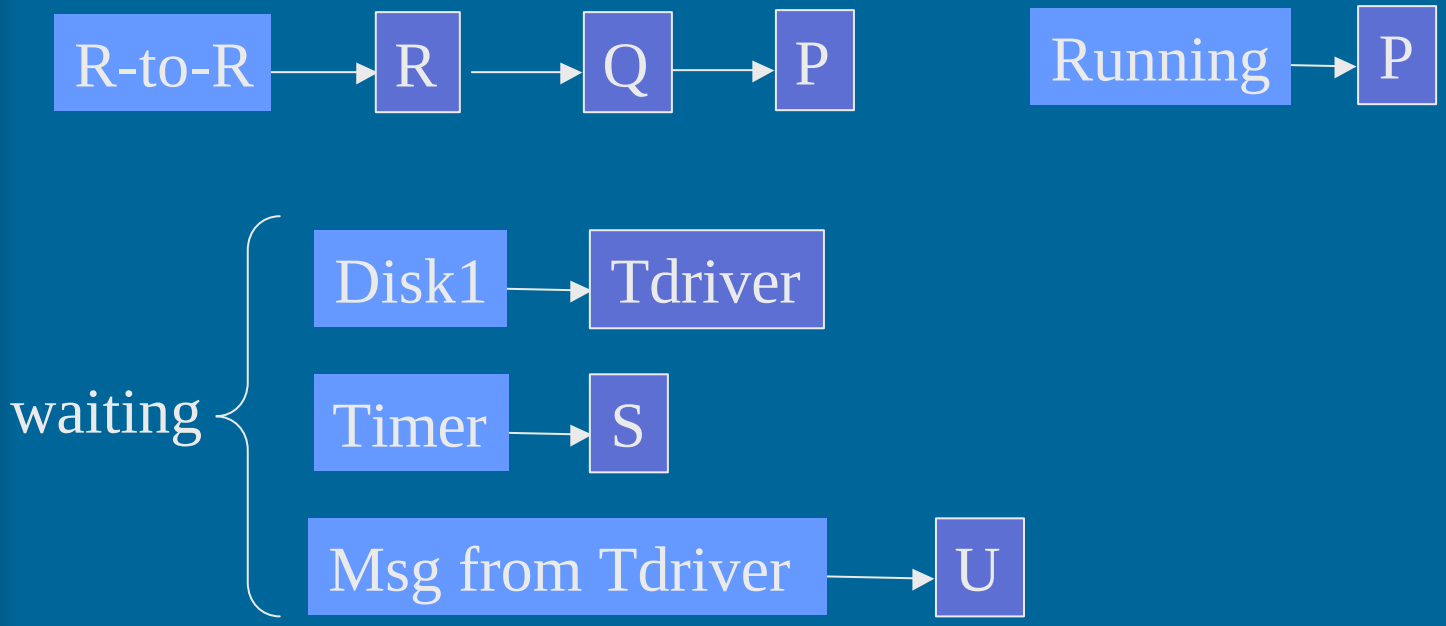
KJ esim: aikaviipalekeskeytys (7)



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?

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 Disk1:ltä?

Prosessin kuvaajan sisältö (9)

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

Prosessin vaihto (4)

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

Prosessin prioriteetti (3)

- Prosessin tärkeysjärjestys suorittimella
  - esim. pieni numero ⇒ iso prioriteetti
- Joka prioriteetti(luokalle) oma R-to-R jononsa
  - KJ prosesseilla parempi prioriteetti kuin käyttäjätason prosesseilla
  - tosiaikasovelluksen prosesseilla parempi prioriteetti kuin KJ prosesseilla
- Prioriteetti voi vaihdella prosessin elinaikana
  - paljon suoritinaikaa ⇒ huonompi prioriteetti
  - kauan R-to-R jonossa ⇒ parempi prioriteetti (prosessi siirretään korkeamman prioriteetin R-to-R jonoon)

