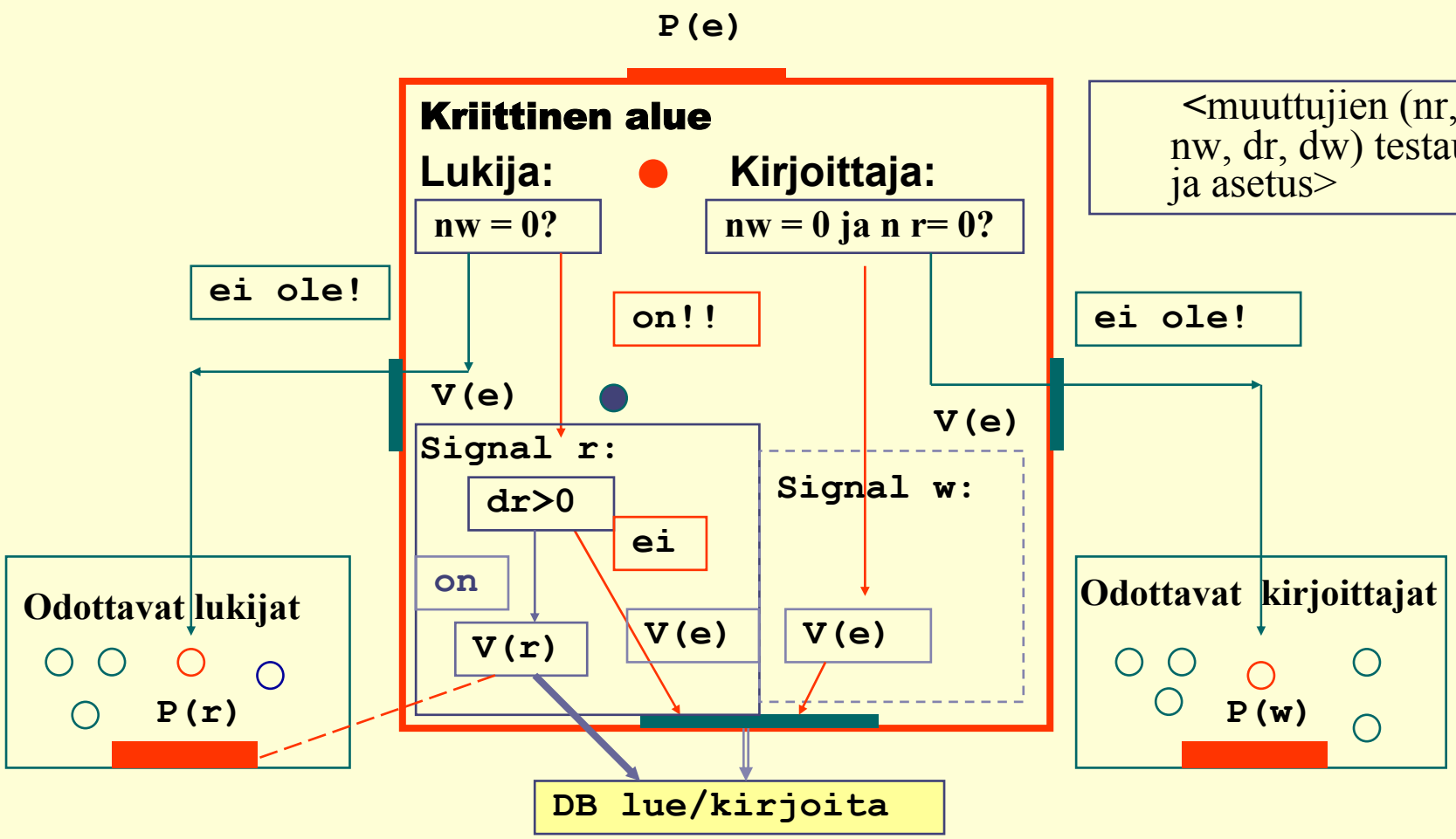
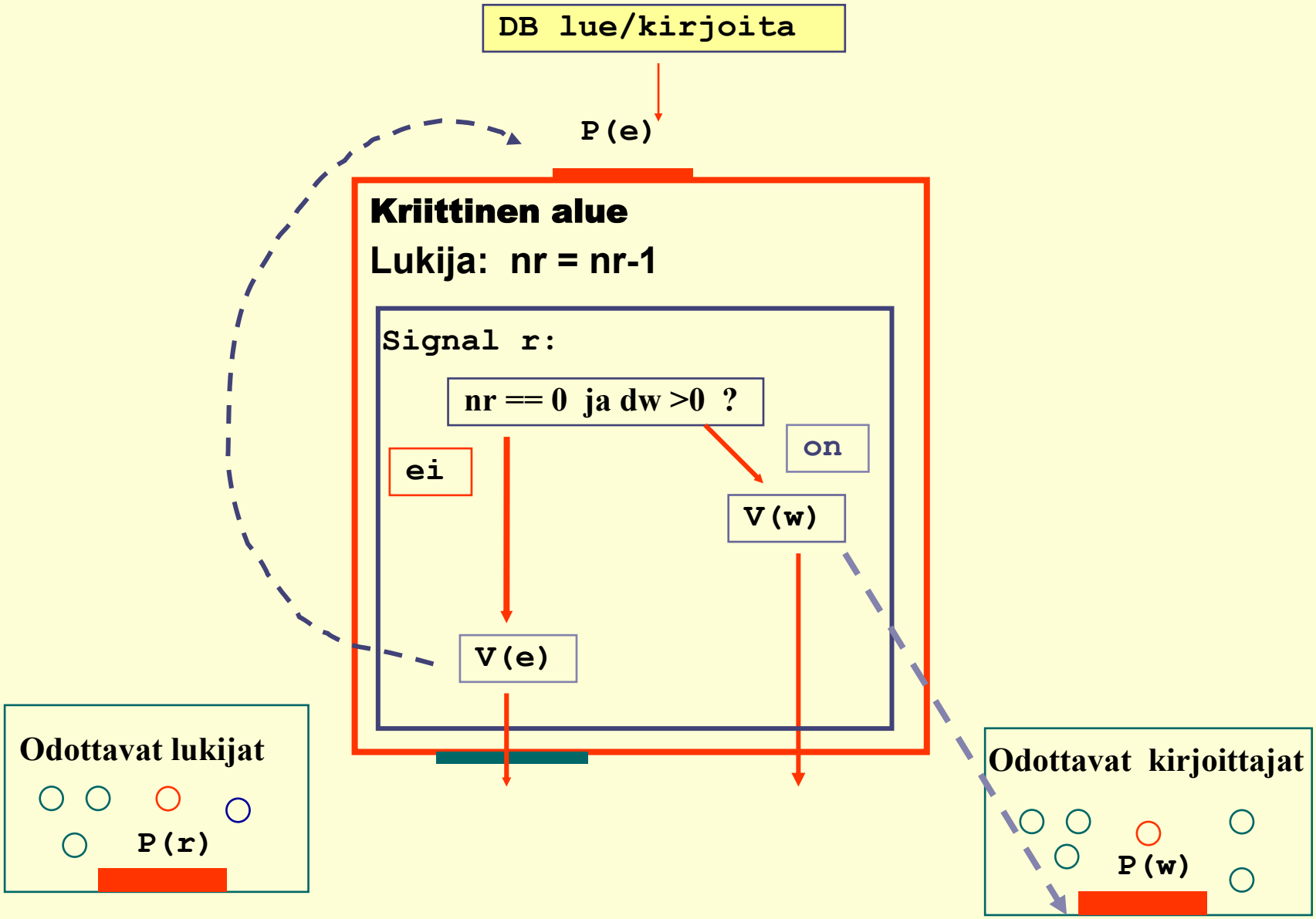
