SDL

(Specification and Description Language)

SDL

(Specification and Description Language)

- ITU-T:n standardi (entinen CCITT) Recommendation Z.100
- Kehitetty erityisesti järjestelmäsuunnitteluun
- Formaalinen kieli (Mahdollistaa automaattisen syntaksin ja semantiikan tarkistukset)
- SDL:ää voidaan käyttää sekä toteutukseen että suunnitteluun
- Graafinen versio (SDL-GR) ja tekstiversio (SDL-PR)

Mihin SDL:ää voidaan käyttää

- Vuorovaikuttavat systeemit
- Reaaliaika-sovellukset
- Hajautetut systeemit
- Tietoliikennejärjestelmät ja protokollat

- Diskreetti systeemi
  - (Laajennettu) äärellinen tilakone, (extended) finite state machine ((E)FSM)
- Ei sovellu
  - Jatkuva-aikaisiin analogisiin systeemeihin (auton ohjaus)
  - Raskaasta laskentaa vaativiin sovelluksiin (Sääennustukset)

Mitä etua SDL:n käytöstä

- Rajapintojen määritys selkeä automaattisesti
- Määrittelystä voidaan vaiheittain siirtyä toteutukseen
  - Määrittely voidaan siirtää suoraan toteutuksen pohjaksi
  - Määrittely voidaan toteuttaa myös UML:llä
- Simulointi mahdollista jo työn alkuvaiheessa
- Diagrammit voidaan tulkita täsmällisesti formaalisuuden vuoksi

SDL on formaalinen kieli

- Systeemit määriteltävä täsmällisesti
- Voidaan käyttää järjestelmäkehityksen eri vaiheissa
  - Määrittely
  - Suunnittelu
  - Toteutus
  - Dokumentointi
  - Simulointi/tarkastaminen
  - OMT-mallinnus

SDL:n yleisiä ominaisuuksia

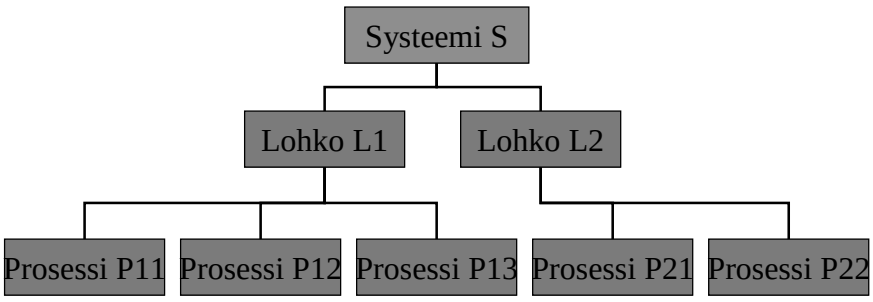
- SDL järjestelmä on malli todellisesta järjestelmästä
  - Järjestelmän systeemikomponenttien määrittely
  - Hierarkkinen malli (järjestelmä, lohkot, alilohkot, prosessit, proseduurit)
- Ei rajoita mallia
  - Ei absoluuttista toteutusta

SDL:n neljä peruselementtiä

- RAKENNE (structure) – Systeemi, lohko, prosessi ja proseduuri –hierarkia
- KÄYTTÄYTYMINEN (behavior) – Prosessien toiminnalliset ominaisuudet
- KOMMUNIKAATIO (communication) – Signaalit, reitit ja kanavat
- DATA – Abstract data types (ADT)

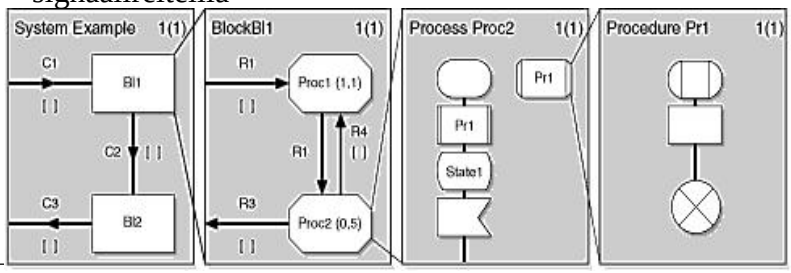
Rakenne (1)

- SDL sisältää neljä hierarkkia tasoa: systeemi (*system*), lohko (*block*), prosessi (*process*) ja proseduuri (*procedure*)



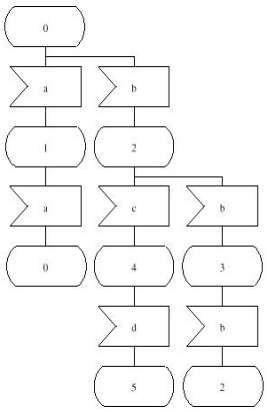
Rakenne (2)

- SDL-systeemin täytyy sisältää aikakin yksi lohko
  - Lohkon täytyy sisältää ainakin yksi prosessi tai alilohko
  - Lohkot ovat liitetty systeemin ympäristöön ja toisiin lohkoihin kanavilla
  - Prosessit ovat liitetty lohkon ympäristöön ja toisiin prosesseihin signaalireiteillä

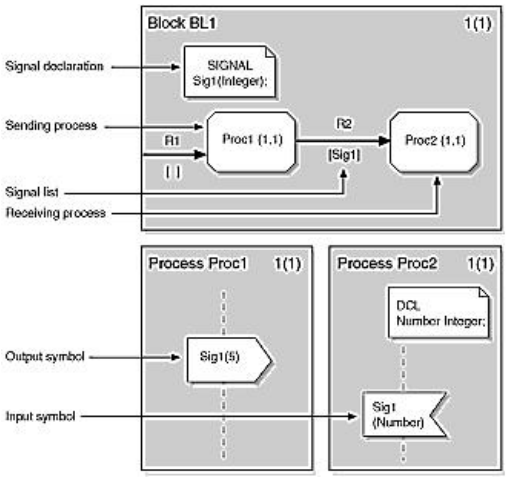


Käyttäytyminen (1)

- SDL:n dynaaminen käyttäytyminen määritellään prosesseissa
- Prosessi on laajennettu äärellinen tilakone, eli tilakone, joka voi lähettää signaaleita ja tallettaa tietoa automaatin lokaaleihin muuttujiin. Prosessin käyttäytyminen määritellään tiloilla ja tilasiirtymillä



