KODUSE ÜLESANDE ABIMATERJAL. UML lühiülevaade

UML (Unified Modeling Language, ühtne mudelikeel) on harustandardiks kujunenud keel tarkvarasüsteemides kasutatavate artefaktide spetsifitseerimiseks, visualiseerimiseks, väljatöötamiseks ja dokumenteerimiseks. UML pakub valmistusele kindla tugistruktuuri, hõlbustades tarkvaraarenduse keerukat protsessi. UML-i töötasid välja firma Rational Software juhtivad süsteemispetsialistid Grady Booch, Ivar Jacobson ja Jim Rumbaugh.

Keel tarkvarakeskse süsteemi artefaktide visualiseerimiseks, spetsifitseerimiseks, väljatöötamiseks ja dokumenteerimiseks.

Allikas: UML keele sõnastik ***http://www.cc.ioc.ee/uml/***

Lühiülevaade UML skeemidest:

Kasutuslooskeem (Use case diagram): Kasutuslooskeeme kasutatakse süsteemi moodustavate tähtsaimate elementide ja protsesside määramiseks. Primaarelemente nimetatakse "aktoriteks" ning protsesse "kasutuslugudeks". Kasutuslooskeem näitab, millised aktorid suhtlevad iga kasutususlooga.

Klassiskeem (Class diagram): Klassiskeemi kasutatakse nö. kasutuslooskeemi viimistlemiseks ning üksikasjaliku süsteemidisaini määramiseks. Klassiskeem liigitab kasutuslooskeemil määratud aktorid omavahel seotud klasside kogumiks. Klassidevaheline suhe või assotsiatsioon võib olla kas "on" või "omab" tüüpi. Iga klassiskeemil toodud klass on võimeline pakkuma teatud funktsionaalsust. Neid nimetatakse klassi meetoditeks. Lisaks sellele on igal klassil olemas rida atribuute mis määravad klassi üheselt.

Objektiskeem (Object diagram): objektskeem on teatud tüüpi klassiskeem. Objekt esitab klassi olekut teatud ajahetkel süsteemi töö käigus. Objektiskeem esitab süsteemi erinevate klasside olekuid ning nendevahelisi relatsioone või assotsiatsioone teatud ajahetkel.

Olekuskeem (State Diagram): nagu ka nimi ütleb näitab olekuskeem erinevaid olekuid, mida läbivad süsteemis olevad objektid oma elutsükli jooksul. Süsteemis olevad objektid muudavad oma olekut vastavalt süsteemis toimuvatele sündmustele. Lisaks sellele näitab olekuskeem ka objekti oleku üleminekut algolekust lõppolekusse vastavalt süsteemi mõjutavatele sündmustele.

Tegevusskeem (Activity diagram): protsesside kulgemist süsteemis kirjeldatakse tegevusskeemi abil. Sarnaselt olekuskeemile koosneb ka tegevusskeem toimingutest, tegevustest, üleminekutest, alg- ja lõppolekust ning tõkisetingimustest

Jadaskeem (Sequence diagram): jadaskeem esitab süsteemi objektide omavahelist suhtlemist. Jadaskeemi oluliseks omaduseks on selle ajaline järjestus. S.o. esitatakse samm-sammult täpne objektide vaheline interaktsioon. Erinevad objektid jadaskeemil suhtlevad omavahel "sõnumite" edastamise kaudu.

Koostööskeem (Collaboration diagram): koostööskeem grupeerib erinevate objektide vahelise interaktsiooni. Interaktsioonid esitatakse nummerdatuna, mis lubab jälgida nende toimumise järjekorda. Koostööskeem lubab kindlaks teha kõikvõimalikud interaktsioonid mis igat objekti teistega seovad.

