TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Yana Sirotkina 179716IABB

ÄRIANALÜÜSI TOETAVA VALDKONNASPETSIIFILISE MODELLEERIMISKEELE PROTOTÜÜBI LOOMINE

Bakalaureusetöö

Juhendaja: Mart Roost
Magistrikraad

Tallinn 2020

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Yana Sirotkina

12.05.2020

Annotatsioon

Lõputöö «Ärianalüüsi toetava valdkonnaspetsiifilise modelleerimiskeele prototüübi loomine» eesmärk on luua Eclipse Sirius platvormi abil valdkonnaspetsiifilise modelleerimiskeele prototüüp ettevõtte transaktsioonide ning väärtusvahetuste modelleerimiseks.

Esmalt kirjeldatakse mudelipõhilist arhitektuuri, selle eeliseid ja puuduseid. Edasi tutvustatakse valdkonnaspetsiifilist keelt ja valdkonnaspetsifilist modelleerimise keelt ja mõisted, mis on nendega seotud. Praktilises osas on kirjeldatud, kuidas väärtusvahetuste ja äritransaktsioonide modelleerimise keeled on loodud ja nende kasutamise näide.

Tulemuseks on nimetatud modelleerimiskeele spetsifikatsioonid, nende põhjal genereeritud tarkvara prototüüp ning selle kasutamise näide.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 37 leheküljel, 6 peatükki, 23 joonist, 0 tabelit.

Abstract

Creating a prototype of a domain-specific modeling language for business analysis

The aim of the thesis "Creating a prototype of a domain-specific modeling language for business analysis" is to create a prototype of a domain-specific modeling language for modeling business transactions and value exchanges using the Eclipse Sirius platform.

At the beginning is described the model-based architecture, its advantages and disadvantages. Further are introduced the domain-specific language, the domain-specific modeling language and related concepts. The practical part describes how value exchange and business transaction modeling languages are created and an example of their usage.

The result is the specifications of this modeling language, the prototype of the software generated from them and an example of its use.

The thesis is in Estonian language and contains 37 pages of text, 6 chapters, 23 figures, 0 tables.

Lühendite ja mõistete sõnastik

UML	Unified Modeling Language
MDA	Model Driven Architecture
BPMN	Business process modeling notation
PIM	Platform Independent Model
PSM	Platform Specific Model
DLS	Domain Specific Language
EMF	Eclipse Modeling Framework
Ecore	EMF mudel

Sisukord

Jooniste loetelu

1 Sissejuhatus

Käesolev töö on kirjutatud teemal «Ärianalüüsi toetava valdkonnaspetsiifilise modelleerimiskeele prototüübi loomine». Antud töö eesmärk on luua valdkonnaspetsiifilise modelleerimiskeele prototüüp, mida selle töö tellija saab kasutada tööriistana oma magistriainetes.

1.1 Taust ja probleem

Vajadus käesoleva töö järele tekkis seoses sellega, et töö tellinud juhendaja soovis oma magistriainetes õpetatavat ettevõtte modelleerimise metoodikat toetada sobivate vabavaraliste tööriistadega. Praegu on ained kasutusel tasuline universaalne modelleerimistarkvara Sparx Enterprise Architect, mis katab ära ettevõtte paljuvaatelise modelleerimise miinimumvajadused (näiteks UML ja BPMN diagrammid), aga mitte kõik vajalikud vaatenurgad ja standardid (näiteks väärtusvahetuste ja äritransaktsioonide modelleerimine). Samuti ei ole universaaltarkvaraga võimalik neid modelleerimisstandardeid metoodika jaoks spetsiaalselt kohandada ning metoodika arengule vastavalt uute meetodite ja tehnikatega täiendada. Valdkonnaspetsiifiliste modelleerimiskeelte tehnoloogiat kasutades saab teha keeled (modelleerimistehnikad) konkreetse metoodika jaoks täpselt sobivaks ja edasi arendatavaks. Arendusplatvormiks on valitud valdkonnaspetsiifiliste modelleerimiskeelte väljatöötamise platvorm Eclipse Sirius, mida kasutades on varem kaitstud lõputöö raames juba loodud samas metoodikas äriprotsesside modelleerimist toetava alamkeele prototüüp. Ettevõtte paljuvaatelise modelleerimise toetamiseks on seda prototüüpi vaja täiendada veel mitmete teiste alamkeeltega, millest esimesed võiksid olla (ettevõtte) äritransaktsioonide ning väärtusvahetuste modelleerimiseks.

1.2 Eesmärk ja oodatav tulemus

Eesmärgiks on luua Eclipse Sirius platvormi abil valdkonnaspetsiifilise modelleerimiskeele prototüüp ettevõtte transaktsioonide ning väärtusvahetuste modelleerimiseks.

Oodatavaks tulemuseks on nimetatud modelleerimiskeele spetsifikatsioonid, nende põhjal genereeritud tarkvara prototüüp ning selle kasutamise näide. Ehk peamine tulemus

on mudeli redaktor, kus saab teha täpselt need mudeleid, mis on kirjeldatud ja teha neid nii, kuidas kasutaja ise soovib.

1.3 Metoodika

Selle töö tööriistaks on valitud Eclipse Sirius, mille eeliseks on see, et iga arendaja saab Eclipse'i oma moodulitega laiendada. Sirius on mudelitega juhitav (model-driven) valdkonnaspetsiifiliste modelleerimiskeelte väljatöötamise platvorm, mis laiendab Eclipse modelleerimise raamistikku (Eclipse Modeling Framework), millega saab kergelt luua oma graafilise modelleerimise tööriista.

Eesmärkide saavutamiseks õigepealt täiendatakse varasema lõputöö raames välja töötatud äriprotsessi valdkonnamudelit, mis abstraktselt kirjeldab äriprotsessi olemust, ka organisatsiooni struktuuri ning väärtusvahetusi ja väärtustegevusi ehk äritransaktsioone. Kasutades seda mudelit on võimalik luua organisatsiooni struktuuri, väärtusvahetuste ja äriprotsesside kirjeldamiseks töövahendi.

Töö käigus modelleeritakse väärtusvahetuste ja äritransaktsioonide valdkonnad (keele abstraktne süntaks) ja genereeritakse mudelist koodi, spetsifitseeritakse diagrammitüüpe (ehk keele konkreetse süntaksi), katsetakse prototüüpi, hinnatakse ja dokumenteeritakse.

Kuna käesolev töö on projekti osa, võimaluse korral lõpus integreeritakse või liidestatakse teise tudengite töötulemusega.

1.4 Töö struktuur

Käesolev töö koosneb kuuest peatükist.

Esimene peatükk sisaldab sissejuhatust, kus on kirjeldatud töö taust ja probleem, eesmärk, oodatav tulemus ja metoodika.

Teine peatükk kirjeldab tööga seotud teoreetilisi aluseid, mida on kasutatud prototüübi loomise, tööriista kasutamise ja järelduse kirjutamise jooksul. Tutvustatakse mudelipõhise arhitektuuri mõistet ja selle arhitektuuri kirjeldavaid mõisteid. Lisaks on kirjeldatud valdkonnaspetsiifilised keeled ja valdkonnaspetsiifilised modelleerimise keeled.

Kolmas peatükk tutvustab prototüübi tegemisel kasutatud tarkvara, milleks on Eclipse Sirius. Kirjeldatakse Eclipse project, Eclipse Foundation ja Eclipse Sirius, mis nendesse kuulub ja ka EMF'i, mis on Eclipse Sirius'es kasutusel.

Neljas peatükk kirjeldab praktilise töö teostust, mis võimaldab käesoleva töö eesmärke saavutada. Esiteks luuakse valdkonnamudel ehk ecore mudel, mis näitab valdkonnaspetsiifilise keele elemente ja nende seoseid. Koostatud mudeli abil edasi on kirjeldatud organisatsiooni struktuuri ning väärtusvahetuse kirjeldamiseks vajalikud diagrammid ja pärast nad on juba näidatud kasutamises. Lõpus on kirjeldatud integreerimise protsess teise tudengi varem kaitstud tööga ja integreerimise tulemust kasutades tehtud protsessidiagramm.

Viies peatükk sisaldab järeldusi tulemuste ja tööriista kohta. Peatükkis ka on kirjeldatud ka prototüübi edasise arenduse plaan.

Kuues peatükk võtab tehtud töö lühidalt kokku.

2 Mudelipõhine arhitektuur

Järgnevas peatükis on kirjeldatud tööga seotud teoreetilisi aluseid, mida on kasutatud prototüübi loomise, tööriista kasutamise ja järelduste kirjutamise jooksul. Peatükk kirjeldab mudelipõhise arhitektuuri mõistet ja seda arhitektuuri kirjeldavaid mõisteid. Lisaks on kirjeldatud valdkonnaspetsiifilised keeled ja valdkonnaspetsiifilised modelleerimise keeled, kuna nad omavad suurt rolli mudelipõhise arhitektuuri arenduses.

Kõigepealt aga defineerime töö probleemvaldkonnaks oleva väärtusvahetuste ja väärtustegevuste valdkonna põhimõisted.