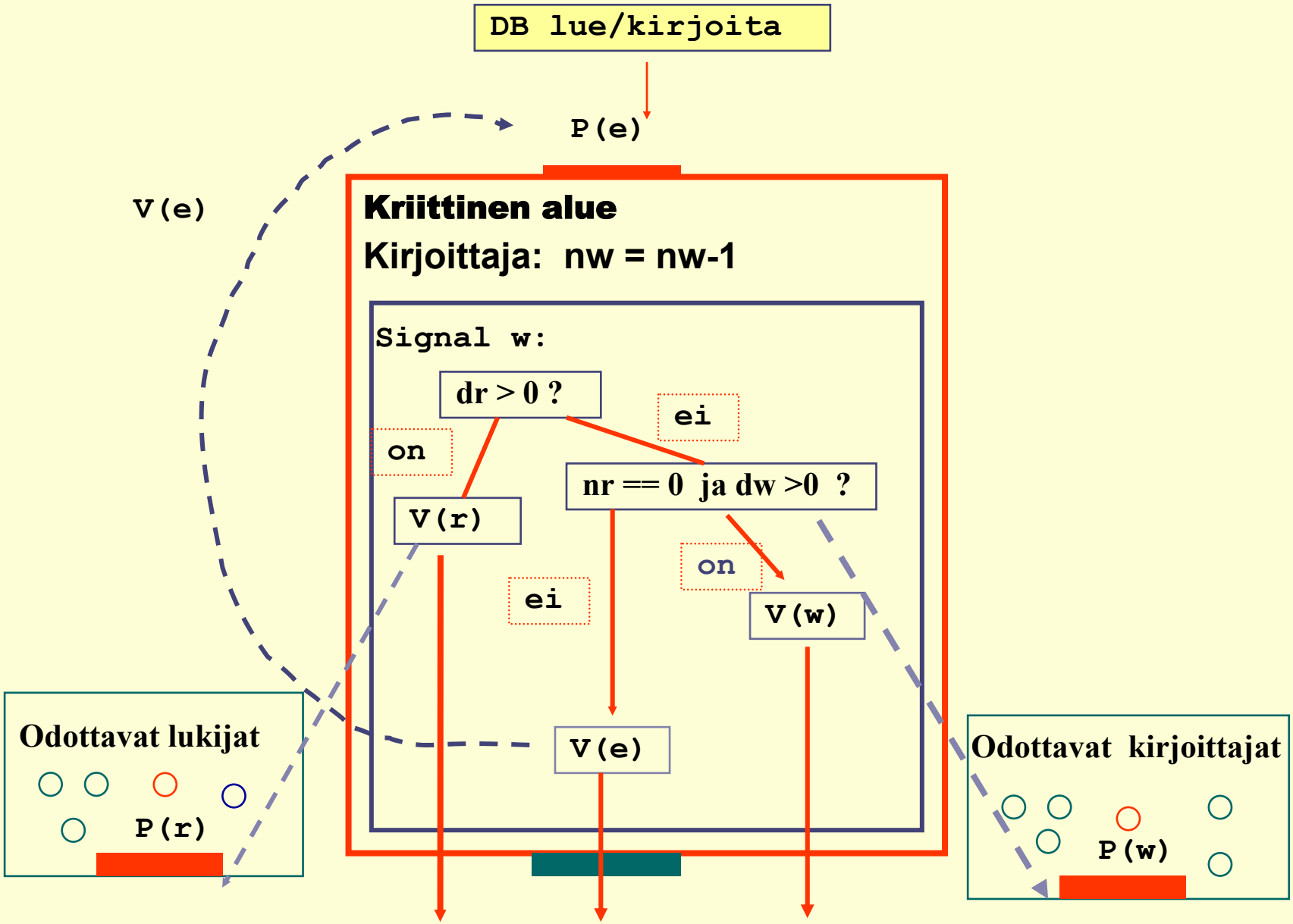