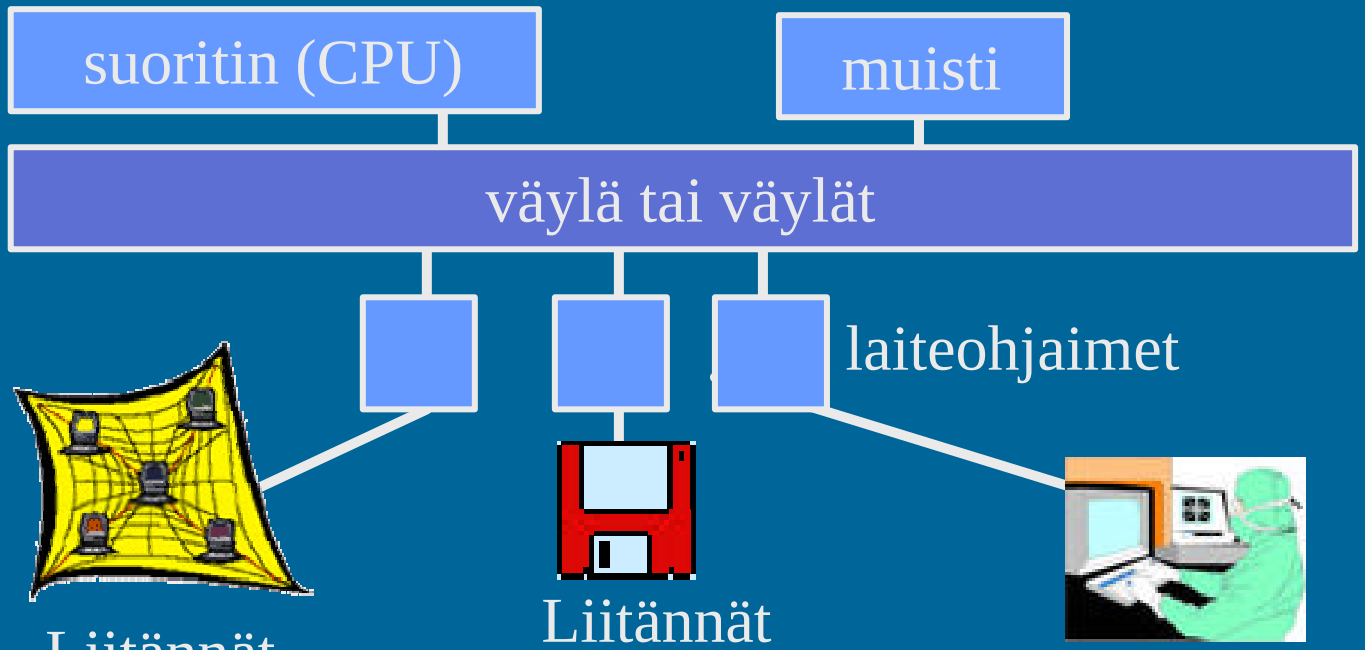
Käyttöjärjestelmän toteutus (5)

- 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?)
    - keskeytyskäsittelijät
  - Saavat kontrollin aina tarvittaessa
    - keskeytykset, SVC, ajastimet

KJ esimerkki: laiteajuri

- Prosessina
  - proseduurina kutsuttu laiteajurin tynkä (stub) lähettää I/O-pyynnön viestinä laiteajurille ja odottaa vastausta
    - tynkä voi olla käyttäjätilainen
    - ajuriprosessi voi olla (joskus) etuoikeutettu
    - vaatii prosessien välistä viestintää
  - Proseduurina
    - laiteajuri suoritetaan KJ-rutiinina tavallisen aliohjelmakutsun tai SVC-kutsun kautta
      - osa tai kaikki koodista voi olla etuoikerutettua
      - vain yksi kutsu kerrallaan suorituksessa?

Laitteiden liittäminen järjestelmään (4)