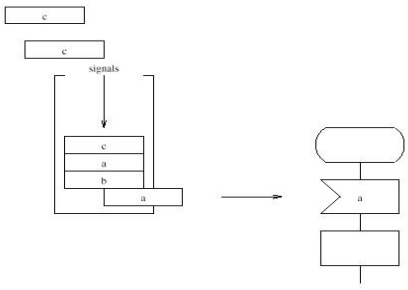
Kommunikointi (1)



- SDL:ssä on käytössä kaksi kommunikointitapaa
  - Asynkroniset signaalit
  - Synkroniset proseduurikutsut
- Kanavien ja reittien avulla määritetään selkeät rajapinnat lohkojen ja prosessien välille

Kommunikointi (2)

- Signaalireitiltä signaalit laitetaan prosessin Input-jonoon, jossa ne käsitellään tulojärjestyksessä
- Lähettäjäprosessi ei tiedä milloin signaali on käsitelty



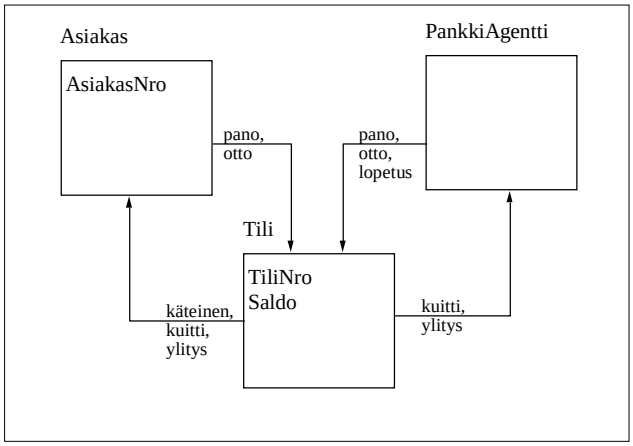
DATA

- Tietotyyppejä voidaan käyttää SDL:ssä kahdella periaatteella:
  - Sisäänrakennettujen piirteiden (tyyppien) käyttö, tavallisimmat muuttujatyypit ovat esimääriteltyinä SDL:ssä
  - Uusien tyyppien ja näiden operaattoreiden määrittely ja käyttö
    - Uusi tyyppi/operaattori(t) voidaan toteuttaa esim. periyttämällä esimääriteltytyyppi ja lisätä tarvittavat erikoispiirteet

Prosessit (Processes)

- Mallinnuksen peruskomponentti
- Prosessien ominaisuuksia
  - Attribuutit
  - Käyttäytyminen
- Prosessit kommunikoivat
  - Keskenään ja ympäristönsä kanssa

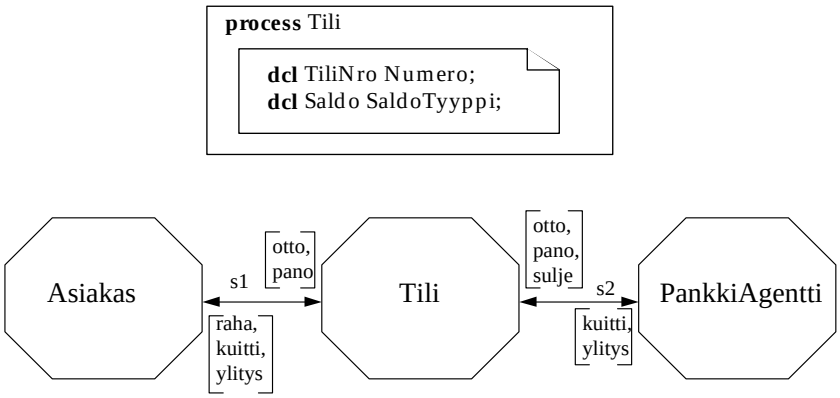
Esimerkki systeemin komponenteista



Prosessit

- Autonomisia osia
- Käyttäytyminen <==> signalointi
- Signalointi on asynkronista
- Kommunikointi määritellään
  - Signaaleilla
  - Signaalilistoilla
  - Signaalireiteillä

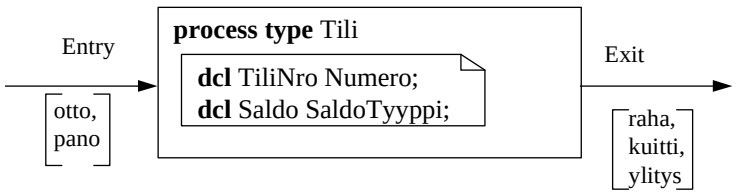
SDL prosessidiagrammi



Prosessityypit

- Komponenteilla samat ominaisuudet
  - Arvot voivat olla erilaiset
- Prosessityypit <==> olioluokat
  - Määrittelee prosessi ilmentymien yleiset ominaisuudet
  - Sisältää muuttujia (*variables*), aliohjelmia (*procedures*), ja käyttäytymisen (*behaviour*)
- Kommunikointiportit (*communication gates)*

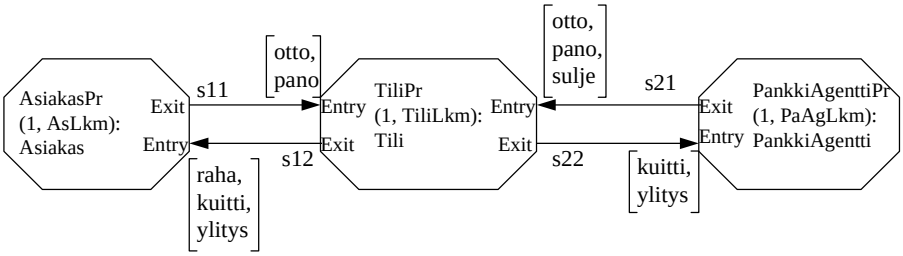
Prosessityypit



Prosessijoukot

- SDL on enimmäkseen prosessi-ilmentymien (*instancies*) ja niiden vuorovaikutuksien määrittelyä
- Prosessi ilmentymä on prosessijoukon jäsen
- Määrittely sisältää
  - Joukon (*set*) nimen
  - Ilmentymien lukumäärän
  - Prosessityypin nimen

Prosessijoukot

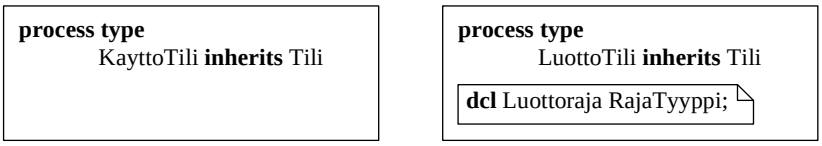


Prosessisotku (?)