2.1 Väärtusvahetus ehk Value Exchange

Väärtusvahetus tegelaste vahel on üks majanduse põhialustest. Põhitehingus alati osaleb 2 tegelast, kus üks on tellija ja teine tegelane on tarbija. Selle protsessi käigus vahetatakse andmeid, raha jms oma eesmarke saavutamiseks.

Väärtusvahetus toimub mitte ainult poes või pizzerias, vaid näiteks ülikoolis toimub väärtusevahetus tudengitega, kus ülikool annab teadmisi ning tudeng annab raha või aega tasuta õppimise korral.[1]

2.2 Äritransaktsioon ehk Business Transaction

Äritransaktsioon on kindel toiming poolte vahel. Esimene tegelane palub teha selle tegevust või teenust, aga teine pakub/suudab täita.

Äritransaktsioon on sündmus, mis hõlmab kauba, raha või teenuste vahetamist kahe või enama osapoole vahel. Tehing võib olla nii lühike kui sularaha ost või sama pikaajaline kui aastate pikkune teenusleping.[2]

2.3 Model Driven Architecture ehk Mudelipõhine arhitektuur

MDA (Model Driven Architecture) kontseptsioon on loodud pakkuma ühist alust enamike olemasolevate standardite kirjeldamiseks ja kasutamiseks, ilma et see piiraks

arendajaid konkreetsete tehnoloogiate valimisel. MDA kasutamine tarkvara arendamiseks ja integreerimiseks võimaldab säästa tehtud investeeringuid äriloogika arendamisse, isegi tehnoloogiaplatvormide vahetamisel.

Esiteks luuakse spetsiaalsete disainiriistade abil rakenduse üldine ja (tehnilisest) platvormist sõltumatu mudel ning seejärel rakendatakse programm mis tahes arenduskeskkonnas. Pealegi põhineb arendusprotsess täielikult mudelil, mis peaks sisaldama kogu programmeerimiseks vajalikku teavet. Selle lähenemisviisi eelised on ilmsed. [3]

Järgnevalt on toodud MDA mudelit kirjeldavad mõisted:

Süsteem – tarkvarasüsteem, mis on MDA kontekstis juba olemasolev või arenduses olev;

Mudel – platvormimudel, mis sisaldab tehnilisi kirjeldusi, liideseid, platvormi funktsioone. Sageli on platvormimudel esitatud tehniliste kirjelduste, juhendite, jooniste ja teksti kombinatsiooni kujul MDA eesmärkide täitmiseks tuleks platvormimudelite kirjeldus esitada ühes kokku lepitud modelleerimise keeles näiteks UML;

Mudelipõhine ideoloogia – süsteemi arendamise protsess, mis kasutab mõistmiseks, konstrueerimiseks, levitamiseks ja muude toimingute jaoks mudelit;

Arhitektuur – süsteemi kuuluvate osade ja ühenduste spetsifikatsioon ja nende koostoimet võimaldavad reeglid;

Vaatepunkt – abstraktsioonitehnika, mis võimaldab keskenduda ühele aspektile, ignoreerides kõiki ebaolulisi üksikasju. vaatepunkti saab esindada ühe või mitme mudeli kaudu;

MDA vaatepunkt – MDA määratleb süsteemis kolm vaatepunktide liiki: struktuurist sõltumatu, platvormist sõltumatu ja platvormist sõltuv.

- **Struktuurist sõltumatu** – mudel varjab realiseerimise detailid ja süsteemiprotsessid. Ta keskendub ainult tarkvaranõuetele ja kontekstile.
- **Platvormist sõltumatu** – vaatepunkt keskendub süsteemi operatsioonidele, varjates konkreetsete platvormidega seotud detaile. Seda vaatepunkti esindab Platvorm Independent Model ehk PIM.

- **Platvormist sõltuv** – vaatepunkt tuleneb platvormist sõltumatu vaatepunkti kohandamisest platvormide konkreetsete detailidega. Seda vaatepunkti esindab Platform Specific Model ehk PSM.

Platvorm – alamsüsteemide ja tehnoloogiate komplekt, mille sees on funktsionaalsuste kogum, mida kasutab iga rakendus ilma implementeerimise detaile täpsustamata;

Platvormist sõltumatus – mudeli kvaliteet, mis tähistab selle sõltumatust konkreetse platvormi omadusest;

Platvormi mudel – tehniliste kirjelduste, tehnoloogiate ja liidete kogum, mis kujundab platvormi;

Mudeli teisendus – ühe süsteemimudeli teisendamise protsess sama süsteemi teisele mudelile.

Implementatsioon – pakub kogu infot süsteemi ehitamiseks ja kasutusele võtmiseks. Peab pakkuma kõiki teabeid objekti loomiseks ja lubama objektile osaleda asjakohaste teenuste pakkumisel süsteemi terviku osana. [4]

2.4 Valdkonnaspetsiifilised keeled

Valdkonnaspetsiifilised keeled ise on piiratud oma tegevustes, kuid oma valdkonnas on nad teistest levinumatest programmeerimiskeeltest paremad. [5]

- Kuna nende maht on piiratud, saab neid lihtsamini ja kiiremini õppida;
- Nende viimine uuele platvormile on lihtne - samad HTML-dokumendid, mis on avatud 2000. aastal PDA-s, on nüüd avatavad ka iPad Pro-s.
- Vigu on lihtsam avastada ja parandada, sest nad on valdkonnaspetsiifilised. Siin ei ole kokkupuutumisi 'null' vigadega, vaid on vead, millest valdkonnaekspert saab aru.
- Nad on ohutumad, nende kasutamisel võib vähem asju valesti minna, mis on kasulik, kui teeme midagi, mis on seotud tervise või rahaga.
- Kuna valdkonnaspetsiifilised keeled on piiratud selles, mida nad saavad teha, on neid lihtsam analüüsida. [5]

Sellest järeldub, et peaaegu kõik valdkonnaspetsiifiliste keelte eelised tulenevad nende piiratud suurusest ja tegevustest.

2.5 Valdkonnaspetsiifilised modelleerimise keeled

Valdkonnaspetsiifilised modelleerimise keeled (DSML) vormistavad rakenduste struktuuri, käitumise ja nõuded konkreetsetes domeenides. DSML võimaldab mudeliarendajatel tajuda end otseselt domeenikontseptsioonidega töötajana. [6]

DSML-i mõiste hõlmab vähemalt kolme aspekti: valdkonnakontseptsioonid ja reeglid (abstraktne süntaks); nende mõistete tähistamiseks kasutatud märgid (sümbolid) – teksti kujul või graafilised (konkreetne süntaks); ja keele semantika:

- **Abstraktne DSML- i süntaksi** määratleb tavaliselt metamudel, mis ise on mudel ja kirjeldab keele mõisteid, nendevahelisi suhteid ning struktureerimise reegleid;
- **DSML-i konkreetne süntaks** pakub selle abstraktse süntaksi realiseerimist metamudeli mõistete ja nende tekstilise või graafilise kujutamise vastavuste kaardistusena. visuaalsete keelte jaoks on vaja luua seosed nende mõistete ja neid esindavate visuaalsete sümbolite vahel. Sarnaselt tekstikeeltega on vaja linke metamudeli elementide ja tekstilise DSML-i süntaktiliste struktuuride vahel;
- **DSML-i semantika** antakse tavaliselt loomuliku keelega. Kasutajad saavad tavaliselt mõista DSML-i enamike terminite tähendust, aga arvuti ei saa selliste eelduste järgi tegutseda. [6]

3 Kasutatud tehnoloogiad

Käesolev peatükk tutvustab tehnoloogiaid, mis on kasutatud prototüübi tegemisel. Eesmärgi saavutamiseks on valitud kõige sobivamad selle jaoks programm ja raamistik. Vastavalt on nendeks Eclipse Sirius ja Eclipse Modeling Framework.

3.1 Eclipse project ja Eclipse Foundation

Eclipse'i projekti lõi IBM (International Business Machines) 2001. aastal ja seda toetasid tarkvaratootjad. Eclipse'i projekti kasutavad miljonid arendajad tänapäevani. Paar aastat hiljem, 2004. aastal, asutati Eclipse'i sihtasutus iseseisva mittetulundusühinguna. Selle eesmärk on jälgida, et Eclipse'i keskkonnas saaks luua müüjate suhtes neutraalse, avatud ja läbipaistva kommuuni. Eclipse'i kommuun koosneb üksikutest arendajatest ja paljudes tööstusharudes tegutsevatest organisatsioonidest. Tarkvara müüjad saavad kasutada Eclipse tehnoloogiat oma kommertstarkvara toodete ja teenuste loomiseks tänu Eclipse'i avaliku litsentsi (EPL) , mille kaudu tarkvara saab kasutada, muuta või jagada. [7], [8]

3.2 Eclipse Sirius

Sirius on Eclipse'i projekt, mis võimaldab teil kergelt luua oma graafilise modelleerimise tööriista, kasutades Eclipse Modeling tehnoloogiaid. See pakub mitmekülgset mudelipõhilist arhitektuuri kujundamise tööriista, mida saab lihtsalt kohandada konkreetsetele vajadustele. Seetõttu seda tööd tehes kasutasime Eclipse Siriust. [9]

Eeliste hulgas, mille pärast kasutajad peaksid Siriust kasutama, on loetletud järgnevalt:

- Sirius hoiab oma sõnavara, nii et kasutajad ei pea õppima mõisteid, mis on nende ärivaldkonna jaoks võõrad ja arusaamatud. Nad peavad lihtsalt õppima tööriista pakutavaid vaateid ja nende vahel liikumist.
- Sirius võimaldab mitte maksta üleliigsete tööriistade eest, mis pakuvad ebavajalikke funktsioone. Selle põhjuseks on asjaolu, et Siriust saab enda jaoks häälestada ning Siriusega loodud modelleerimisriistad pakuvad vaateid ja funktsioone, mis vastavad erinevate kasutajate vajadustele. [10]

3.3 Eclipse Modeling Framework

Eclipse'i modelleerimise raamistik (EMF) on Eclipse'i plug-in'ite komplekt, mida saab kasutada andmemudeli modelleerimiseks ja selle režiimi põhjal genereeritud koodi või muu väljundi jaoks. EMF seletab erinevust metamudeli ja mudeli vahel nii, et metamudeli abil kirjeldatakse mudeli ülesehitust, aga mudel on selle metamudeli konkreetne näide. EMF võimaldab arendajal luua just metamudelit, kasutades erinevaid tööriistu näiteks XMI, Java annotatsioone, UML või XML-skeeme. [11]

EMF koosneb kolmest põhiosast:

- **EMF** - põhiline EMF-i raamistik koosneb Ecore'ist, mis võimaldab luua metamudeleid mudelite kirjeldamiseks.
- **EMF.Edit** - EMF.Edit raamistik sisaldab korduvkasutatavaid EMF-i klasse, et saaks mudelite jaoks redaktoreid ehitada.
- **EMF.Codegen** - selle eesmärk on genereerida kõike, mis on vajalik EMF-mudeli täieliku redaktori ehitamiseks. See sisaldab GUI-d, mille põhjal saab genereerimisvalikuid täpsustada ja generaatoritele tugineda. [12]

EMF toetab kolme taset koodi genereerimiseks:

- **Mudel** - pakub Java liideseid ja rakendusklasse kõigi mudeli klasside jaoks. Lisaks mudel pakub implementatsiooni klasse metaandmete jaoks.
- **Adapterid** - genereerivad implementatsiooni klasse (nn ItemProviders), mis on seotud mudeliklasside muutmise ja kuvamisega.
- **Redaktor** - loob õigesti struktureeritud redaktori, mis vastab Eclipse EMF-i soovitatud stiilile.

Kui kasutaja teeb muudatusi, toetavad kõik need tasemed koodi taastamist. Neid saab kasutada kas graafilise liidese kaudu või otse käsurealt. [12]

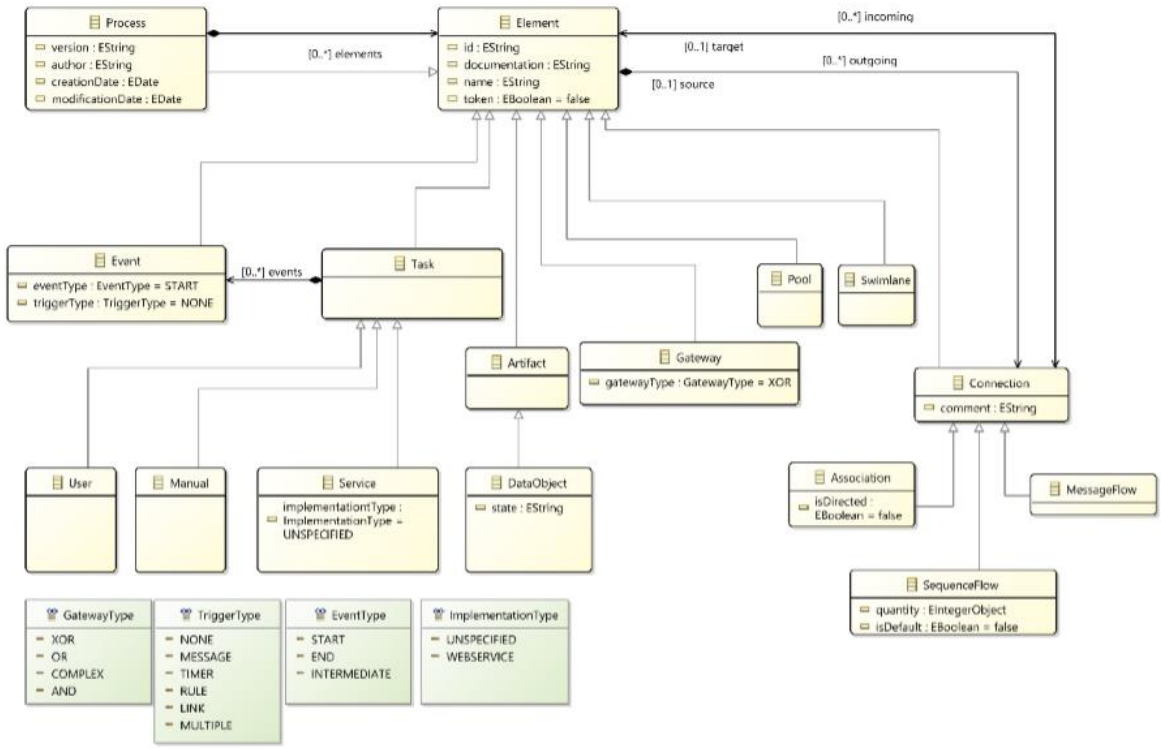
4 Valdkonnaspetsiifilise modelleerimiskeele prototüübi loomise realisatsioon

Käesolev peatükk kirjeldab, kuidas ärianalüüsi toetava valdkonnaspetsiifilise modelleerimiskeele prototüüp on tehtud samm-sammult. Relisatsiooni jaoks on kasutatud Eclipse Sirius tarkvara. Selle tarkvara abil esiteks luuakse valdkonnamudel ehk ecore mudel, mis näitab modelleerimiskeele elemente ja nende seoseid. Koostatud mudeli abil edasi on võimalik genereerida modelleerimiskeele elementide java klassid ja pärast kasutada neid juba konkreetse äriprotsessi diagrammi koostamises.

Praktilised lahenduted on tehtud allikate [15] - [19] põhjal. Ka oli kasutusel juhendaja poolt magistriainete jaoks tehtud esitlused.

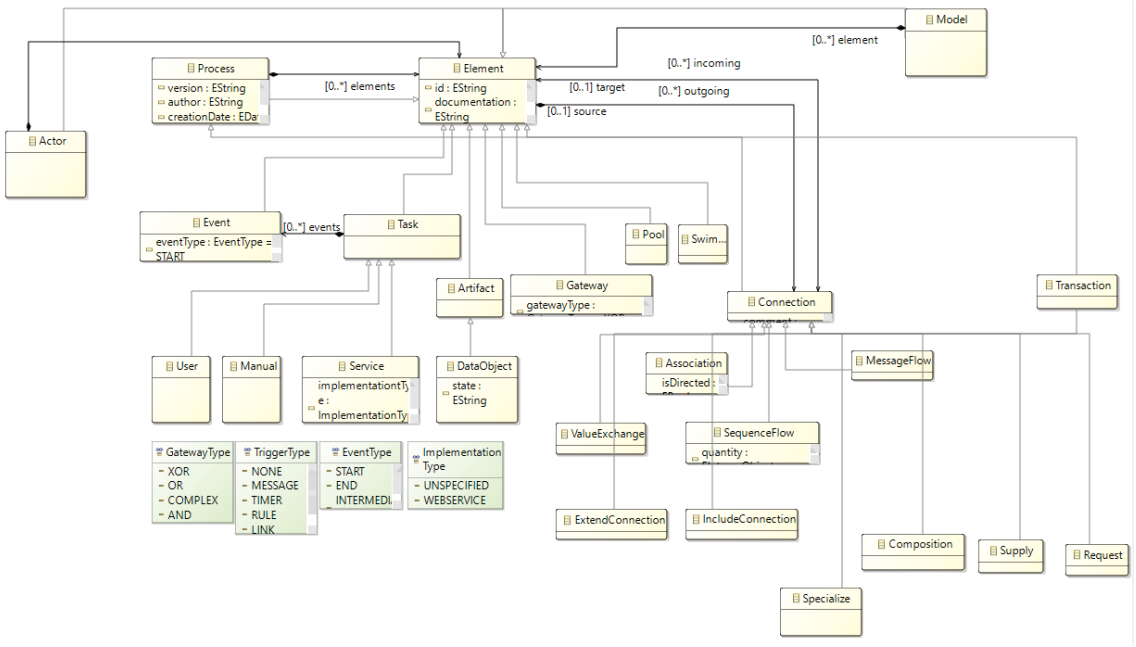
4.1 Valdkonnamudel

Kuna käesolev töö on «ÄRIPROTSESSIDE MUDELIPÕHINE ARENDAMINE ECLIPSE SIRIUS PLATVORMIL» (autor Ain-Martin Kink) töö jätk, siis see tähendab, et äriprotseside valdkonna metamudel on juba tehtud. Äriprotsesside modelleerimiskeele BPMN põhiobjektide kirjeldamiseks ja kasutamiseks oli vaja luua sobiv ecore mudel. Just eelneva töö alguses oli ecore mudel tehtud (joonis 1). See pärineb Richard C. Gronback Eclipse Modeling Project: A Domain Specific Language Kit raamatust. Valdkonnamudel on kirjeldatud Ain-Martini töös, kus ta kirjeldab objekte ja seletab, miks oli vaja allikas antud mudelit muuta.