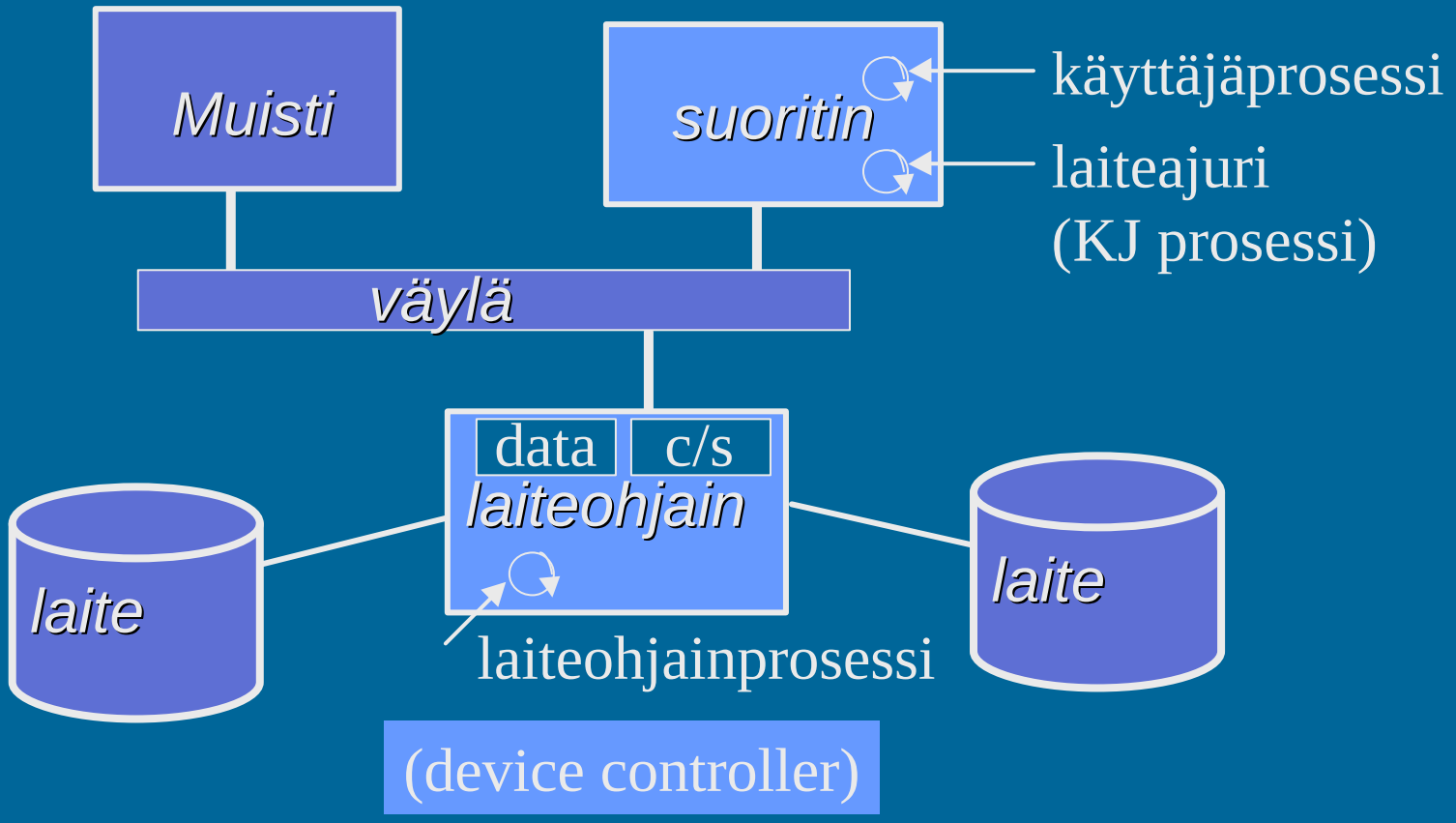
Liitännät muiden koneiden kanssa

Liitännät laitteiden kanssa

Liitännät ihmisten kanssa

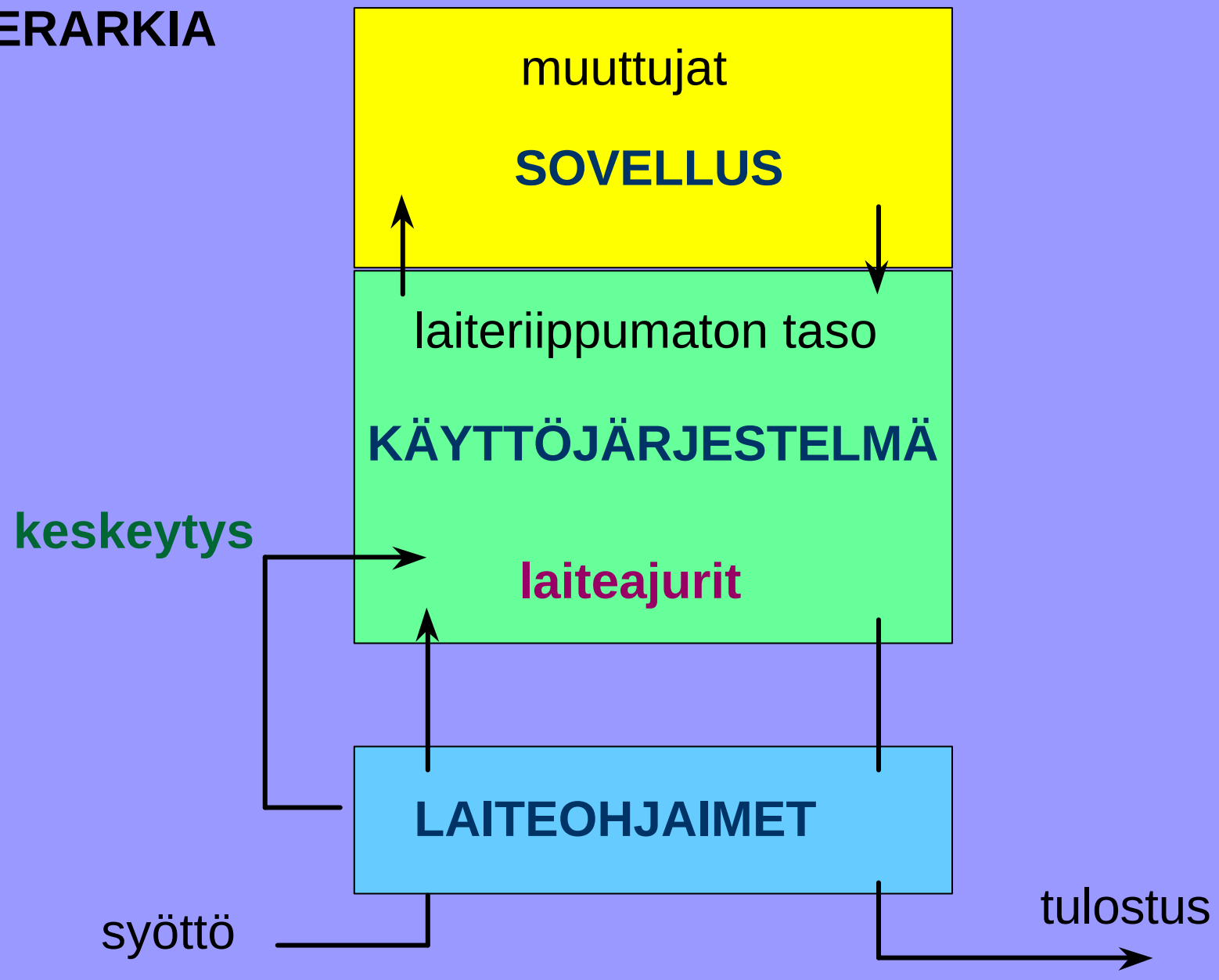
Kaikki samalla tavalla: ks. Fig. 2-29 [Tane99]

