Lühiülevaade Jacksoni skeemide põhimõistetest

Jackson System Development Diagrams

Jacksoni süsteemiarendus (Jackson System Development, JSD) on süsteemi spetsifitseerimise ja disaini meetod. Selle lähenemise väljatöötamisel on kasutatud Jacksoni stuktuurprogrammeerimise (Jackson Structure programming, JSP) põhimõtteid.

JSDs on kasutusel kahte tüüpi diagrammid süsteemide modelleerimiseks - Entity Structure Diagrams (ESD olemidiagrammid) ja Network Diagrams.

JSD-l on kolm põhietappi:

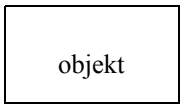
modelleerimisetapp (selle tulemusena tekib ESD)
network-etapp
juurutamisetapp

Entity Structure Diagram (ESD).

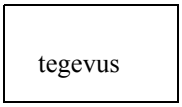
ESD on JSD väga oluline osa, mis tekib modelleerimisetapi tulemusena. ESD abil kirjeldatakse süsteemi olemi tegevused nende täitmisjärjekorras.

ESD elemendid:

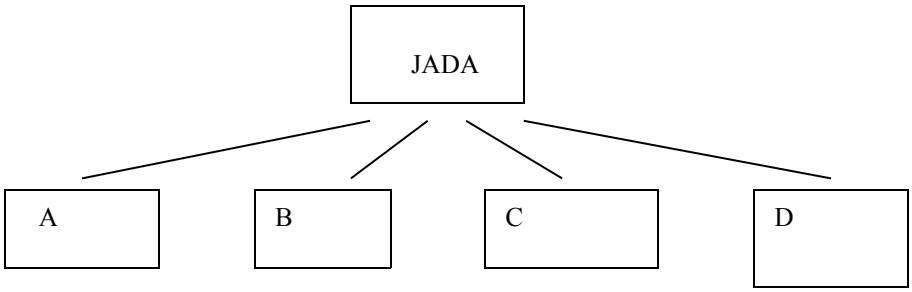
Olem (entity) - objekt, mis teeb ise midagi või millega teostab tegevusi süsteem. Olem võib olla kas füüsiline või kontseptuaalne, s.o. olemiks võib olla näiteks inimene või mingi projekt (või ka programm). ESD puhul on olemas ainult üks olem ja see olem on ESD puu juureks. Olemit kujutatakse ristkülikuna:



Tegevus (action) - lihtsalt öeldes on see midagi, mida olem teeb (või mida olemiga tehakse). ESDdes on tegevused omavahel ja olemiga seotud "vanema-lapse" suhte kaudu. Nagu ka olemit, kujutatakse tegevust ristkülikuna:



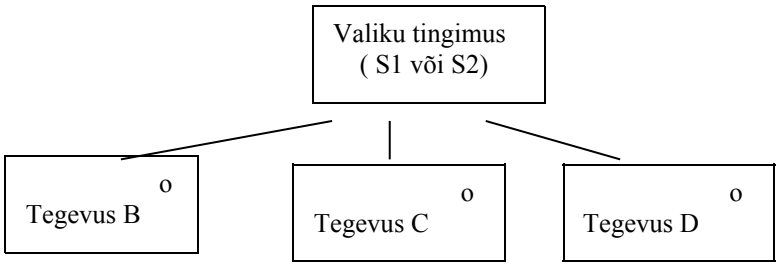
Jada, järjestus (sequence) - koosneb mitmest (kahest või enamst) tegevusest, mida täidetakse teineteise järel. Jada kujutatakse puusarnase konstruktsioonina, kus ülemisel tasandil asub "vanem" (olem või tegevus) ja alumisel tasandil asuvad "lapsed", mis on järjestatud vasakult paremale, vastavalt täitmisejärjekorrale.



Joonisel kujutatud situatsioonile vastaks pseudokood:

```
{
  Tegevus A;
  Tegevus B;
  Tegevus C;
  Tegevus D;
}
```

Valik (selection) - koosneb kahest või enamast üksteist välistavast osast, nendest täidetakse vaid üks. Valikut kujutatakse riskülikuna, mille ülemises paremas nurgas on "o"(tähistab ingliskeelset sõna "option"). Valikud on oma "vanemaga" (olemi või tegevusega) seotud joonte abil:

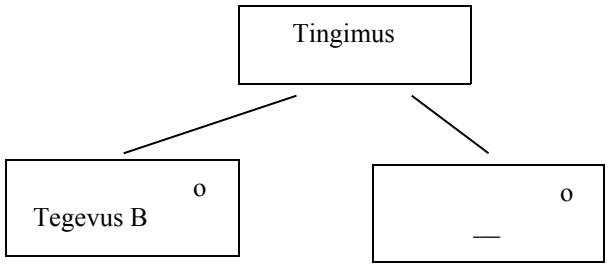


Joonisel kujutatud situatsioonile vastaks pseudokood (S1 ja S2 on tingimused vastavate tegevuste täitmiseks):

```
if <S1> {
  Tegevus B
else
 if <S2> {
  Tegevus C
else
  Tegevus D }}
```

Märkus: juhul kui tegemist on vaid kahe valikuga (if-then-else), siis võib põhimõtteliselt tegevuse A asemel olla tingimus; juhul, kui tingimus on tõene (TRUE), täidetakse vasakut valikut (then, +), kui aga tingimus on mittetõene (FALSE), täidetakse paremat valikut (else, -).