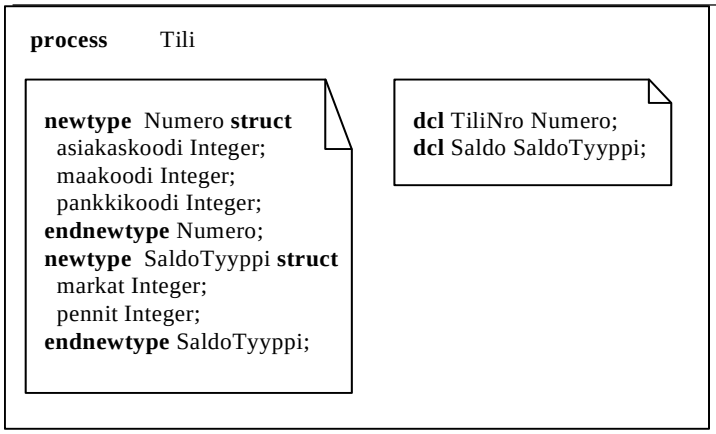
- Selviä käsitteellisiä eroja
  - *Tyypit* määrittelevät *ilmentymien* yleiset ominaisuudet
  - *Joukko* muodostuu yhdestä tai useammasta *ilmentymästä*
  - Signaalireitit yhdistävät *joukkoja*, eivät *ilmentymiä*
  - *Ilmentymällä* on attribuutit ja käyttäytyminen
  - Kaikilla ilmentymillä samat muuttujat joiden arvot voivat vaihdella

Prosessien alityypitys

- Peritään kuten objektit luokat (C++ object class)



Esimerkki prosessin attribuuteista



Esityksen lopussa lisää tiedon esittämisestä SDL:ssä

Käyttäytymisen määrittely

- Laajennettu äärellinen tilakone, extended finite state machine (EFSM)
  - Käyttää muuttujia “tila-avaruuden” laajennuksena
- Prosessin käynnistäminen
  - Käynnistyssymboli
  - (Tehtäväsymboli)
  - Tilasymboli
    - Ensimmäinen todellinen tila

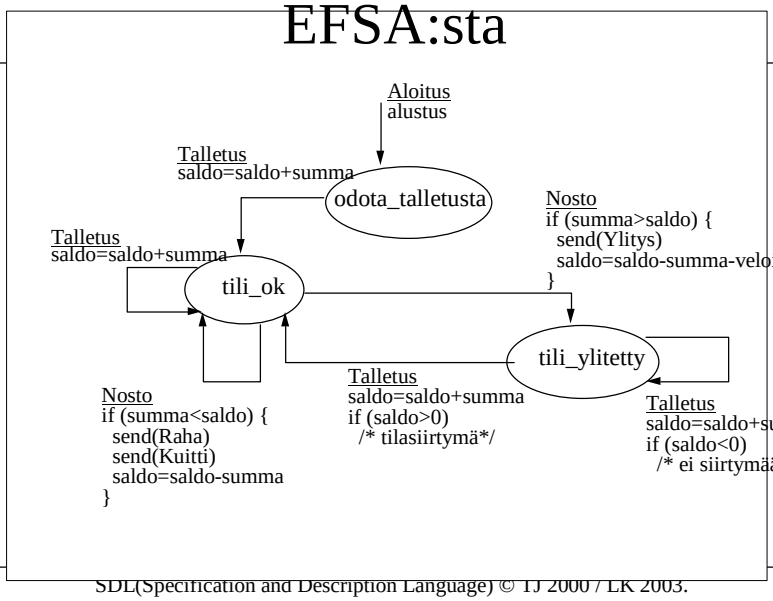
Käyttäytymisen määrittely

- Tilan muutokset
  - Input-symboli
    - Signaali toiselta prosessilta
    - ulkoinen proseduurikutsu (*remote procedure call*)
  - Toiminnot
    - Tehtävät, output-symbolit, proseduurikutsut jne.
  - Uusi tila
    - Tilan muutos tai jääminen samaan tilaan

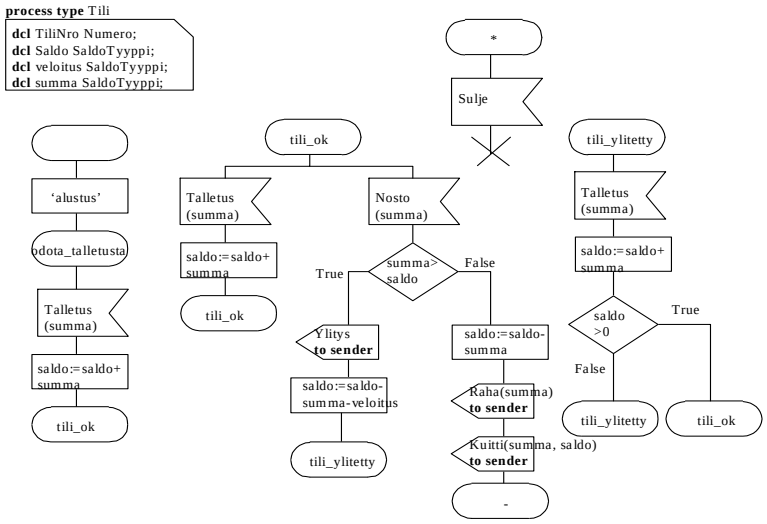
Käyttäytymisen määrittely

- Kontrolli rakenteet
  - Ehto symboli
    - Vertailu => päätös
  - In- and out-kytkennät
  - Makrokutsut
  - Mahdollistavat ’normaalit’ ohjelmointi rakenteet SDL:ssä