Laiteohjain (I/O Moduuli)



Ks. Fig 6.4 [Stal99]

SIIRRÄNNÄN HIERARKIA



- Siirrännän hierarkia
  - **sovellusohjelmataso**
    - loogisia kokonaisuuksia, tietueita ja tiedostoja
    - ohjelman sisäisiä nimiä
    - Readln (File1, X)
    - Open (Tdsto, RW)
  - käyttöjärjestelmätaso
  - laitteisto

- Siirrännän hierarkia
  - sovellusohjelmataso
  - **käyttöjärjestelmätaso**
    - rutiinit, jotka toteuttavat ja valvovat siirräntää
    - laiteriippumaton siirräntä
      - sovellukselle yhtenäinen tapa käyttää kaikkia siirräntäpalveluita
    - laiteriippuva siirräntä
      - laitteiden todelliseen käyttöön liittyvä ohjausohjelmisto
      - koodattu **laiteajureihin**
  - laitteisto

- Siirrännän hierarkia
  - sovellusohjelmataso
  - käyttöjärjestelmätaso
  - **laitteisto**
    - siirräntä voidaan toteuttaa kokonaan prosessorin valvonnassa
      - ei hyödynnetä rinnakkaisuutta
    - **laiteohjain** (siirräntään erikoistunut prosessori) huolehtii itsenäisesti siirrännästä
      - prosessorin ja ohjainten välinen kommunikointi

- laitekuvaaja
  - yksi kutakin laitetyyppiä varten
  - talletettavat tiedot riippuvat laitteesta
    - laitteen yksilöivä tunnus (väyläosoite)
    - ohjeet laitteen käytöstä
      - urien, sektorien ja levypintojen määrä, lohkon koko
    - viitteet näppäimistön merkinmuunnostauluihin
    - laitteen tilatietoa
      - varattu/vapaa/rikki
    - viitteet jonottaviin palvelupyyntöihin
    - viite laitetta käyttävän prosessin kuvaajaan

- laiteriippumattoman siirrännän tehtäviä
  - loogisesta tiedostonimestä => käytettävän laitteen tyyppi
  - pitää kirjaa levytilan vapaista ja varatuista alueista
  - siirrännän puskurointi (levylohko)
  - luku/kirjoituskohdan ylläpito
  - tarvittaessa käynnistää fyysisen siirrännän
    - antaa laiteajurille tehtävän

- laiteajurin tehtäviä
  - tehtävät riippuvat laitteesta
    - muodostaa parametrien ja laitekuvaajan perusteella laitetta ohjaavat käskyt
      - esim. levylohkonumeroiden muuttaminen levypinnan, uran ja sektorin numeroiksi
    - levypyyntöjen optimointi
    - ohjaimella tehtävän fyysisen siirrännän käynnistys
    - siirrännän kirjanpito
    - siirron oikeellisuuden tarkistukset ja virheiden korjausyritykset

Laitteiden käytön toteutus (5)

ks. laiteohjainkuva

- Käyttäjäohjelma kutsuu käyttöjärjestelmän laiteajuria tekemään I/O:n. Laiteajuri suoritetaan samalla suorittimella kuin käyttäjäohjelmakin.
- Laiteajuri ohjaa laitteen toimintaa laitteen laiteohjaimella olevien kontrollirekisterien (muistialue) avulla
- Laiteajuri voi lukea laitteen tilatietoa laiteohjaimella olevien statusrekisterien (muistialue) avulla
- Laiteajuri voi lukea (kirjoittaa) laitteen lukemaa (laitteelle kirjoitettavaa) tietoa laiteohjaimella olevien datarekistereiden (muistialue) avulla
- Kontrolli-, status- ja datarekisteri kolmikko muodostaa ”I/O portin” suorittimen näkökulmasta

Laiteohjaimen rekistereihin viittaaminen (5)

- Ongelma: miten suorittimella suorittavan laiteajuri viittaa eri kortilla oleviin rekistereihin?