Joonis 1. Esimene valdkonnamudel («Äriprotsesside mudelipõhine arendamine eclipse sirius platvormil» tööst)

Käesolev töö on pühendatud väärtusvahetuste ja äritransaktsioonide modelleerimisele, selle jaoks Ain-Martini valdkonnamudelis puuduvad olulised elemendid. Täeindamise jooksul oli loodud 9 uut elementi (joonis 2).



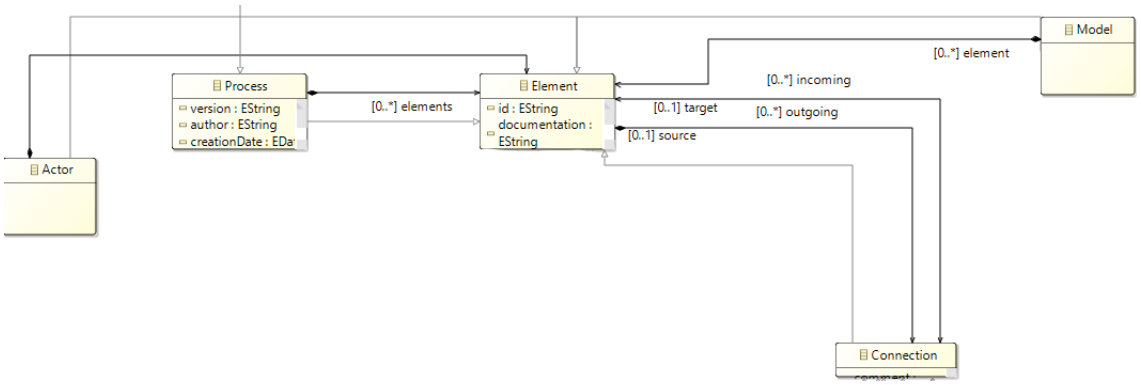
Joonis 2. Täiendatud valdkonnamudel

Kui äriprotsesside valdkonna mudelit esindavaks elemendiks on Protsess, siis väärtusvahetuste ja äritransaktsioonide mudelite omanikelemendiks ei pea tingimata olema protsessielement, vaid selleks võib olla näiteks tervet organisatsiooni või selle üksikut tegelast või tervet valdkonda ehk mudelit esindav element. Seepärast sai loodud elemendid *Model* ja *Actor* (joonis 3*). Process, Connection* ja *Element* elemendid oli juba tehtud eelmises töös. Need elemendid on tähtsad mitte ainult äriprotsesside, vaid ka äritransaktsioonide ja väärtusvahetuste kirjeldamisel.

Process on Element, mis esindab protsessi, näiteks äriprotsessi.

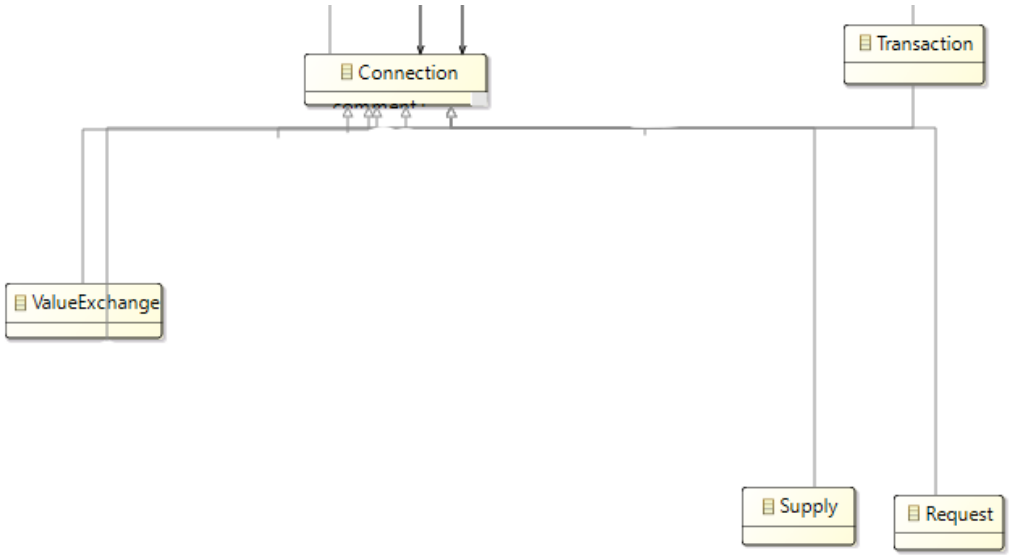
Element on abstraktsioon üle kõikide erinevat tüüpi elementide.

Model on Element, mis esindab konkreetset mudelit tervikuna.



Joonis 3. Actor, Process, Element, Model ja Connection elemendid ecore mudelis

Edasi on antud töö käigus loodud elemendid *ValueExchange, Transaction , Supply ja Request* (joonis 4). Kõik need elemendid väljendavad seost ehk ühendust kahe muu elemendi vahel, seepärast, sellepärast spetsialiseerivad nad Connection elementi.



Joonis 4. Transaction, ValueExchange, Supply ja Request elemendid ecore mudelis

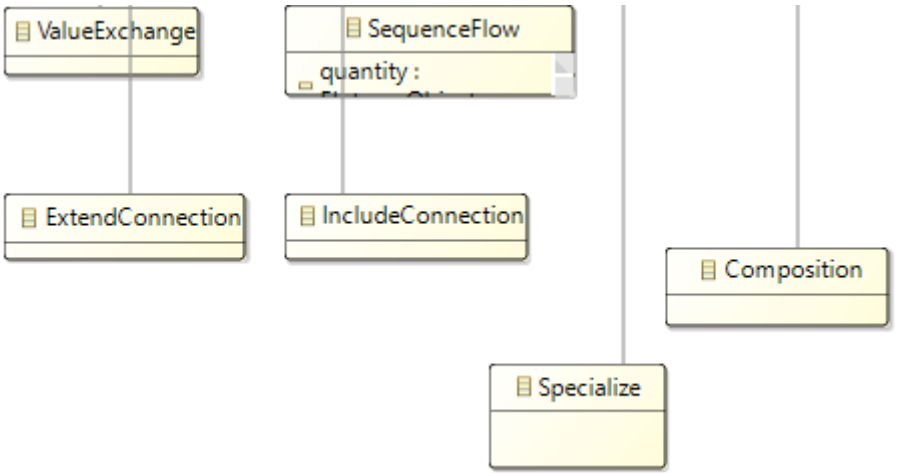
ValueExchange on väärtusvahetus kahe tegelase (Actor) vahel.

Transaction on väärtustegevus ehk äritransaktsioon, mis realiseerib ühte või enamat väärtusvahetust.

Supply on tegelase (Actor) ühendus (Connection) äritransaktsiooniga (Transaction), kus vastav tegelane on äritransaktsiooni täitja rollis.

Request on tegelase (Actor) ühendus (Connection) äritransaktsiooniga (Transaction), kus vastav tegelane on äritransaktsiooni kui tegevuse-teenuse tellija (soovija) rollis.

Viimasena loodi töö käigus elemendid *ExtendConnection, IncludeConnection, Specialize* ja *Composition* (joonis 5). Need elemendid on ka ühendused kahe muu elemendi vahel ning samuti spetsialiseerivad Connection olemit.



Joonis 5. ExtendConnection, IncludeConnection, Specialize ja Composition elemendid

ExtendConnection on laiendamise seos, mida antud töös kasutatakse kahe äritransaktsiooni (Transaction) vahel (üks äritransaktsioon laiendab teist).

IncludeConnection on sisaldamise seos, mida antud töös kasutatakse kahe äritransaktsiooni (Transaction) vahel (üks äritransaktsioon sisaldab teist).

Specialize on spetsialiseerimise seos, mida antud töös kasutatakse kahe tegelase (Actor) vahel (konkreetsem tegelane spetsialiseerib üldisemat tegelast).

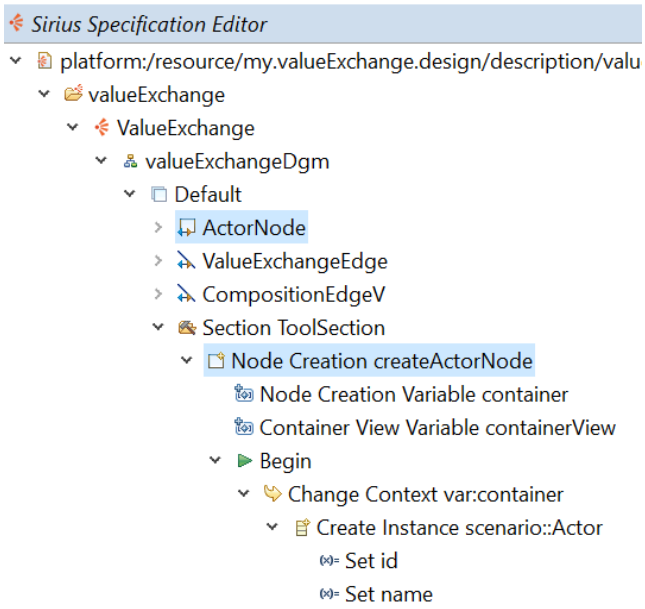
Composition on kompositsiooni seos, mida antud töös kasutatakse kahe tegelase (Actor) vahel (suurem tegelane sisaldab koostisosana väiksemat tegelast).