Esimerkki yksinkertaisesta EFSA:sta



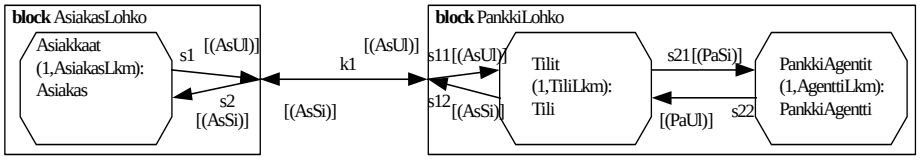
SDL:n tilakonediagrammi



Prosesseista lohkoihin

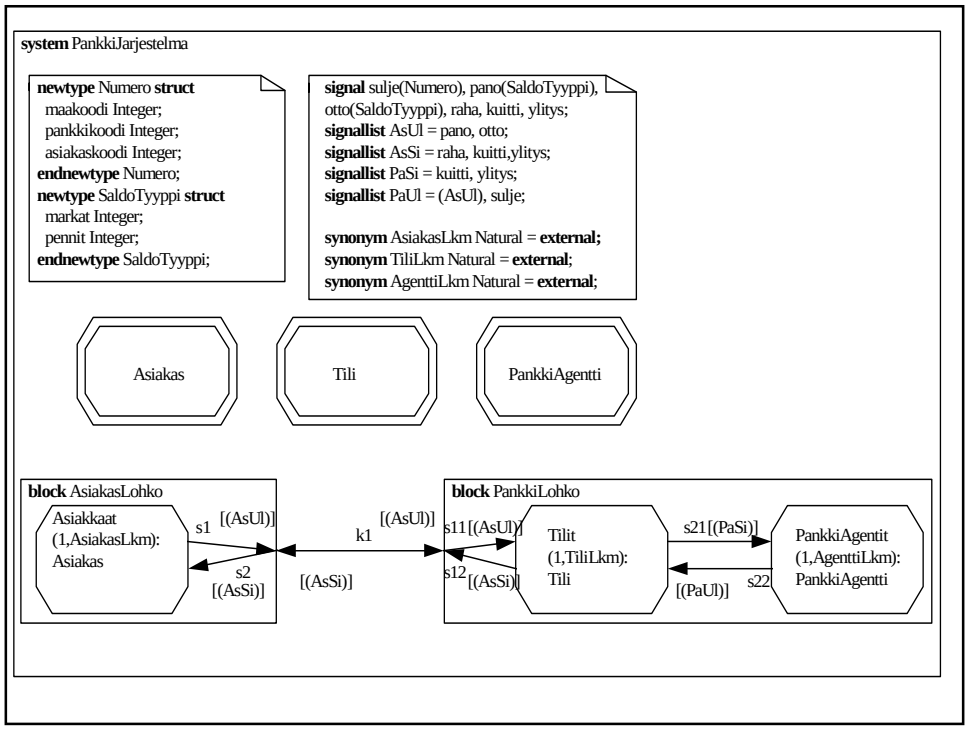
- Hierarkkiset rakenteet
- Lohkot (*blocks*) yhdistetään kanavilla
  - Aivan kuten prosessit signaali reiteillä
- Ei määriteltyä käyttäytymistä
  - Abstraktit rakenteet, mustat laatikot
- Sisältää yleensä signaali- ja datatyyppi määritykset
  - Paikallinen nimiavaruus

SDL lohkokaavio



Lohkoista järjestelmiin

- Lohkot yhdistetään kanavilla
- Kanavat myös ympäristöön
- Myös ympäristöllä on prosesseja
  - Niitä ei ole määritelty SDL:ssä
- Ympäristö valvoo ja kontrolloi systeemiä
  - Käyttöliittymä, näppäimistö, jne



Yhteenveto

- *Järjestelmä/Systeemi* muodostuu
  - *Lohkoista*, jotka yhdistetään signaali *kanavilla*
  - Joukosta *lohkoja*
- *Lohko* muodostuu
  - *Prosesseista*, jotka yhdistettään signaali *reiteillä*
  - *Ali-lohkoista*, jotka sisältävät ali-lohkoja ja prosesseja

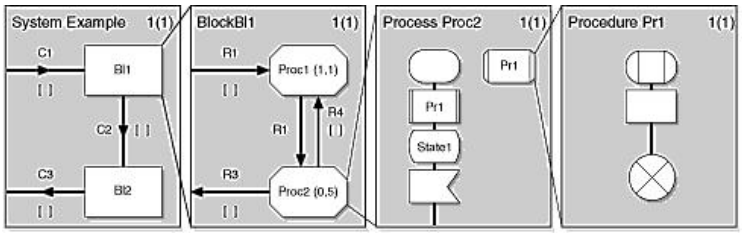
Yhteenveto

- *Prosessilla* on
  - Attribuutteja, jotka ovat määritelty *muuttujilla* ja *ulkoisilla aliohjelmilla*
  - Käyttäytyminen, joka on määritelty *EFSM:llä*, *tiloin (states)* ja *siirtymin (transitions)*

Yhteenveto

- *Tyypitystä* voidaan käyttää
  - Systeemeihin, lohkoihin ja prosesseihin
- *Tyyppejä* voidaan käyttää
  - määrittelemään *ilmentymäjoukkoja (instance sets)* ja *ilmentymiä (instances)*
  - määrittelemään ali-tyyppejä *periyttämällä (inheritance)*
- Myös Muuttujia/tietotyyppejä (ADT) voidaan tyypittää

Yhteenveto



SDL:n komponenttien hierarkkinen suhtautuminen toisiinsa

Tiedon esittäminen SDL:ssä

- Tietotyyppejä voidaan käyttää SDL:ssä kahdella periaatteella:
  - Sisäänrakennettujen piirteiden (tyyppien) käyttö, tavallisimmat muuttujatyypit ovat esimääriteltyinä SDL:ssä
  - Uusien tyyppien ja näiden operaattoreiden määrittely ja käyttö
    - Uusi tyyppi/operaattori(t) voidaan toteuttaa esim. periyttämällä esimääriteltytyyppi ja lisätä tarvittavat erikoispiirteet

Tietotyyppien määrittely

- Kaikki SDL-tietotyypit ovat abstrakteja *(abstract data types, ADTs),* mikä tarkoittaa jokaisen tietotyypin jakoa rajapinta ja käyttäytymisosaan
  - Rajapintaosassa luetellaan tyypin literaalit ja operaattorit
  - Käyttäytymisosassa määritellään literaalien ja operaattoreiden semantiikka
- ADT voidaan määritellä myös SDL:n ’ulkopuolella’ esim. C-kielen avulla