ks. laiteohjainkuva

- Ratkaisu 1: omat I/O-konekäskyt tätä tarkoitusta varten
  - käskyssä annetaan laiteohjaimen identifikaatio ja rekisterin nro (I/O osoiteavaruus)
  - vaikea laajentaa käyttöä uusiin laitteisiin, joilla rekisterit voivat olla hyvinkin erilaisia

x86: IN, OUT
INS, OUTS

KOKSI: IN R1, =KBD,
OUT R2, =CRT

Ratkaisu 2: muistiinkuvattu I/O (5)

ks. laiteohjainkuva

- Laiteajuri lukee/kirjoittaa laiteohjaimella olevia rekistereitä (data, status/kontrolli) tavallisilla muistin luku/kirjoitus käskyillä
  - ei tarvita erillisiä I/O-konekäskyjä!
  - laiteohjaimella olevat ”rekisterit” ovat samanlaista viitattavaa muistia kuin ”normaali muisti”
  - muistisoitteen ensimmäiset bitit valitsevat, mille laitteelle (vai tavallisen muistiin) viittaus kohdistuu
  - voidaan käyttää rinnan I/O käskyjen kanssa (laiterekistereihin voi siis viitata sekä I/O-käskyillä että muistiinkuvatun I/O:n avulla)

esim. Intelin arkkitehtuurit

I/O tyypit (2)