4.2 Väärtusvahetuste ja äritransaktsioonide mudelite elementide kirjeldamine

Erinevat tüüpi ärimõistete (sealhulgas äriprotsesside, -tegelaste, nende väärtusvahetuste ja toimingute/transaktsioonide kirjeldamine toimub diagrammide abil, mis koosnevad graafilistest elementidest. Graafiline kujundamine aitab kasutajatel kiiresti aru saada nende mõistete loogikast. Et kasutada elemente, esmalt neid kirjeldatakse valdkonnamudelis – ecore mudelis. Sellega jätkus minu töö pärast valdkonnamudeli täiendamist. [13]

Kirjeldamine toimub Eclipse Sirius runtime keskkonnas, kuskohas *description* kataloogis asub selle jaoks oluline fail - *.odesign* (käesoleva töö puhul see on *valueExchange.odesign*). Selles failis kirjeldatakse diagramme, elemente, tööriista ja muud.

Sirius Specification Editori abil on loodud vajalikud elemendid ja elementidega seotud tööriistad. Näiteks kui esmalt tegelesin Value Exchange diagrammiga (valueExchangeDgm), oli loodud ja kirjeldatud *ActorNode*, kuna *Actor* või tegutseja on kõige oluline osa väärtusvahetuse protsessis. Edasi on tehtud sellele ActorNode'ga seotud tööriist *Node Creation createActorNode*, mille sees on kirjeldatud ka name ja id (joonis 6). Nii on tehtud kõik elemendid, mis on kirjeldatud järgmistes peatükkides.

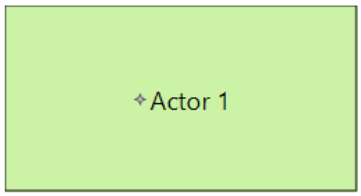


Joonis 6. Objektide kirjeldamine runtime keskkonnas

Oluline on teada, et elementide kuju on lihtne muuta ning on võimalik leida sobivaid kujundeid. Käesoleva töö käigus on kasutatud standartkujundeid.

4.2.1 Actor ehk Tegevuses osaleja element

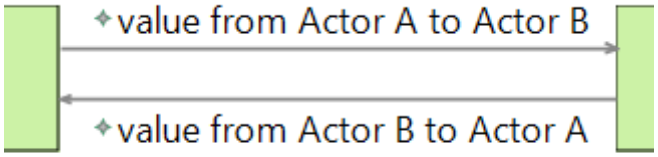
Actor on majanduslikult sõltumatu olem, näiteks tarbija või ettevõte, isik või organisatsioon. Diagrammis näeb vastav element töös pakutud esmases lahenduses välja nagu heleroheline nelinurk *Actor'*i nimega. [14]



Joonis 7. Tegevuses osaleja kujutamine protsessi koostamise prototüübis

4.2.2 ValueExchange ehk Väärtusvahetus element

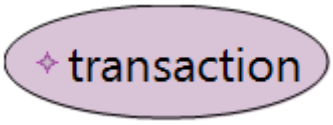
Väärtusvahetuse element koosneb väärtusest, mis on kirjutatud noole peal või all. Väärtus näitab, mis konkreetne asi läheb ühest Actorist teisele. Väärtuseks saab olla teenus, toode, raha või näiteks kogemus. Nool aga näitab suunda, kellest kellele läheb väärtus. [14]



Joonis 8. Väärtusvahetuse kujutamine protsessi koostamise prototüübis

4.2.3 Transaction ehk Transaktsioon element

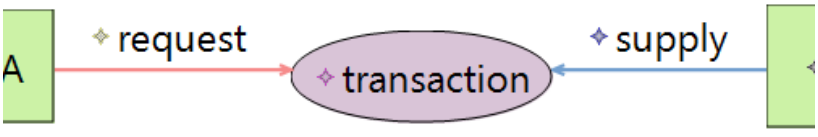
Transaction element on kommunikatsiooniakt täpselt kahe osapoole vahel. Üks *Actor* on selle äritransaktsiooni tellija ja teine *Actor* on täitja. Visuaalselt kujutatakse seda diagrammil tihti ovaalina.



Joonis 9. Transaktsiooni kujutamine protsessi koostamise prototüübis

4.2.4 Request ja Supply ehk Nõue ja Pakkumine elemendid

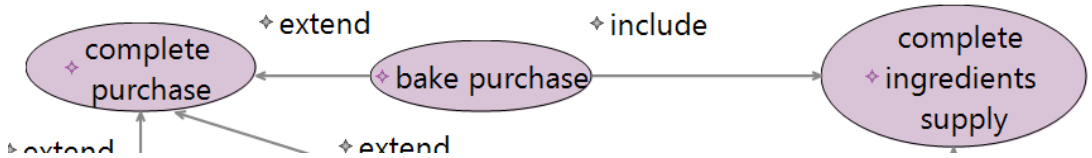
Äritransaktsioonide puhul alati on objekt, kes soovib midagi saada ja objekt, kes midagi pakub. Selle jaoks on loodud vastavalt elemendid *Request* ja *Supply*, mis töötavad koos Transaction'iga. Mõlemaid olemeid kujutatakse diagrammil noolena. Request nool läheb Tellijast Täitjale ja näitab, et üks Actor nõuab tegevuse täitmist teisest Actor'ist. Supply nool aga läheb vastupidi Täitjast Tellijale ning näitab, et Täitja teostab toimngu, mis on kirjutatud Transaction elemendis.



Joonis 10. Request ja Supply kujutamine protsessi koostamise prototüübis

4.2.5 ExtendConnection ja IncludeConnection ehk Sisaldamise ühenduse elemendid

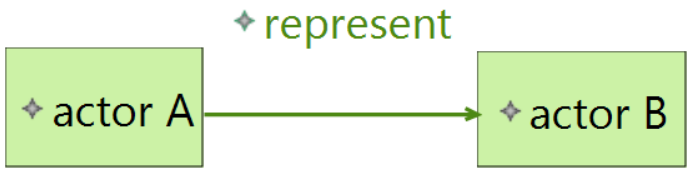
ExtendConnection ja *IncludeConnection* elemendid on tekkinud, et näidata seoseid transaktsioonide vahel. Kui on eesmärk teha diagrammi tavakasutajatele arusaadavam, siis võib kasutada ainult IncludeConnection seose. Siis noole tuleb vastupidi panna extend noole võrreldes. Erinevus on ainult selles, et include tähendab teise protressi või teenuse lülitamise käesoleva protsessiga. Extend aga tähendab, et alamprotsess lõikab suurema protsessi tükki välja ja lisab detailsust.



Joonis 11. ExtendConnection ja IncludeConnection kujutamine protsessi koostamise prototüübis

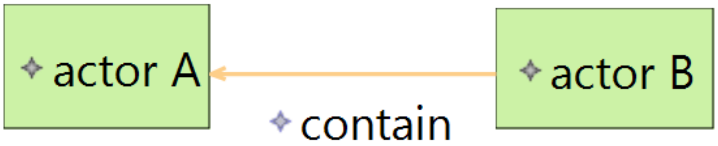
4.2.6 Specialize ja Composition ehk Spetsialiseerumine ja Kompositsioon elemendid

Specialize element näitab, et üks Actor esindab teist ning nool läheb esindajast ja noole peal on kirjutatud represent.



Joonis 12. Specialize kujutamine protsessi koostamise prototüübis

Composition olem ka on kujutatud noolena ja seda kasutatakse kui on vaja näidata, et üks Actor sisaldab teist.



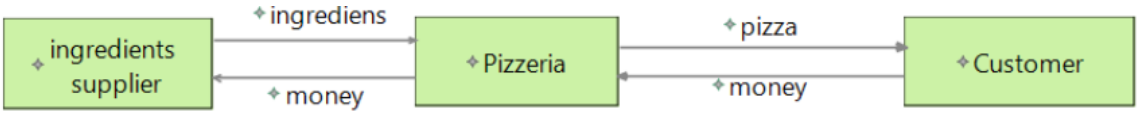
Joonis 13. Composition kujutamine protsessi koostamise prototüübis

4.3 Loodud alamkeele elementide kasutamise näited

Pärast töö teemaks oleva ärianalüüsi alamkeele elementide kirjeldamist on loodud 6 diargammit, kus on nähtav, kuidas neid olemeid saab kasutada. Näite teemaks on valitud pizzeria ja selle sees toimuvad protsessid ja tegutsevad osalejad. *ValueExchangeSimpleDgm* diagrammis osalevad ainult kaks osalejat , kelle vahel tehakse kaks väärtusvahetust. *ActorStructureDgm* diagrammis on näidatud suhted Actor'ite vahel. *ValueExchangeDgm* on esimese diagrammi täienenud versioon, kus on rohkem osalejaid. *BusinessTransactionSimpleDgm* kujutab lihtsat skeemi Actorite ja Transaction'idega, mille täienenud versioon on *businessTransactionDgm* diagramm, mis on keerulisem. *BusinessProcessStructureDgm* kirjeldab seoseid transaktsioonide vahel.