P(e)
Kriittinen alue
Lukija:
Kirjoittaja:
nw = 0?
nw = 0 ja n r= 0?
<muuttujien (nr, nw, dr, dw) testaus ja asetus>
ei ole!
on!!
ei ole!
V(e)
V(e)
Signal r:
Signal w:
dr>0
ei
on
V(r)
V(e)
V(e)
Odottavat lukijat
P(r)
Odottavat kirjoittajat
P(w)
DB lue/kirjoita


DB lue/kirjoita
P(e)
Kriittinen alue
Lukija: nr = nr-1
Signal r:
nr == 0 ja dw >0 ?
on
ei
V(w)
V(e)
Odottavat lukijat
P(r)
Odottavat kirjoittajat
P(w)


DB lue/kirjoita
P(e)
V(e)
Kriittinen alue
Kirjoittaja: nw = nw-1
Signal w:
dr > 0 ?
on
ei
V(r)
nr == 0 ja dw >0 ?
on
ei
V(w)
V(e)
Odottavat lukijat
P(r)
Odottavat kirjoittajat
P(w)

Yleisiä ohjeita semaforien käytöstä

- **Semaforeille on vain operaatiot: alustus, P(), V()**
- **Semaforin alkuarvo**
  - etenemislupien lkm/vapaiden resurssien lkm
  - kuinka moni prosessi voi tehdä P() -operaation joutumatta semaforin jonoon

  ⇨ tarkista semaforien oikeat alkuarvot!
- **Missä tilanteissa odotettava?**
  - kutakin tilannetta kohti oma semafori
  - prosessi odottamaan ⇨ P( )
- **Missä tilanteissa odottaminen päättyy?**
  - päästä odottaja etenemään ⇨ V( )

- **Onko P()-operaatioita kriittisen alueen sisällä?**
  - pääseekö joku varmasti vapauttamaan odotuksesta
- **Jos tarve tietää semaforin arvo tai onko joku jonossa**
  - ⇨ oma laskurimuuttuja!
  - muista sillekin poissulkeminen
- **Jos tarve priorisointiin** (semaforin jono FCFS!)
  - ⇨ ylläpidä omaa jonoa
  - jonon alkiossa prioriteetin (järjestyksen) määräävä arvo sekä yksityinen odotussemafori
  - ⇨ Nyt algoritmi voi rauhassa tutkia jonon tilaa ensin ja päättää sitten päästääkö jonkun etenemään

- **Resurssien varaamisen perussapluuna**

  P(entry)
  if (pyyntöön ei voi suostua) *DELAY*
  anna resurssi
  SIGNAL

- **Resurssien vapauttamisen perussapluuna**

  P(entry)
  palauta resurssi
  SIGNAL

- *SIGNAL:* toteuta haluamasi skedulointijärjestys
  - mikä odottavista prosesseista pääsee etenemään?
  - voiko uusi ilmaantua paikalle?