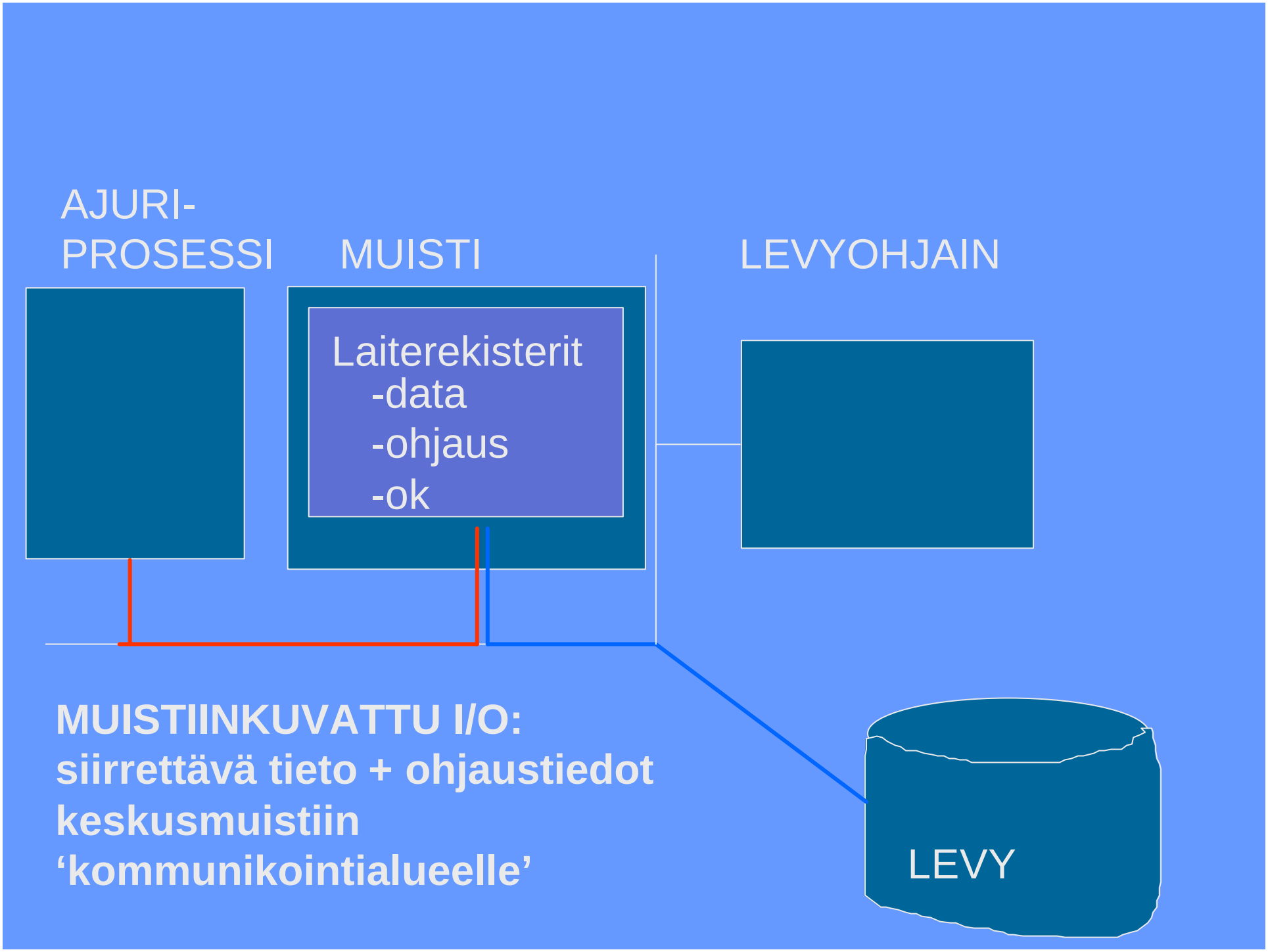
ks. laiteohjainkuva

- Suora I/O: laiteajuri odottaa tiukassa silmukassa, kunnes laiteohjaimen statusrekisteri ilmoittaa I/O-pyynnön valmistuneen (direct I/O)
  - laiteajuri siirtää tietoa muistin ja datarekisterin välillä
- Epäsuora I/O: I/O:n odotusaikana suorittimella suoritetaan jotain muuta ohjelmaa (indirect I/O interrupt driven I/O)
  - Kun I/O-pyyntö valmistuu, laiteohjain antaa keskeytyksen (laitekeskeytys, I/O interrupt) suorittimelle, joka (jonkin ajan kuluttua) jatkaa kesken jäänyttä I/O-pyynnön esittänyttä ohjelmaa.
  - laiteajuri siirtää tietoa muistin ja datarekisterin välillä

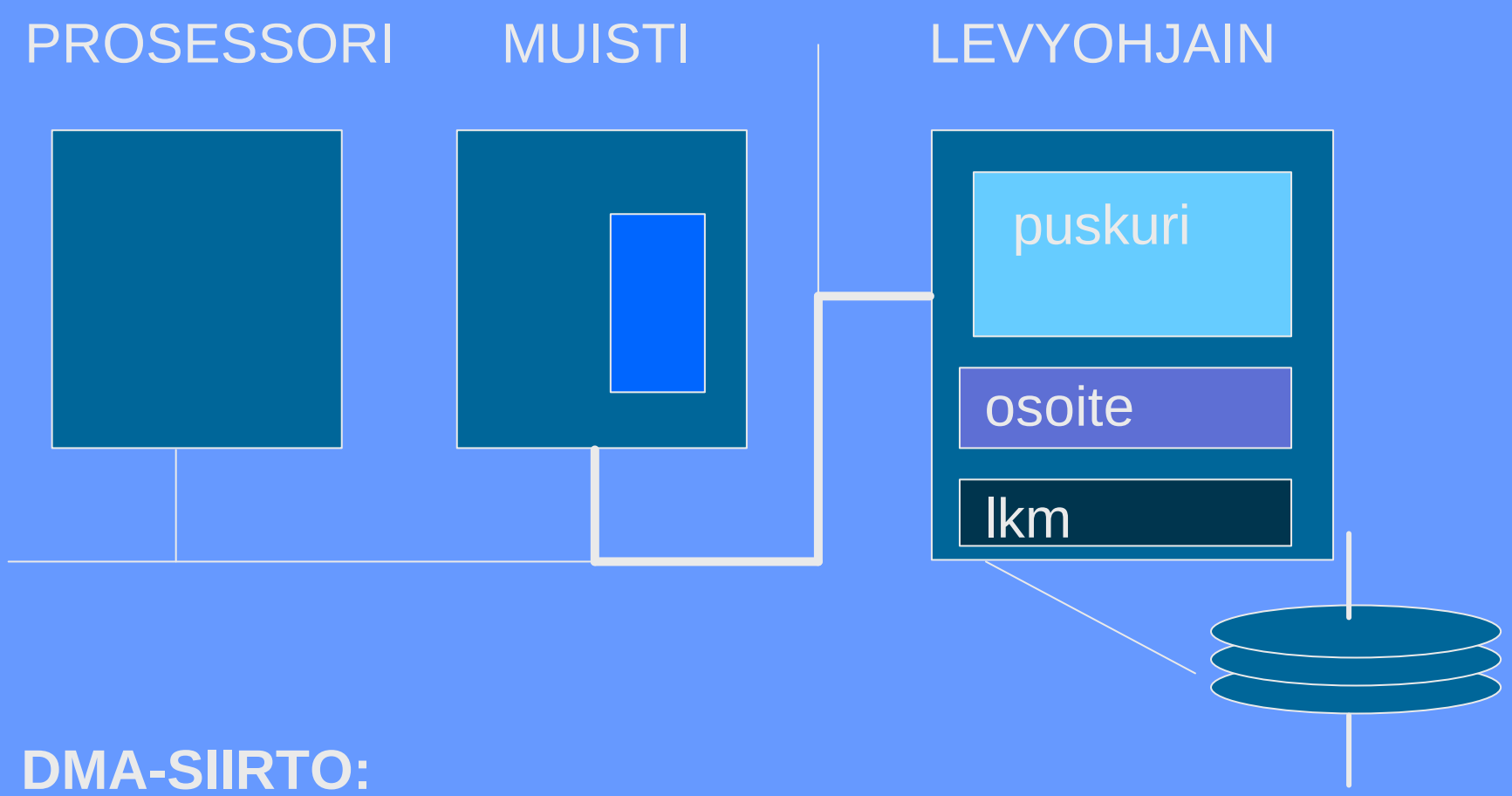
I/O tyypit (jatkoa) (4)