Tietotyyppien määrittely

- Literaalit (*literals)* määrittelevät tietotyypin arvojen nimet esim. kokonaislukutyypin literaaleja ovat mm. 0, 1, 2, 3, ...
- Operaattorit (*operators*) toimivat kuten funktiot, mutta eivät aiheuta muutoksia argumenteilleen esim. kokonaislukutyypille määritellyt operaattorit +, - ja *
- Seuraavassa lyhyesti keskeisimmät SDL:ssä esimääritellyt (abstraktit) tietotyypit

Esimääritellyt tietotyypit

- *integer*
  - kokonaislukumuuttuja
  - määritelty välillä -∞ –> ∞
  - normaalit operaatiot kuten esim. +, - ja *
- *natural* = **positiiviset** kokonaisluvut
- *boolean*
  - voi saada vain arvot *true* tai *false*
  - käyttää operaattoreita kuten esim. **not, or, xor** ja **and**

Esimääritellyt tietotyypit

- *real*
  - reaalilukumuuttuja
  - määritelty välillä -∞ –> ∞
  - normaalit operaatiot kuten esim. +, - ja *
- *character*
  - ASCII-merkkityyppi
  - voi saada arvoja, kuten ’A’, ’#’, CR, LF ja FF
  - myös vertailu operaattoreita kuten < tai >= voidaan käyttää

Esimääritellyt tietotyypit

- *charstring*
  - jonomuuttuja
  - voi sisältää merkkijonon kuten ’Hello’
  - Operaattoreina mm.
    - last(’Hello’) palauttaa *character*-merkin ’o’
    - length(’Hello’) palauttaa *integerin* 5
    - *’Hel’//’lo’* palauttaa *charstring*-jonon ’Hello’ (katenointi)
    - substring (’Hello’,2,3) palauttaa *charstring*-jonon ’all’

Esimääritellyt tietotyypit

- PId (*Process Identity*) eli prosessin tunnus
  - yksilöi SDL-prosessin yksikäsitteisesti
  - voi saada seuraavat arvot:
    - **self**, joka viittaa prosessiin itseensä
    - **sender,** joka viittaa viimeisimmän signaalin lähettäneeseen prosessiin
    - **parent**, joka viittaa prosessin luoneeseen ilmentymään
    - **offspring**, joka viittaa ilmentymään, jonka tämä prosessi viimeksi loi

Esimääritellyt tietotyypit

- **Self** ja **parent** tunnisteet ovat (luonnollisesti) vakioita koko prosessi-ilmentymän elinajan
- **Sender** ja **offspring** tunnisteiden arvot voivat muuttua prosessin suorituksen edetessä
- On myös mahdollista luoda PId-tyyppinen muuttuja johon tunnisteen arvo voidaan sijoittaa myöhempää käyttöä varten

Esimääritellyt tietotyypit

- *Time*
  - muuttujan arvo esittää ajanhetkeä todellisessa maailmassa
  - operaattoreina mm. -, + ja <
  - *now*-avainsanalla saadaan nykyhetki sijoitettua muuttujaan
- *Duration*
  - muuttujalla ilmaistaan tapahtuman kestoa
  - muuttujan arvo=kahden *time* muuttujan arvojen erotus

Esimääritellyt tietotyypit

- Aikamuuttujiin liittyy myös käsite ajastin, *Timer*
- Ajastimella saadaan ajastettua aikaväli, jonka jälkeen ajastin lähettää nimisensä signaalin
- Ajastin asetetaan tehtäväsymbolissa seuraavasti: SET (now + 2.5, MyTimer)
- Ajastin laukeaa 2.5 aikayksikön kuluttua ja lähettää signaalin MyTimer

Tietotyypin määritys

- Esimerkkinä Bool-tietotyypin määritys:

```
newtype Bool
literals true, false;
operators
   not : Bool -> Bool;
axioms
   not(true) == false;
   not(false)== true;
endnewtype Bool;
```

Tietotyyppien perintä

- Tietotyyppien perintä on mahdollista SDL:ssä.
- Perittävän tietotyypin ominaisuuksista voidaan valita, joko kaikki tai uuden tyypin toiminnalle välttämättömät
- Ominaisuuksia voidaan myös tarvittaessa lisätä

```
newtype Newinteger inherits Integer
    operators all;
endnewtype;
```


Tietotyypin ”uudelleen” nimeäminen

- SDL:ssä on mahdollista luoda tietotyyppejä, jotka ovat perustyypin kanssa täysin yhtenäisiä
- Uuden tyypin ominaisuuksia voidaan kuitenkin myös tarvittaessa rajata
- Käytetään esim. taulukoiden indeksoinnissa

```
syntype Smallint = Integer
    constants 0:10;
endsyntype;
```

Parannellut tyypit

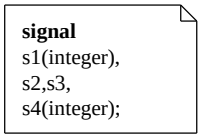
- Paranneltuina tyyppeinä SDL:ssä on mm.
  - Struct, jolla määritellään mm. C-kielen tavoin muuttujarakenteita (käyttö ks. dia numero 26)
- Generaatorit
  - Array
  - Powerset
  - String
- Esimerkiksi Array-generaattorilla luodaan SDL:ään taulukko rakenteita

SDL/SDT-symbolit

- Systeemin käyttäytyminen määritellään käyttäen yleensä piirrossymboleita (SDL-GR), mutta myös tekstipohjainen määritys mahdollinen (SDL-PR)
- Nyt tarkastellaan SDL-GR esitystä
- Seuraavassa yhteenveto yleisimmistä määrityksessä käytettävistä symboleista

Systeemitason symbolit

- Teksti *Text*
  - Sisältää määrityksiä esim. signaaleista ja abstrakteista muuttuja tyypeistä
  - Systeemissä voi olla useita teksti laatikoita, jotka ovat tasa-arvoisia
  - Määritykset voimassa kaikkialla systeemissä



Systeemitason symbolit

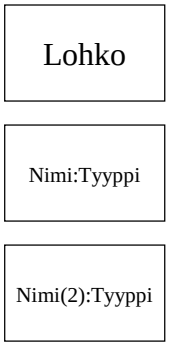
- Kommentti *Comment*
  - ’Laatikkoon’ voidaan sijoittaa ohjelman suoritukseen vaikuttamaton kommentti