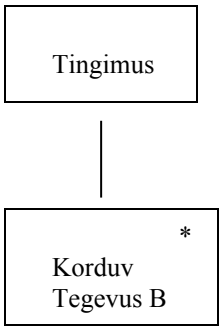
Null-komponent e. null-valik (null component, null selection) - tühi tegevus; kasutatakse näiteks if-then-else konstruktsioonis juhul, kui else puhul pole vaja mitte midagi teha (s.o. on olemas vaid if-then osa). Valiku puhul võib ESDs olla vaid üks null-komponent. Tähistatakse ristkülikuna, mille sees on horisontaalne kriips:



Joonisel kujutatud situatsioonile vastaks pseudokood (S1 on tingimus vastava tegevuse täitmiseks):

if <S1> {
Tegevus B}

Kordus (iteration) - tegevus, mida korratakse null või enam korda. Lihtsamalt öeldes - korduv tegevus (tsükkel). Ühel "vanemal" (olemil või tegevusel) võib olla täpselt üks "laps"-kordus (s.o. korduv tegevus). Kordust tähistatakse ristkülikuga, mille paremas ülemises nurgas on tärn "*":



Joonisel kujutatud situatsioonile vastaks pseudokood (I1 on kordamistingimus):

while <I1> {
Tegevus B }

<u>Diagrammide koostamise reeglid:</u>

Jacksoni olemidiagrammid on hierarhilise struktuuriga. See tähendab seda, et ei ole ühendusi tegevuste vahel, mis asuvad samal tasemel, ega ka eri harudes olevate tegevuste vahel. Lihtsamalt öeldes on igal tegevusel ainult üks vanem.

Diagrammis võib olla ainult üks olem ja see olem on diagrammipuu juureks.
Kõik ühe "vanema" "lapsed" peavad olema sama tüüpi s.o. ei tohi olla läbisegi jadategevusi, kordusi ja valikuid.

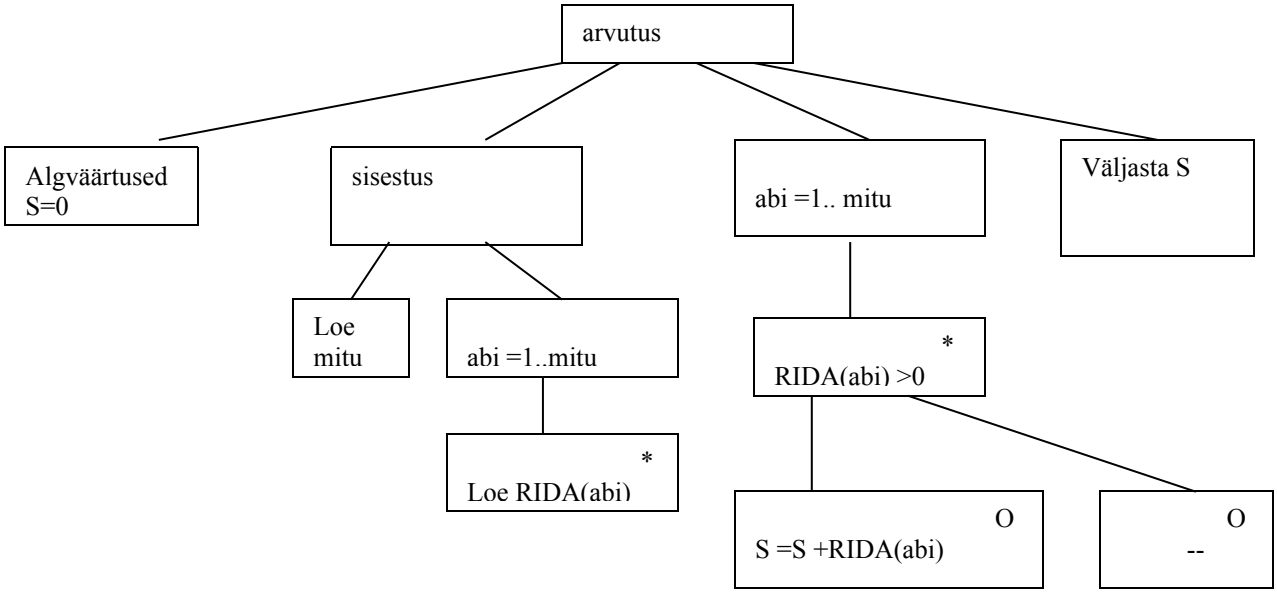
Korduvtegevusel ei ole "õdesid-vendi", ta on oma "vanema" ainuke "laps".