ks. laiteohjainkuva

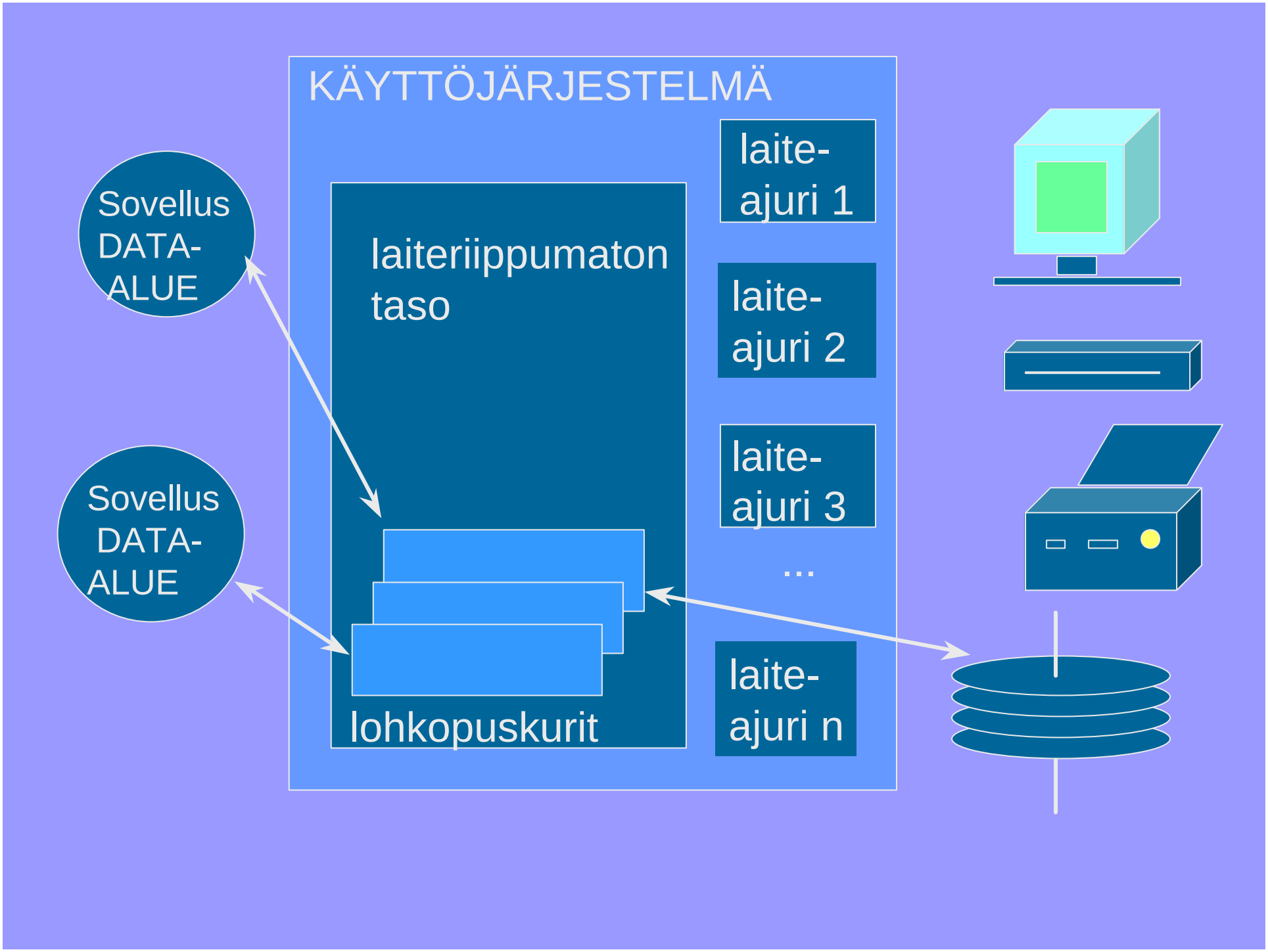
- DMA - Direct Memory Access
  - älykkäämpi laiteohjain
  - laiteohjain voi suoraan kopioida tiedot keskusmuistiin (laiteajurin ei tarvitse siirtää tietoa muistin ja datarekisterin välillä)
  - laiteohjain tekee paljon suuremman määrän työtä itsenäisesti (kuin epäsuorassa I/O:ssa) ennen suorittimelle annettavaa laitekeskeytystä