Systeemitason symbolit

- Tekstin laajennus *Text extension*
  - ’Laatikkoon’ voidaan sijoittaa varsinaiseen symboliin mahtumaton koodi

Systeemitason symbolit

- LOHKO *Block Reference*
  - Määrittää lohkon osaksi systeemiä
  - Ei sisällä varsinaista lohkon määrittelyä
  - Systeemissä voi olla useita lohkoja myös sisäkkäin
  - Voi määrittää myös abstraktin lohkon käyttöön oton

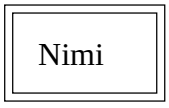


Systeemitason symbolit

- Proseduuri viittaus *Procedure Reference*
  - Sisältää viittauksen toisaalla määriteltyyn aliohjelmaan
  - Jos viittaus systeemitasolla myös kaiki lohkot ja prosessit tässä systeemissä voivat käyttää kyseitä proseduuria

Systeemitason symbolit

- Lohkotyyppi viittaus *Block Type Reference*
  - Sisältää viittauksen toisaalla määriteltyyn lohkotyyppiin
  - Jos systeemissä käytetään lohkotyypin ilmentymiä on siinä oltava myös viittaus käytettävään lohkotyyppiin



Systeemitason symbolit

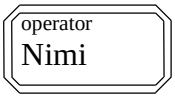
- Prosessityyppi viittaus *Process Type Reference*
  - Sisältää viittauksen toisaalla määriteltyyn prosessityyppiin
  - Jos viittaus systeemitasolla myös kaiki lohkot tässä systeemissä voivat käyttää kyseitä proseduuria, ei ilmentymiä systeemitasolla

Systeemitason symbolit

- Palvelutyyppi viittaus *Service Type Reference*
  - Sisältää viittauksen toisaalla määriteltyyn palvelutyyppiin
  - Jos viittaus systeemitasolla myös kaiki lohkot ja niiden prosessit tässä systeemissä voivat käyttää kyseitä proseduuria, ei ilmentymiä systeemitasolla

Systeemitason symbolit

- Operaattorityyppi viittaus
  Operator Type Reference
  - Sisältää viittauksen toisaalla määriteltyyn operaattorityyppiin
  - Mahdollistaa operaattorin graafisen määrityksen



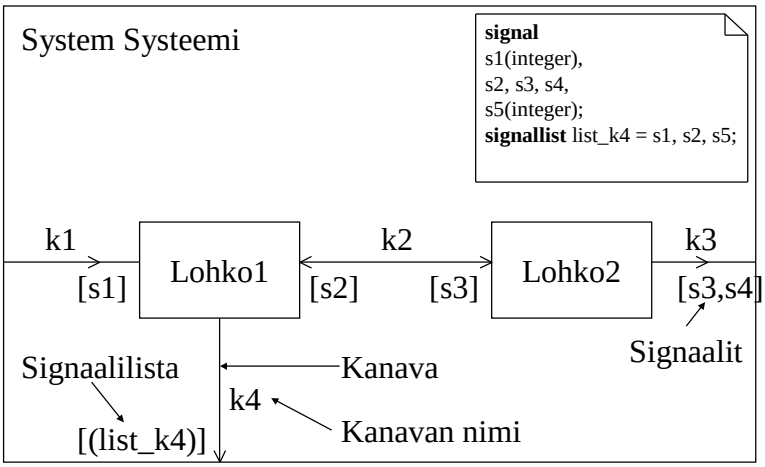
Systeemitason symbolit

- Kanavat *Channel*
  - Määrittävät signaalien kulkukanavat systeemitasolla lohkojen välillä ja ympäristöön
- Vain nimetyt signaalit voivat kulkea kanavassa
- Signaalien määrittely joko listoilla (*signallist)* tai yksittäin
- Kanavat nimettävä

Systeemitason symbolit

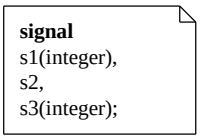
- Kanavat voivat olla, joko viiveellisiä tai viiveettömiä
- Viiveetön kanava määritellään vetämällä kanavamäärityksessä nuolen päät kanavan loppuihin (Seuraavan kuvan kanava k2)
- Prosessit tallentavat kanavista reittien kautta tulevat signaalit vastaan otto puskuriinsa (FIFO)
- Signaalien järjestys ei vaihdu kanavissa

Systeemitason symbolit



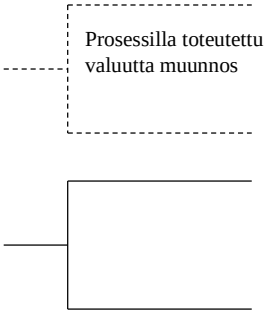
Lohkotason symbolit

- Teksti *Text*
  - Sisältää määrityksiä esim. signaaleista ja abstrakteista muuttuja tyypeistä
  - Lohkossa voi olla useita teksti laatikoita, jotka ovat tasa-arvoisia
  - Määritykset voimassa kaikkialla tämän lohkon alla



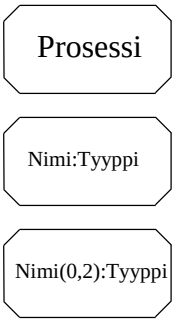
Lohkotason symbolit

- Kommenetti *Comment*
  - Ks. Systeemi
- Tekstin laajennus *Text extension*
  - Ks. Systeemi
- Käytetään vastaavasti kuin systeemitasolla



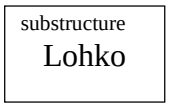
Lohkotason symbolit

- Prosessiviittaus *Process Reference*
  - Sisältää viittauksen toisaalla määriteltyyn prosessiin tai prosessityyppiin
  - Jos viitataan prosessityyppiin käytettävä myös prosessityyppi viittausta (jollei käytetä jo systeemitasolla)



Lohkotason symbolit

- ’Lohkon alirakenneviittaus’ *Block Substructure Reference*
  - Sisältää viittauksen toisaalla määriteltyyn lohkon alirakenteeseen
  - Symbolilla voidaan määrittää lohkolle lisää ominaisuuksia
  - Nimi voi olla sama, kuin varsinaisella lohkolla



Lohkotason symbolit

- Proseduuri viittaus *Procedure Reference*
- Lohkotyyppi viittaus *Block Type Reference*
- Prosessityyppi viittaus *Process Type Reference*