4.3.1 ValueExchangeSimpleDgm diagramm

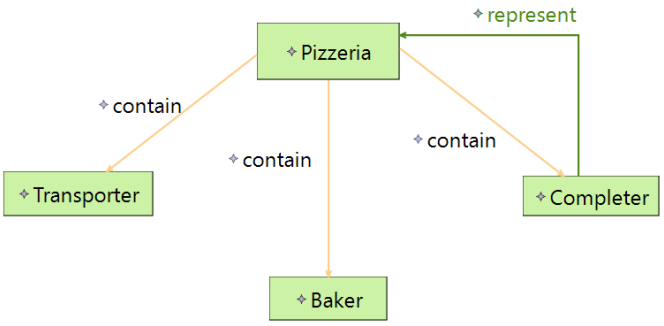
Käesoleval skeemil on näidatud lihtne seos osalejate vahel, kes on ühendatud väärtusvahetustega. Pizzeria on seotud Ingredients supplier'iga nii, et Pizzeria koostisosade vastu annab raha. Aga Customer'ile Pizzeria annab toodet ehk pizzat ausa tööga saadud raha vastu. Nissuguseid seoseid saab näidata ValueExchange elemendi abil.



Joonis 14. ValueExchangeSimpleDgm diagramm

4.3.2 ActorStructureDgm diagram

Selle skeemi vajadus tekkis sellepärast, et osalejaid ja nende seoseid lähemalt uurida. *ActorStructureDgm* diagrammis on kujutatud Pizzeria actor, mis sisaldab Transporter, Baker ja Completer osalejaid. Omakorda Completer esindab Pizzeriat. Sellega isegi tavakasutaja saab aru, et Pizzeria on suur ettevõte, kus töötavad ning kuhu kuuluvad inimene, kes teeb pizzat ehk Baker, inimene, kes toob pizzat kliendile ehk Transporter ja klienditeenindaja, kes suhtleb klientide, kulleri ja kokaga ja esindab niimodi pizzeriat. Nissuguseid seoseid saab näidata *Composition* ja *Speialize* elementide abil.



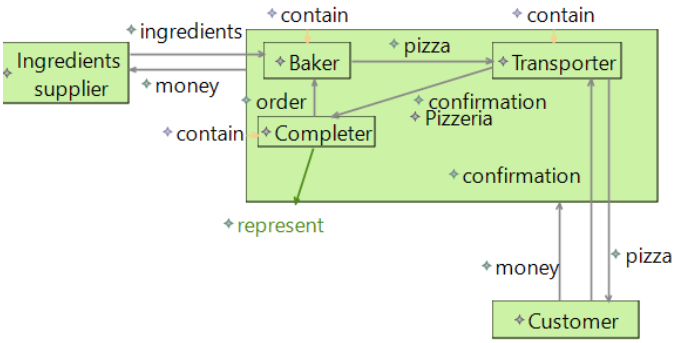
Joonis 15. ActorStructureDgm diagramm

4.3.3 ValueExchangeDgm diagramm

Käesolev skeem on esimese ehk ValueExchangeSimpleDgm diagrammi täienenud versioon. Erinevus on selles, et nüüd on nähtavad pizzeria sees töötavad Actorid. Nende seos Pizzeriaga on näidatud *contain* väärtusega, mis kuulub Composition elemendile.

Nüüd, kui sisesed Actorid osalevad skeemis, väärtusvahetuste seoseid on ka muutunud.

Klient teeb tellimust suheldes klienditeenindajaga, kes esindab pizzeriat ning Customer ja Completer vahel on väärtusvahetus *order* ehk *tellimus*, mis läheb kohe kokale. Kokk teeb pizzat koostisosadest, mis ta sai Ingredients supplier'ist, aga raha nende eest on maksnud pizzeria. Valmistoodet kokk annab kullerile, misjärel kuller toob pizzat kliendile ja saab kinnitust näiteks allkirjana (käsitleme juhtu, kui maksmine toimub kaardiga või pizzeriakontole, vastasel juhul klient annab kullerile ka raha). Kinnitust kuller saadab klienditeenindajale, tellimus on täidetud ning raha läheb kliendiarvelt pizzeriaarvele.

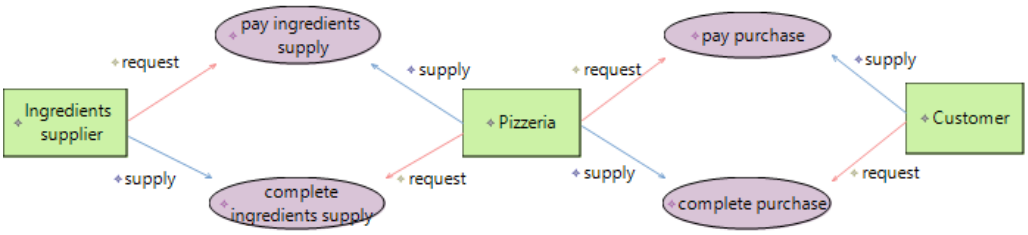


Joonis 16. ValueExchangeDgm diagramm

4.3.4 BusinessTransactionSimpleDgm diagramm

BusinessTransactionSimpleDgm diagramm kujutab lihtsat skeemi Actorite ja Transaction'idega, kus nende vahel seoste näitamise jaoks on kasutatud RequestConnection ja IncludeConnection.

Kui pizzeria soovib osta koostisosasid ehk *palub, et tarne toimiks*, siis selle asemel ta annab lubadust raha maksta. Koostisosade tarnija siis vastupidi palub pizzeriat maksta raha oma toodangu eest, mis ta pakub pizzeriale. Sel ajal pizzerial on veel seos kliendiga, kes soovib pizzat (request complete purchase) ja pakub selle eest raha.



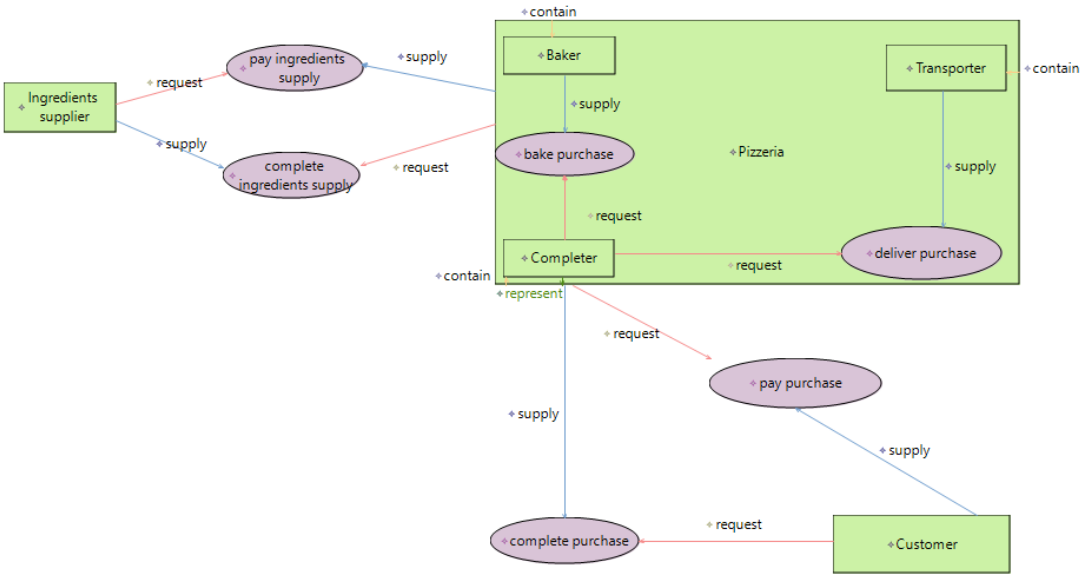
Joonis 17. BusinessTransactionSimpleDgm diagramm

4.3.5 BusinessTransactionDgm diagramm

Käesoleva diagrammi erinevus eelmisega on selles, et siin on kirjeldatud äritransaktrioonid siseste objektide vahel.

Pärast klientide tellimuse avaldamist klienditeetindaja palub kokka pizzat valmistada ja kokk pakub seda teenust. Edasi klienditeenindajalt tuleb nõue pizzat transportidakliendile

ja kuller saab seda teha. Ülejaanud äritransaktsioonid jäävad samaks nagu BusinessTransactionSimpleDgm diagrammis.