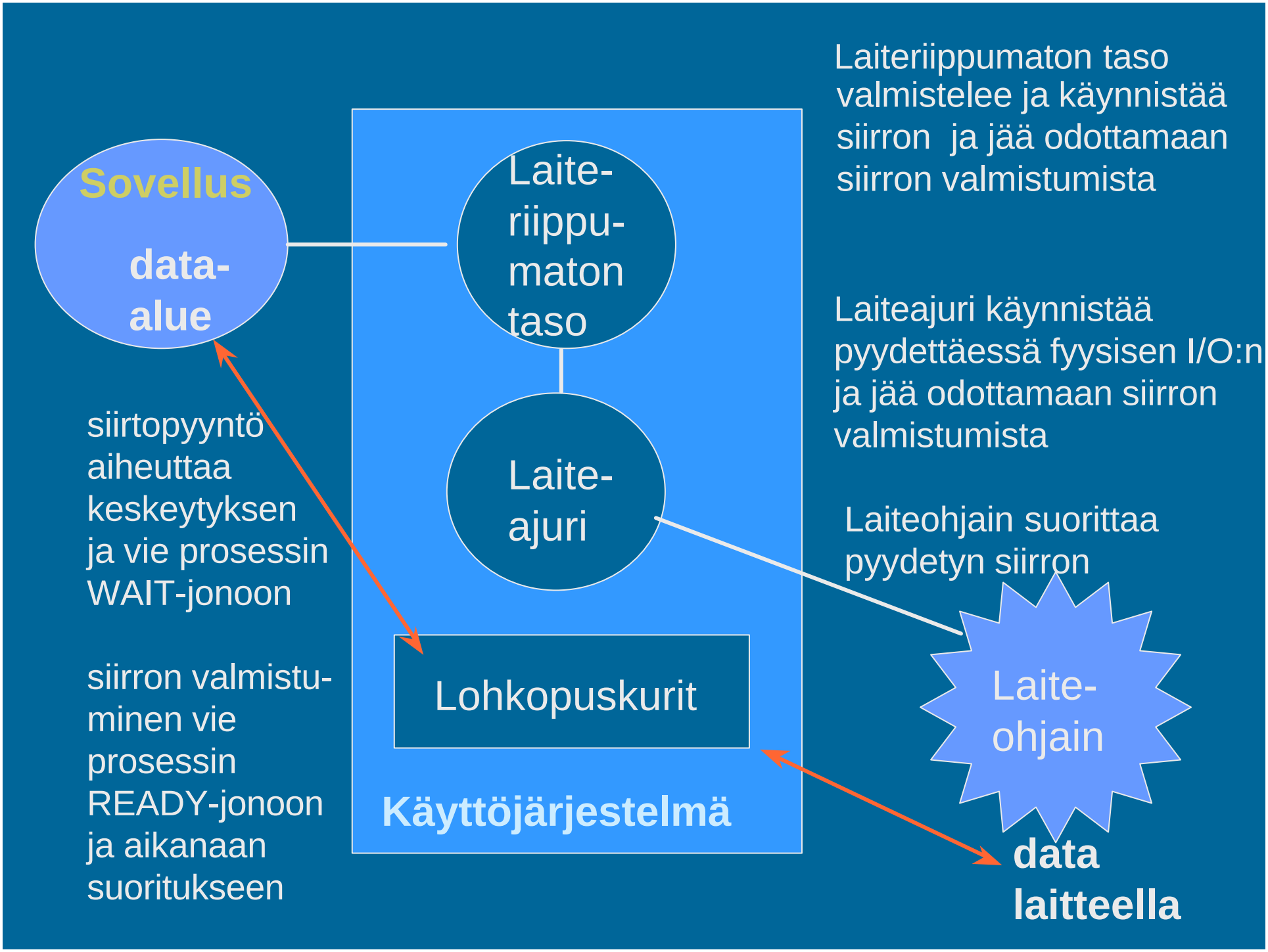


MUISTIINKUVATTU I/O:
siirrettävä tieto + ohjaustiedot keskusmuistiin ‘kommunikointialueelle’



DMA-SIIRTO:
ohjain siirtää itsenäisesti tietoa levyltä keskusmuistiin


KÄYTTÖJÄRJESTELMÄ
Sovellus DATA-ALUE
Sovellus DATA-ALUE
laiteriippumaton taso
lohkopuskurit
laite-ajuri 1
laite-ajuri 2
laite-ajuri 3
...
laite-ajuri n


Laiteriippumaton taso valmistelee ja käynnistää siirron ja jää odottamaan siirron valmistumista
Sovellus
data-alue
Laite-riippu-maton taso
Laite-ajuri
Laiteajuri käynnistää pyydettäessä fyysisen I/O:n ja jää odottamaan siirron valmistumista
siirtopyyntö aiheuttaa keskeytyksen ja vie prosessin WAIT-jonoon
Laiteohjain suorittaa pyydetyn siirron
Lohkopuskurit
Laite-ohjain
siirron valmistu-minen vie prosessin READY-jonoon ja aikanaan suoritukseen
Käyttöjärjestelmä
data laitteella

-- 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