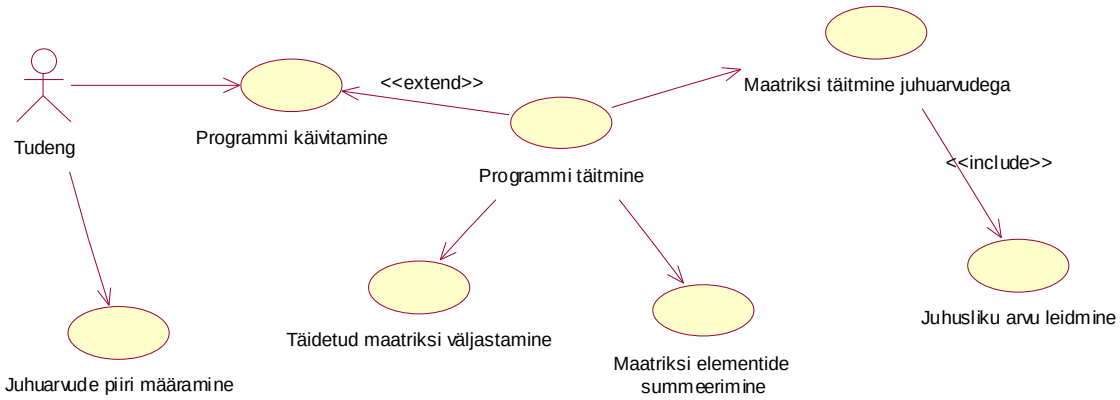
Komponentskeem (Component diagram): komponentskeemi abil kujutatakse kõrgtaseme osi, millest süsteem koosneb. See skeem esitab millised komponendid süsteemi moodustavad ning kuidas nad omavahel seotud on.

Levitusskeem (Deployment diagram): levitusskeemi abil kujutatakse rakenduse käitusaegeseid elemente.

NÄIDE

Ülesanne: M x N maatriks täidetakse juhuarvudega. Saadud maatriks väljastatakse ekraanile ning kasutajal palutakse sisestada summeeritavate väärtuste alam- ja ülempiir. Etteantud vahemikku jäävad arvud summeeritakse, kuvatakse ekraanile ning kuvatakse ka leitud summa.

Kasutuslooskeem - UML Use Case Diagram



Kasutuslooskeem on oma olemuselt üsna lihtne ning selles on kasutusel kahte tüüpi elemendid: üks neist esitab rolle ja teine protsesse.

Aktor (Actor): aktor on keegi või miski mis täidab teatud rolle antud süsteemis. Kasutuslooskeemil suhtleb aktor mingi kasutuslooga. Näiteks pangasüsteemi modelleerimisel on aktoriks klient. Samas on teenust pakkuv isik samuti aktor. Süsteemi modelleerija otsustada on, millised aktorid mõjutavad süsteemi käitumist ja funktsionaalsust vajalikul määral. Kui mingi olem ei mõjuta funktsionaalsust mida parajasti modelleeritakse, siis ei ole mõtet vastavat olemit ka aktorina kujutada.

Kasutuslugu (Use case): kasutuslugu on mingisuguse süsteemi funktsionaalsuse visuaalne esitus kasutuslooskeemil. Vajalike kasutuslugude kindlakstegemisel tuleb määrata konkreetse süsteemi (ülesande) funktsioonid. Iga selline leitud funktsionaalsus on potentsiaalne kasutuslugu. Kui loodava süsteemi funktsionaalsus muutub selgemaks, siis muutuvad silmnähtavamaks ka kasutuslood .

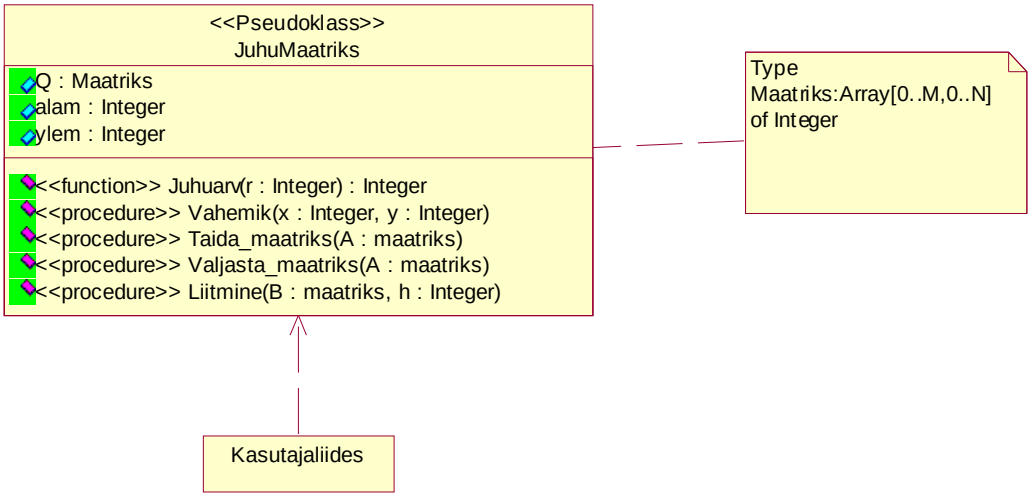
Kasutuslugude puhul on kasutusel erinevad relatsioonitüübid, mis näitavad ära nendevahelise sõltuvuse. Kasutusloo puhul võime rääkida järgmistest relatsioonidest:

- **Include:** kui üks kasutuslugu kasutab mingi teise funktsionaalust siis on nendevaheline relatsioon include.

- **Extend:** juhul kui kahe kasutusloo vahel on selline relatsioon, siis "laps" lisab "vanema" olemasolevale karakteristikutele midagi juurde.

Üldistus (generalization): seos üldisema ja spetsiifilisema elemendi vahel. Spetsiifiline element on täiesti kooskõlas üldisema elemendiga, aga sisaldab täiendavat informatsiooni.

Klassiskeem - UML Class Diagram

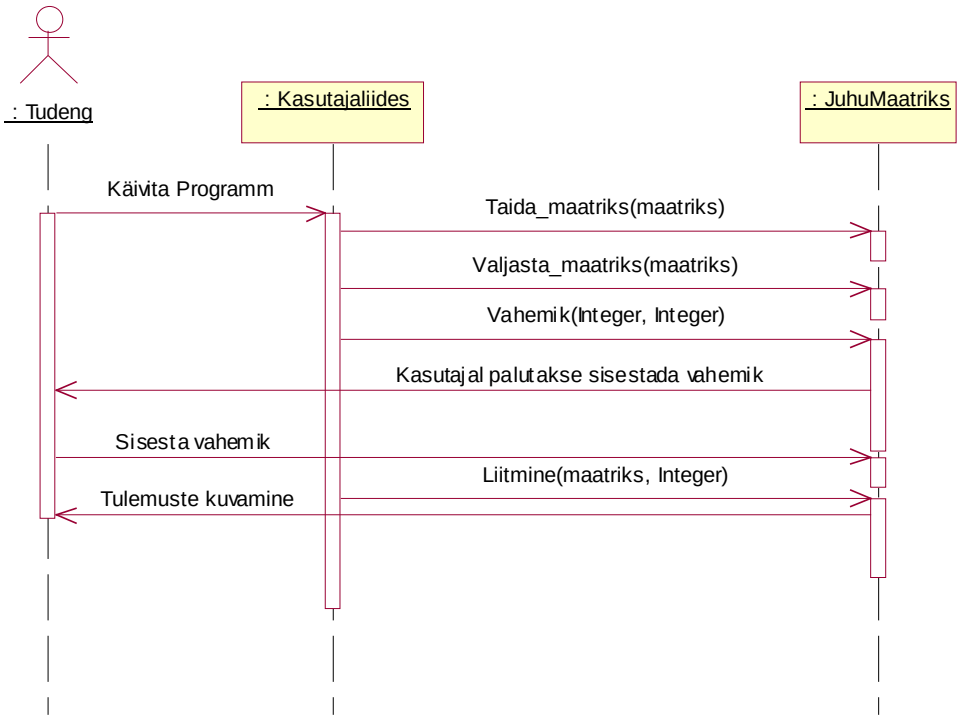


Klassiskeem on süsteemi disaini detailne piltlik kujutis. Klassiskeem on süsteemi staatiline vaade. Süsteemi struktuuri esitatakse klassiskeemide abil.

Klass: klass esitab antud süsteemi olemit, kajastades seejuures ka selle olemi funktsionaalsust. Klassi funktsionaalsus mida teised klassid kasutada saavad, nimetatakse meetodiks. Lisaks sellele on igal klassil olemas ka atribuudid.

Klassidevahelisi relatsioone on väga mitmeid. Selleks et näidata, et kaks klassi on omavahel seotud, piisab, kui nende vahele joon tõmmata.

Jadaskeem - UML Sequence Diagram



Jadaskeem - skeem, mis esitab objekti interaktsioonid ajalises järjestuses. Konkreetselt on skeemil näha interaktsioonis osalevad objektid ja vahetatavate sõnumite jada.