Toiminnot kuten systeemitasolla, määrittely pätee nyt kuitenkin vain lohkotasolla

Lohkotason symbolit

- Palvelutyyppi viittaus *Service Type Reference*
- Operaattorityyppi viittaus *Operator Type Reference*

Toiminnot kuten systeemitasolla, määrittely pätee nyt kuitenkin vain lohkotasolla

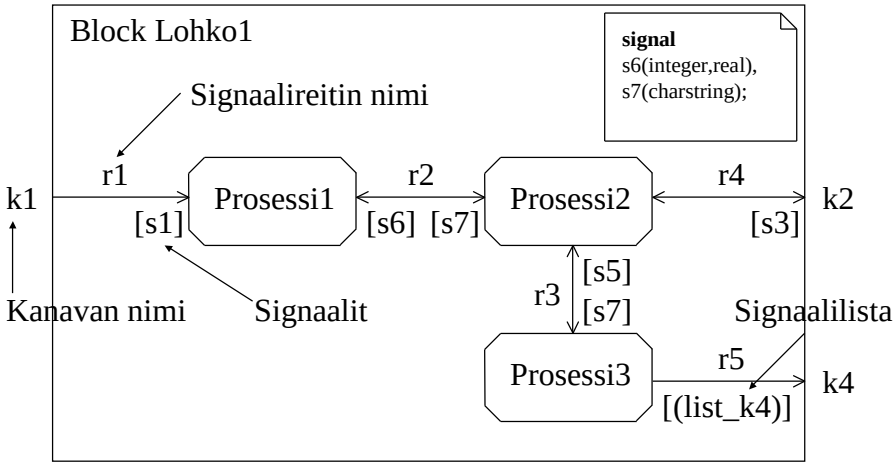
Lohkotason symbolit

- Signaalireitit *Signal Route*
  - Määrittävät signaalien kulkureitit lohko-tasolla prosessien välillä ja ympäristöön
- Vain nimetyt signaalit voivat kulkea reitillä
- Signaalien määrittely joko listoilla (*signallist)* tai yksittäin
- Nimettävä reitit ja kanavat joihin reitit yhdistyvät
- Lohkon sisällä (reitissä) kulkevalle signaalille ei aiheudu viivettä, eikä järjestys vaihdu

Lohkotason symbolit

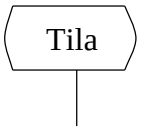
- Portti *Gate*
- Lohkotyypin määrittelyn yhteydessä on sille määriteltävä myös ns. kommunikointiportti(t)
- Porteilla määritellään tyypille tulevat (sen hyväksymät) ja siitä lähtevät signaalit
- Portti ei päästä läpi muita, kuin määriteltyjä signaaleja

Lohkotason symbolit



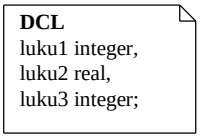
Prosessitason symbolit

- Tila *State*
- Tilassa prosessi odottaa haluttua syötettä
- Alustuksen jälkeen prosessi siirtyy alkutilaan
- Prosessin suoritus päättyy yleensä johonkin tilaan



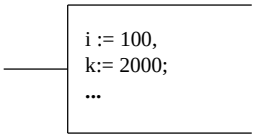
Prosessitason symbolit

- Teksti *Text*
  - Sisältää esim. muuttujien esitelyjä
  - prosessissa voi olla useita teksti laatikoita, jotka ovat tasa-arvoisia
  - Määritykset voimassa vain prosessin alla



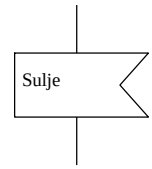
Prosessitason symbolit

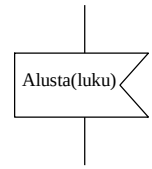
- Kommentti *Comment*
  - ’Laatikkoon’ voidaan sijoittaa ohjelman suoritukseen vaikuttamaton kommentti
- Tekstin laajennus *Text Extension*
  - ’Laatikolla’ voidaan laajentaa symbolin kirjoitustilaa
  - Voi sisältää esim. sijoituksia



Prosessitason symbolit

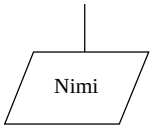
- ’Input’ *Input*
  - Prosessi käyttää symbolia lukeakseen signaalin syötejonostaan
- Jos vastaan otettavalla signaalilla on parametrejä on ne ’kopioitava’ prosessissa esiteltyihin muuttujiin





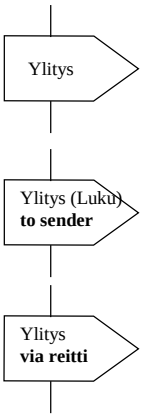
Prosessitason symbolit

- Talleta *Save*
  - Käytetään pitämään nimetty signaali syötejonossa tai siirtämään se talletusjonoon
- Talletettu signaali uudelleen käytössä tilan muuttuessa
- Luettua signaalia ei voida uudelleen tallettaa



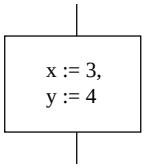
Prosessitason symbolit

- ’Output’ *Output*
  - Käytetään lähettämään signaali toiselle prosessille tai ympäristöön
- Signaalissa voi olla parametrejä
- Signaalin kohde (Pid) tai reitti voidaan määrittää (TO tai VIA)



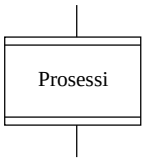
Prosessitason symbolit

- Tehtävä *Task*
  - Käytetään mm. muuttujien arvojen sijoittamisessa ja ulkopuolisten C-, PL/M- tai SDL-funktioiden kutsuissa



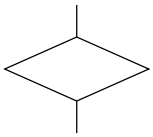
Prosessitason symbolit

- Luo *Create*
  - Käytetään luomaan uusi esiintymä parametrinä annetusta prosessista
- Luonnin onnistuessa sijoitetaan luovan prosessin OFFSPRING-muuttujaan luodun prosessin pid
- Ainoastaan Master-prosessi voi luoda uusia prosesseja