Tegevuste ja atribuutide tabelis peavad olema ära toodud kõik vastava olemi tegevused, nende tegevuste kirjeldused ja nende tegevustega seotud atribuudid.

tegevus tegevuse kirjeldus tegevuse atribuudid
tegevus1 ... ...
tegevus2 ... ...
... ...

Näide

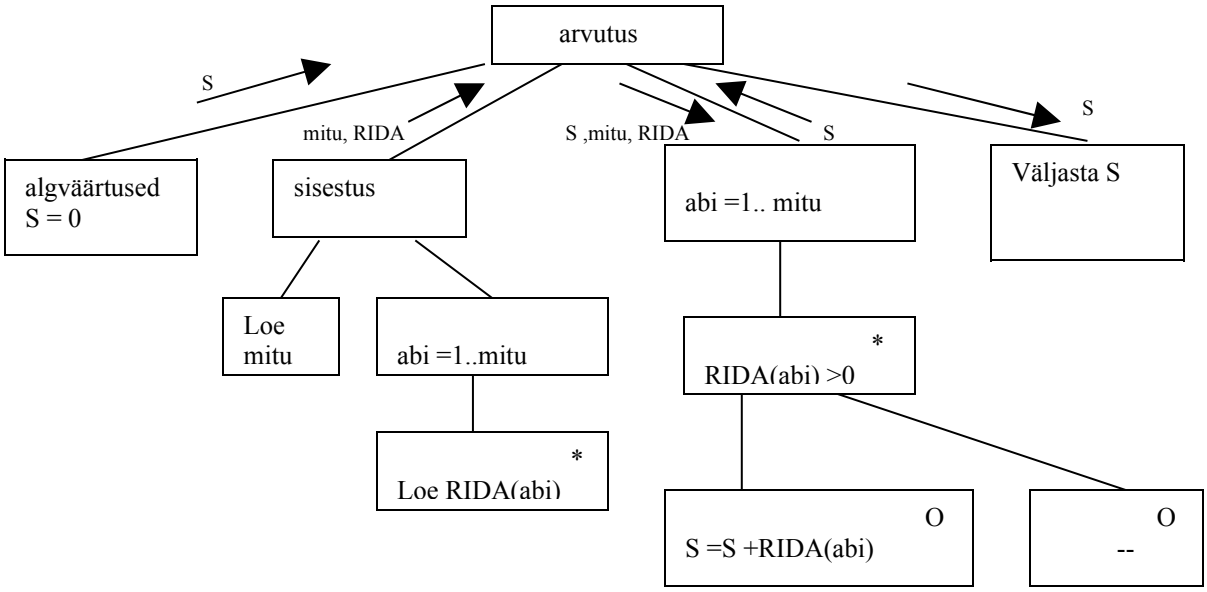
1.koosta algoritm $S=\Sigma A_I$, kus I= 1…N ja A_I >0 (st tuleb leida rea positiivsete elementide summa).



2.*Tegevuste ja atribuutide tabel*

algväärtuste omistamine	S=0	S – reaalarvude summa
sisestus		
reapikkuse sisestus	loeme klaviatuurilt “mitu”	mitu- liikmete arv reas, täisarv
kordus 1		
korduse tingimusekontroll 1	anname muutujale “abi “ väärtuseid 1..mitu	abi, mitu – täisarvud
korduse teostus 1	loeme rea elemente	RIDA - ühemõõtmeline massiiv abi- indeks
kordus2		
korduse tingimusekontroll 2	anname muutujale “abi “ väärtuseid 1..mitu	abi, mitu – täisarvud
korduse teostus,valik2	kontrollime, kas rea element on positiivne	RIDA - ühemõõtmeline massiiv abi- indeks
altern1	elemendi liitmine rea summale	S – reaalarvude summa RIDA - ühemõõtmeline massiiv abi- indeks
altern2	tühi tegevus	puuduvad

3.*Mõningatel juhtudel on piltlikkuse tõstmiseks otstarbekas kirjutada atribuudid , koos liikumise suunda näitavate nooltega skeemile.*



Refereeritud lehekülgedelt:

Jackson System Development Diagrams (05.05.2003)
http://panoramix.univ-paris1.fr/CRINFO/dmrg/MEE/misop002

Entity Life Histories (ELH) (01.09.2003)
http://www.doc.mmu.ac.uk/online/SAD/T08/elh.htm

How to Draw Jackson System Development (JSD) Diagrams (01.09.2003)
http://www.smartdraw.com/resources/centers/software/jsd.htm

Tutorial on JSP & JSD (01.09.2003)