Jadaskeem võib olla üldkujul (kirjeldab kõiki võimalikke stsenaariume) või eksemplarkujul (kirjeldab ühte tegelikku stsenaariumi).

(Erinevalt koostööskeemist kujutab jadaskeem ajalist järjestust, kuid ei näita objektide seoseid. Jada- ning koostööskeemid väljendavad sama informatsiooni eri kujul.)

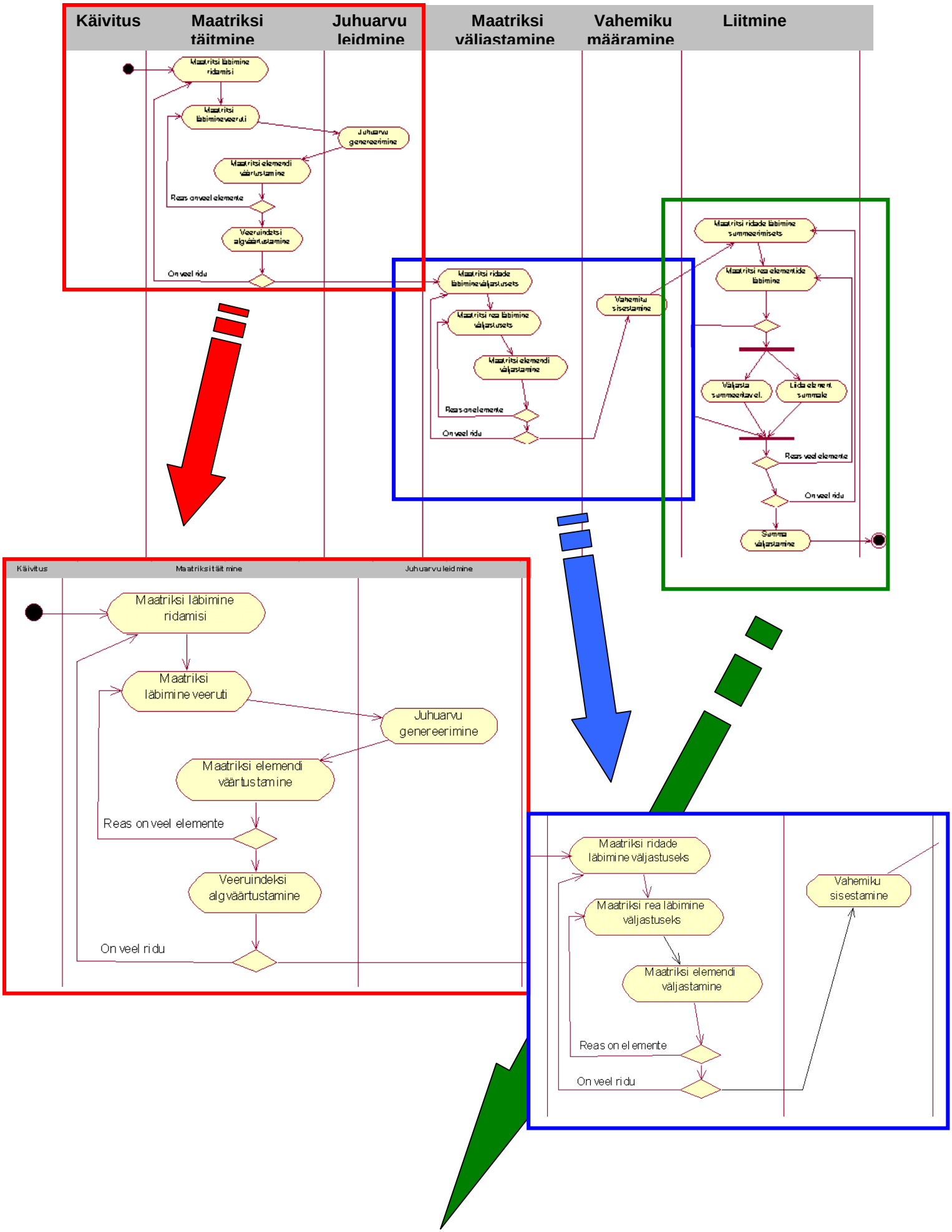
Tegevusskeem - UML Activity Diagram

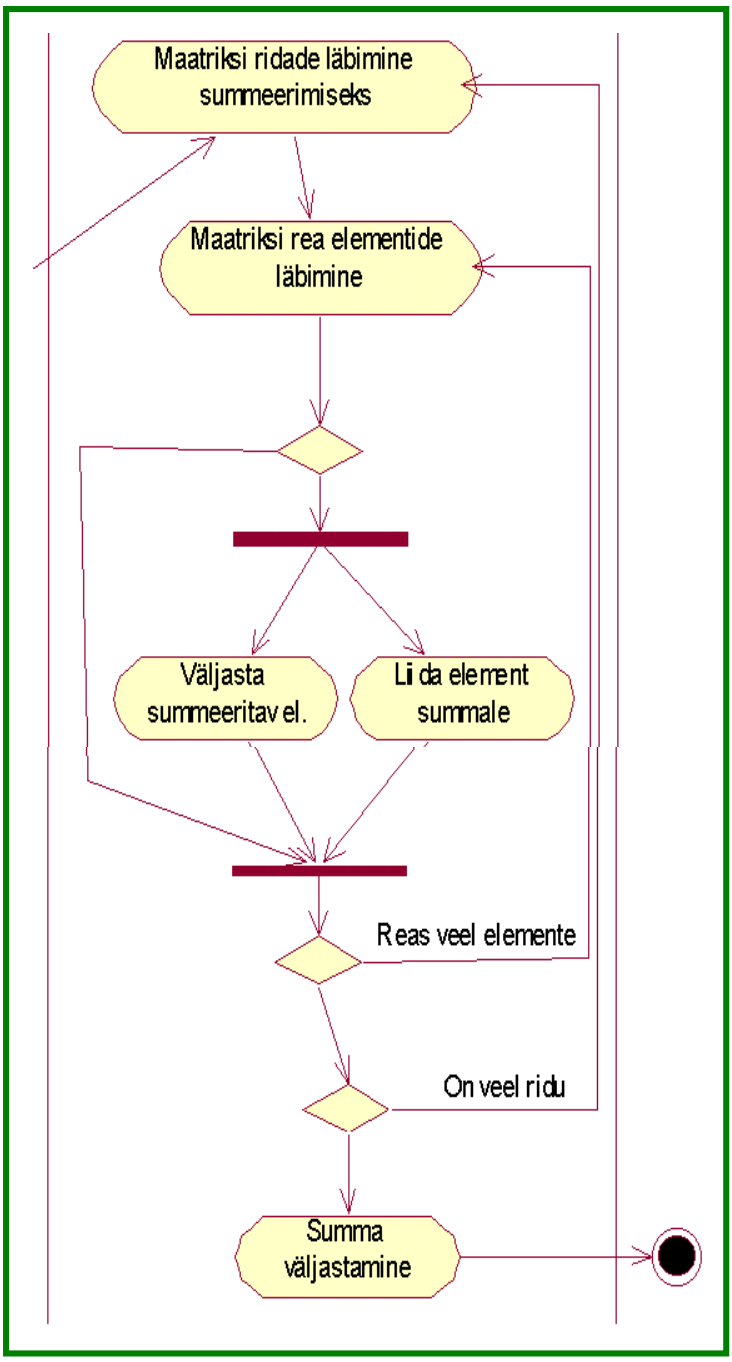
Tegevusskeemi kasutatakse süsteemis toimuvate sündmuste kujutamiseks. Tegevusskeem on dünaamiline skeem, mis näitab tegevust ning ka sündmust mis põhjustab objekti teatud olekus olemist.

Tegevusskeemi elemendid:

- algtevevus (kujutatakse musta ringina)
- tegevus (kujutatakse ümmarguste nurkadega nelinurgana)
- otsustused (kujutatakse rombikujulistena)
- signaal
- paralleelsed tegevused
- lõpptegevused

Swimlane'd võimaldavad tegevuste gruppeerimist tegijate kaupa.





Matrjali koostamisel kasutatud allikad:

UML Overview (29.03.2010): http://www.developer.com/design/article.php/1553851
Creating Use Case Diagrams(29.03.2010): http://www.developer.com/design/article.php/2109801
The UML Class Diagram: Part (29.03.2010): http://www.developer.com/design/article.php/2206791
The UML Class Diagram: Part II(29.03.2010): http://www.developer.com/design/article.php/2210821
Activity Diagram in UML(29.03.2010): http://www.developer.com/design/article.php/2247041