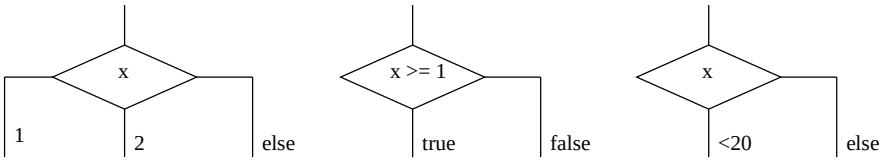
Prosessitason symbolit

- Ehto *Decision*
  - Käytetään, kun halutaan ohjata prosessin toimintaa jonkun arvon mukaisesti
- Vertailu tarkan arvon lisäksi myös vertailumerkein (< , >=, …) tai true, false tai else menetelmällä
- SDL:n IF-ELSE, WHILE ja FOR



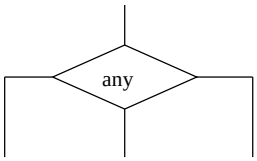
Prosessitason symbolit

- Ehto *Decision* voidaan toteuttaa mm. seuraavilla tavoilla



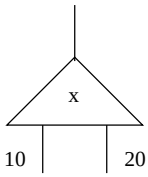
Prosessitason symbolit

- Ehtosymbolin yhteydessä voidaan käyttää avainsanaa *any*, jolloin saadaan toteutettua prosessiin ei deterministinen käyttäytyminen
- Suorituksen yhteydessä valitaan satunnainen ehtosymbolin haara



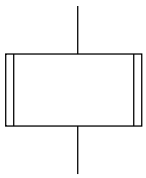
Prosessitason symbolit

- Vaihtiehto *Alternative*
  - Käytetään, kun halutaan ohjata prosessin toimintaa jonkun vakio-arvon mukaisesti
- Ehtoina siis vain vakioarvoja



Prosessitason symbolit

- Kutsu *Call*
  - Käytetään, kutsuttaessa SDL-proseduuria, joka ei paluta arvoa
- Arvon palauttavaa proseduuria kutsutaan, joko tehtävä- tai ehtosymbolista avainsanalla *call*

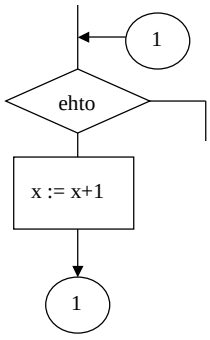


Prosessitason symbolit

- Stop *Stop*
  - Pysäyttää prosessin suorituksen
  - Tuhoaa myös kaikki syötejonossa olevat signaalit ja pysäyttää prosessin käynnistämät ajastimet

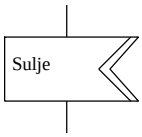
Prosessitason symbolit

- Nimeä ja yhdistä *Label and Join*
  - Käytetään goto:n tapaan
- Jos koodi ei mahdu yhdelle sivulle voidaan luoda linkki eri sivujen symbolien välille käyttämällä tätä symboliparia



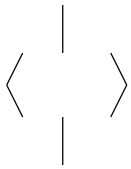
Prosessitason symbolit

- Priorisoitu Input *Priority input*
  - Jos syötejonossa on useita signaaleja käsitellään ensin priorisoidut signaalit saapumisjärjestyksessä
- Priorisointi koskee vain sen hetkisen tilan hyväksymiä signaaleja



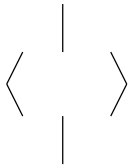
Prosessitason symbolit

- Jatkuva signaali *Continuous Signal*
  - Käytetään, kun halutaan tarkkailla jonkin tietyn signaalin muutosta jatkuva-aikaisesti



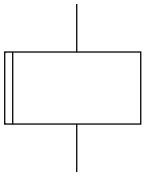
Prosessitason symbolit

- Mahdollistava ehto *Enabling Condition*
  - Mahdollistaa käyttäjälle lisäehtojen asettamisen tilasiirtymälle
  - Esim. muuttujan arvon tarkkailu



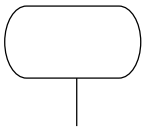
Prosessitason symbolit

- Makrokutsu *Macro Call*
  - Käytetään kutsuttaessa määritettyä makroa
- Makro voidaan määrittää hoitamaan esim. joku useissa prosesseissa toistuva toimenpide
- Käytetään koodin siistimiseen



Prosessitason symbolit

- Alku *Start*
  - Ilmoittaa prosessin alkupisteen
- Prosessissa on vain yksi alkusymboli, josta siirrytään prosessin luonnin yhteydessä ensimmäiseen varsinaiseen tilaan
- Alkusymboli ja ensimmäisen tilaan välissä voidaan suorittaa esim. muuttujien alustaminen



Prosessitason symbolit

- Proceduuri viittaus *Procedure Reference*
- Operaattori viittaus *Operator Reference*
- Portti *Gate*, jolla määritellään prosessityypin kommunikointiväylät

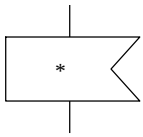
Toiminnot ja symbolit kuten systeemitasolla, määrittely pätee nyt kuitenkin vain kutsuvassa prosessissa

Prosessitason symbolit

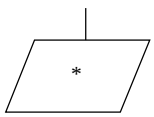
- Joidenkin symbolien yhteydessä on mahdollista käyttää esitystä yksinkertaistavia merkintöjä.
- Seuraavassa muutamia esimerkkejä….

Prosessitason symbolit

- ’Input’ *Input*
  - * tarkoittaa, että käsitellään kaikki signaalit, joita ei erikseen talleteta tai vastaan oteta
  - Voi sisältää myös useita signaaleja luettelona

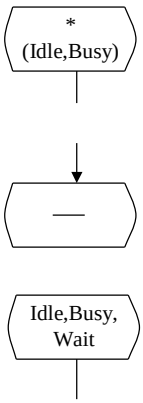


- Talletus *Save*
  - Talletetaan kaikki signaalit, joita ei erikseen talleteta tai vastaan oteta



Prosessitason symbolit

- Tila *State*
  - * tarkoittaa, että määrittely koskee kaikkia tiloja, joita ei mainita suluissa
  - - tarkoittaa, että lopputila on sama, kuin lähtötila
  - Voidaan myös erikseen nimetä käsiteltävät tilat luettelona



Proseduuritason symbolit

- Proseduuri tasolla mm. seuraavat erikoissymbolit
- Proseduurin alku *Procedure Start*
  - Proseduuri alkaa tällä symbolilla
- Proseduurin palutus *Procedure Return*
  - Lopettaa proseduurin toiminnan ja palauttaa määrätyt arvot kutsujalle