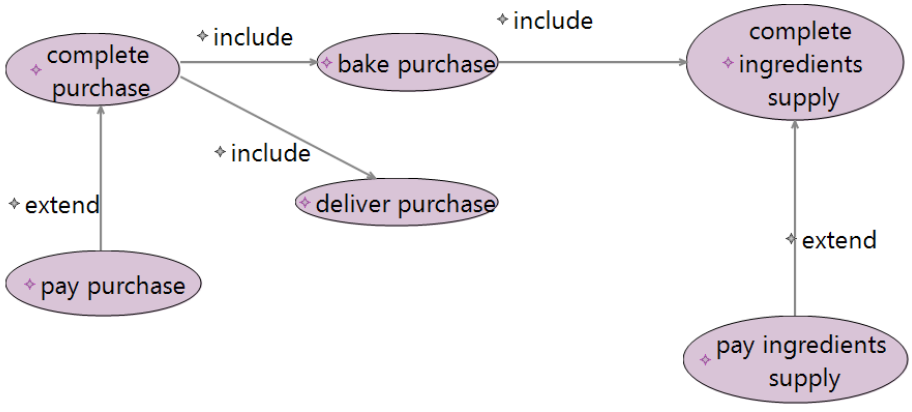


Joonis 18. BusinessTransactionDgm diagramm

4.3.6 BusinessProcessStructureDgm diagramm

Businessprocessstructuredgm diagramm näitab lähemalt struktuuriseoseid äritransaktsioonide vahel (mitte nende ajalise toimumise järjekorda).

Kõige suuremaks protsessiks on complete purchase, mis koosneb küpsetamisest ja kohaletoimitamisest ja mille eest maksmise sammu laiendab pay purchase. Küpsetamine omakorda sisaldab koostisosade tarne tehingut, mille eest maksmise sammu laiendab tarne eest maksmine (pay ingredients supply).



Joonis 19. 4.3.6 BusinessProcessStructureDgm diagramm

4.4 Prototüübi integreerimine eelnevalt kaitstud töösse

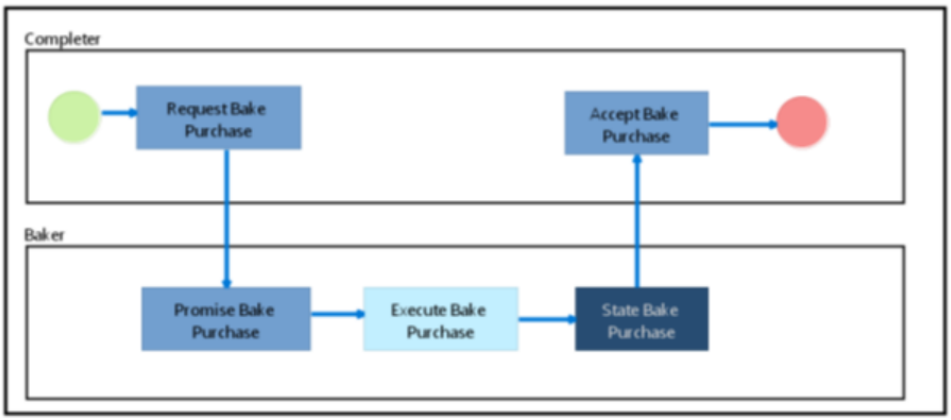
Kui kõik elemendid on kirjeldatud ja on tehtud nende kasutamise näited võib integreerida neid töösse, mis on tehtud eelnevalt. Pärast integreerimist sai võimalikuks lisada joonisel 20 illustreeritud lihtne stsenaariumi diagramm äritransaktsiooni bake purchase kohta. Üldisemalt väljendudes: nüüd on võimalik loodud prototüübi abil kirjeldada sobiv stsenaarium ükskõik millise mudelis esindatud äritransaktsiooni või muud tüüpi protsessi kohta.

4.4.1 Integreerimise protsess

Integreerimise jaoks on stsenaariumi diagrammitüüpi kirjeldav fail *scenario.odesign, mis sisaldab process viewpont'i* ehk eelnevas töös tehtud elemente, pandud eclipse sirius modelleerimise projekti description kataloogi, kus asub ka käesoleva töö põhitulemust hoidev fail valueexchange.odesign. Nii saavad mõlemas failis sisalduvad *viewpoint*-id töötada koos eelnevalt (vt.) modifitseeritud ja nüüd ühise *ecore* metamudeli alusel ja nende abil sai nüüd teha järgneva protsessidiagrammi bake purchase äritransaktsiooni jaoks.

4.4.2 Bake purchase äritransaktsiooni protsessidiagramm

Uurimise all olev protsess on *Bake purchase* ehk *küpsetamine*. On toodud lihtne näide, kuna töö käigus rohkem keskenduti tööriista arendamisele ja näidete katsetamisele. Kahe prototüübi liitmise tulemusena koostati kõnealune protsessikirjeldus.



Joonis 20. Bake purchase Process diagramm

Protsess hõlmab endas klienditeenindaja kui ka koka poolt tehtavaid tegevusi, mis nad teevad, et kliendile toodet pakkuda.

5 Järeldused

Käesolevas peatükkis kirjeldatakse tehtud töö tulemuste analüüsi, sealhulgas analüüsitakse modelleerimiskeele elementide kujusid ja prototüübi kasutamist. Räägitakse tellija poolt realiseerivast modelleerimis ökosüsteemist, eelneva ja käesoleva tööde võimalusest ja selle ökosüsteemi edasistest suundadest. Viimasena on toodud mis jäi töös tegemata.

5.1 Tulemuste analüüs

Käesoleva töö käigus arendati prototüüp, milllega on võimalik väärtusvahetuste protsesse kirjeldada. Kõik elemendid ja nende koostöö on katsetatud ning on tehtud kasutamise näided. Pärast prototüübi arendamist töö sisse integreeriti eelnevalt tehtud tööd, tänu millele sai võimalikus töös (joonisel 20 esitatud) Bake purchase protsessidiagrammi tegemine. Kõik enne püstitatud eesmärgid on saavutatud.

5.1.1 Modelleerimiskeele elementide kujundite analüüs

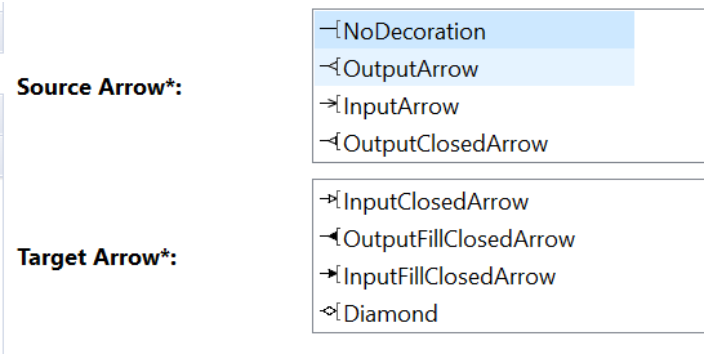
Nagu peatükkis 4.2 oli kirjutatud, elementide kuju on lihtne muuta ning on võimalik leida sobivaid kujundeid. Käesoleva töö käigus on kasutatud standartkujundid, mis võib olla ei ole kõige sobivamad. Ka kujundite loomise käigus ei kuulunud palju aega värvidele. On kasutatud kerged värvid, mis natuke sobivad kokku. Suurte diagrammi loomisel ja paljude elementide kasutamisel, võib olla diagramm läheb liiga kirjaks ja sellepärast arusaamisele keeruliseks. Aga seda ei jõudnud teha, kuna töö eesmargid on keskendatud rohkem tööriista arendamisele ja katsetamisele.

Näiteks kuna Transaction peaaegu alati on kujundatud ovaalvormiga, ei ole mõtet Actori jaoks kasutada ringi või muud ringvormi, sest äritransaktsiooni diagrammis kõik läheb segasemaks. Nii saab kasutada nurgelist kujundit nagu 22. joonisel või tavakasutajatele tavapärast kujundit nagu 21. joonisel. Tegelikult kujundite asemel saab panna igasugust sobivama pilti, ainult märge on, et pilt peab olema *.svg* formaadis. [16]

Joonis 21. Actor'i võimalik nurgeline kuju

Joonis 22. Võimalik Actor'i kuju

Ühendavale elemendile saab leida ka mitmeid erinevaid võimalusi. Peale värvi ja laiuse, saab muuta ka noole algus- või lõppukujundit. Eclipse Sirius pakub palju võimalikke variante (osa neist on näidatud 23. joonisel).



Joonis 23. Ühendava elemendi kujundite muutmise võimalused

5.1.2 Prototüübi kasutamise analüüs

Töö käigus arendati prototüüp, väärtusvahetuste protsesse kirjeldamise eesmärgiga.

Antud prototüüp töötab ilusasti, elemendid tekivad eelnevalt nendele antud nimedega, kui mõne elemendi kasutatakse mitu korda, väärtuse lõppu tekitavad numbrid vastavalt järjekorrale. Aga selle väärtuse või tema puuduse ikka saab muuta.

Kuna kogemust valdkonnaspetsiifilise keele modelleerimisega enne ei olnud, ei ole kogemust ka sellega töötava tarkvaraga, millepärast ei saa tehtud prototüübi võrrelda teiste tarkvaraga, mis keskenduvad äriprotsesside arendamisele.

Teoreetiliste andmete pohjal analüüsi tehes lähtub, et modelleerimistarkvarad on universaalsed tooted, mis järgivad konkreetseid standardeid (BPMN, UML jne) ja nendega saab teha kindlat tüüpi diagramme täpselt nii, nagu standardid ette näevad ja tarkvara tootjad neid standardeid on tõlgendanud, mitte rohkem ega vähem. Tegelikus arendustöös on tihti vaja erinevaid standardeid või nende osasid kombineerida (näiteks, BPMN + UML klassidiagramm), lihtsustada ja muuta, see tähendab - (arendus)situatsiooni jaoks kohandada. Eksisteeruvad modelleerimistarkvarad tihti ei pakku niisugust võimalust või nende kasutamine on ebamugav. Käesoleva töö ja prototüübi vajadus tekkis just sellisel olukorral, sest tellijal on vaja kombineerida ja lihtsustada (vajadusel ka uute omadustega täiendada) konkreetseid komponente (diagrammitüüpe) erinevatest standarditest (UML, BPMN, ArchiMate, e3-Value, DEMO).

5.2 Arenev modelleerimise ökosüsteem

Ettevõtte modelleerimiseks on vaja kas ühte suurt ja väga universaalset keelt (UML, ArchiMate) või paljude väiksemate konkreetsemate valdkonnaspetsiifiliste keelte ajas arenevat ökosüsteemi. Aine jaoks, mille vajadusest käesolev töö on kirjutatud, nüüd on olemas kolmet valdkonnaspetsiifilisest keelest koosnev ökosüsteem - lihtsustatud BPMN, lihtsustatud ValueExchange, lihtsustatud Business or Economic Transactions. Ökosüsteemi on vaja arendada kuni on valmis prototüüp või tarkvara, mille kasutamine vastaks magistriaine vajadusele.

5.2.1 Eelneva ja käesoleva tööde koostöö võimalused

Esimene töö, mis on tehtud selle ökosüsteemi arendamiseks, on eelmine töö, mille tulemuseks on prototüüp, mille abil saab äriprotsesside ajas kulgemist kirjeldada. Käesoleva tööga liitmisel on võimalus käsitleda ka protsesside, nendes osalevate tegelaste ning nende käigus toimuvate väärtusvahetuste ja väärtustegevuste struktuuri ja seda saab teha mitmes erinevas vaates.

Näiteks pizzeria näides saab mitte ainult näidata, et pizzat teeb küpsetaja, annab kullerile, kes annab seda kliendile, vaid kirjutada, mida klient annab kullerile tagasi või mis tegevus ühendab küpsetajat koostisosade tarnijaga ehk kirjeldada väärtusvahetusi ja äritransaktsioone.

5.2.2 Ökosüsteemi edasised suunad

Ettevõtte modelleerimiseks vajaliku modelleerimiskeelte ökosüsteemi väljaarendamiseks on vaja palju tööd teha ja mitmeid väiksemaid alamkeeli lisada. Järgmine tudeng juba laiendab seda ökosüsteemi veel kahe keelega: domeenimudeli (lihtsustatud UML klassidiagramm) ja elutsükli mudeliga (lihtsustatud UML olekudiagramm), mis võimaldavad näidata skeemis ärivaldkonna klasside ehk põhimõistete struktuuri ja objektide olekute muutumist.

Edasi saab liikuda mitmetes erinevates suundades, mida platvorm võimaldab. On võimalik arendada mudeliteisendusi (Model Transformations), selhulgas näiteks väärtusvahetuste mudeli automaatne teisendamine äritransaktsioonide mudeliks. See võiks olla aluseks järgmiste lõputööde tegemiseks paljudele tudengitele.

5.3 Tehtud töö edasiarendamine

Kõik püstitatud eesmärgid on saavutatud, prototüüp töötab, teda on kasutatud ja varem kaitstud lõputöö tulemusega integreeritud. Edasiarendamisel saab sellele liita juurde nii väärtusvahetuste modelleerimise kui ka äritransaktsioonide modelleerimise detailsemaid omadusi. Näiteks saab väärtusvahetuste mudelit täiendada tegelaste väärtuspakkumisteja väärtusliideste mõistetega väärtusvahetuste modelleerimise teooriast [15].

6 Kokkuvõte

Bakalaureusetöö „Ärianalüüsi toetava valdkonnaspetsiifilise modelleerimiskeele prototüübi loomine“ lahendab probleemi, et töö tellija poolt magistriainetes õpetatavat ettevõtte modelleerimise metoodikat on vaja toetada sobivate vabavaraliste tööriistadega, mis peab katma ära kõik vajalikud vaatenurgad ja standardid ja võimaldama metoodika arengule vastavalt uute meetodite ja tehnikatega täiendada. Mudelipõhise arenduse põhimõtete abil oli eesmärgiks võetud Eclipse Sirius platvormi abil valdkonnaspetsiifilise modelleerimiskeele prototüübi loomine ettevõtte äritransaktsioonide ning väärtusvahetuste modelleerimiseks.

Tuginedes MDA põhimõtetele, valdkonnaspetsiifilise keele ja modelleerimise keele teoreetilise alusele on arendatud prototüüp väärtusvahetuste ja äritrasaktsioonide modelleerimiseks. Prototüübi aluseks oli valdkonnamudel, mis kirjeldati teise lõputöö tulemuseks olnud äriprotsesside valdkonnamudelit modifitseerides. Saadud valdkonnamudelile toetudes kirjeldati väärtusvahetuste ja äritransaktsioonide mudelite elemente, nendevahelisi võimalikke seoseid ja elementide kujundust diagrammis. On läbi viidud integreerimine eelnevalt tehtud töösse, mille tulemuse demonstreerimiseks loodi ühe äritransaktsiooni põhistsenaariumit illustreeriv protsessidiagramm.

On näidatud variante, kuidas diagrammi ja selle elementide kujundust saab teha ilusamaks ja arusaadavamaks, kuna tööriistale keskendudes ei jõutud seda veel teostada. järelduste peatükis on ka kirjeldatud käesoleva töö tulemuse asukohta ettevõtte modelleerimise alamkeelte suuremas pildis ehk ökosüsteemis ja tehtud prototüübi analüüs teoreetiliste aluste põhjal.

Püstitatud eesmärk on saavutatud, kuna ärianalüüsi toetav valdkonnaspetsiifilise modelleerimiskeele prototüüp sai tehtud, integreeritud ja näide põhjal katsetatud.

Kasutatud kirjandus

[1] I. Stockley, “Rethinking value exchange”, [Võrgumaterjal]
Available: https://www.mycustomer.com/experience/voice-of-the-customer/rethinking-value-exchange-consumers-trust-us-with-their-data-how-do

[2] P.Gleeson, “Business Transaction Definition & Examples”, [Võrgumaterjal]
Available: https://smallbusiness.chron.com/business-transaction-definition-examples-25244.html

[3] S. Alhir, “Understanding the Model Driven Architecture”, [Võrgumaterjal]
Available: https://www.methodsandtools.com/archive/archive.php?id=5

[4] F. Truyen, “The Fast Guide to Model Driven Architecture”, [Võrgumaterjal]
Available: https://www.omg.org/mda/mda_files/Cephas_MDA_Fast_Guide.pdf

[5] F. Tomassetti, “The complete guide to (external) Domain Specific Languages”, [Võrgumaterjal]
Available: https://tomassetti.me/domain-specific-languages/

[6] „Definition of a Domain-Specific Modelling Language“, [Võrgumaterjal]
Available: https://www.miuc.org/definition-of-a-domain-specific-modelling-language/

[7] Eclipse, “About the Eclipse Project”, [Võrgumaterjal]
Available: https://www.eclipse.org/eclipse/

[8] Eclipse, “About the Eclipse Foundation”, [Võrgumaterjal]
Available: https://www.eclipse.org/org/

[9] Eclipse, “About Eclipse Sirius”, [Võrgumaterjal]
Available: https://www.eclipse.org/sirius/doc/

[10] Eclipse, “The easiest way to get your own Modeling Tool”, [Võrgumaterjal]
Available: https://www.eclipse.org/sirius/

[11] L. Vogel “About the Eclipse Modeling Framework Tutorial”, [Võrgumaterjal]
Available: https://www.vogella.com/tutorials/EclipseEMF/article.html

[12] Ecipse, “Eclipse Modeling Framework (EMF)”, [Võrgumaterjal]
Available: https://www.eclipse.org/modeling/emf/

[13] Amit, “Business Process Modeling: Definition, Benefits and Techniques”, [Võrgumaterjal]
Available:https://tallyfy.com/business-process-modeling/

[14] InterOP, “Interoperability Research for Networked Enterprises Applications and Software”, 2006

[15] Enterprise Architecture: Creating Value by Informed Governance, Authors: Op't Land, M., Proper, E., Waage, M., Cloo, J., Steghuis, C, 2009

[16] Eclipse Fpundation, “Eclipse Sirius Starter Tutorial”, [Võrgumaterjal]
Available: https://wiki.eclipse.org/Sirius/Tutorials/StarterTutorial

[17] The value engineers, „How to model and analyze a value model”, [Võrgumaterjal] Available: https://www.thevalueengineers.nl/tutorials/model-analyze-value-model/

[18] “Beginners Guide to Business Process Modeling”, [Võrgumaterjal] Available: https://www.smartsheet.com/beginners-guide-business-process-modeling

[19] U.Frank «Multi-Perspective Enterprise Modeling: Foundational Concepts, Prospects and Future Research Challenges», 2012