KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACINIŲ SISTEMŲ INŽINERIJOS STUDIJŲ PROGRAMA

ANDRIUS SROGIS

AUTOMATIZUOTAS GRAFINIO MODELIO PERFORMULAVIMAS Į NATŪRALIĄ KALBĄ

Magistro darbas

Darbo vadovas
doc. R. Butkienė

KAUNAS, 2013

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACINIŲ SISTEMŲ INŽINERIJOS STUDIJŲ PROGRAMA

ANDRIUS SROGIS

AUTOMATIZUOTAS GRAFINIO MODELIO PERFORMULAVIMAS Į NATŪRALIĄ KALBĄ

Magistro darbas

Darbo vadovas:
doc. R. Butkienė
2013-05-22

Recenzentas:
dr. Š. Packevičius
2013-05-22

Atliko:
IFM-1/4 gr. studentas
Andrius Srogis
2013-05-22

KAUNAS, 2013

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Santrauka

Automatizuotas grafinio modelio performulavimas į natūralią kalbą

Grafinių modelių projektavimas yra plačiai naudojamas tiek mokslo, tiek verslo srytyse. Pasaulyje naudojama įvairių kalbų, skirtų tiek sistemų architektūrų, tiek verslo procesų projektavimui.

Daugumai kalbų yra sukurta įvairių įrankių, leidžiančių jų naudotojams projektuoti įvairius procesus ar statines sistemas.

Vienai labiausiai paplitusių kalbų (UML) trūksta metodikos ir įrankių, gebančių korektiškai perteikti natūralia kalba sistemų architektų aprašytus grafinius modelius asmenims, mažai kvalifikuotiems grafinių modelių sudaryme, skaityme. Perteikimas tuo natūralesnis ir labiau suprantamesnis, kuo jis artimesnis natūraliai kalbai.

Yra metodikų ir įrankių atliekančių grafinio modelio verbalizavimą, tačiau nėra koncentruotų ties diagramomis UML kalba, kurios geba formuoti ne tik statiką, bet ir dinamiką.

Pagrindinis darbo tikslas yra sukurti metodiką ir realizuoti įrankį, kuris gebėtų grafinį modelį išreikštą UML kalba performuluoti natūralia kalba.

Summary

Automated Reformulation of Graphical Model in Natural Language

The graphical model architecture design is widely used for scientific and enterprise purposes. There are many languages concentrated on enterprise processes and static systems designing.

One of the most popular modeling language (UML) is missing methodology and tools suitable for correct reformulation of graphical models (formulated by the UML) in natural language. The main purpose of the graphical model reformulation in natural language is to make models easier to understand for people whose are not specialized in UML.

Methodology and tool which is capable of reformulating graphical models in natural language already exists, but it isn't concentrated on UML or capable of reformulating static and dynamic processes.

The main goal of this work is to define a methodology and implement a tool, which would be capable of translating the graphical UML model to a natural language text.

Terminų ir santrumpų žodynas

SANTRUMPA	PAAIŠKINIMAS
Įrankis	Kompiuterinė programa atliekanti numatytas užduotis su duomenimis ar kt. informacija
UML	Modeliavimo kalba, naudojama objektiškai orentuotame projektavime
ER	Esybių – ryšių modeliavimo požiūris
OP	Objektinis modeliavimo požiūris
Verbalizavimas	Grafinio modelio perteikimas (performulavimas) natūralia kalba
ORM-ML	Objektų – rolių modeliavimo žymėjimo kalba
XML	Bendros paskirties duomenų struktūrų bei jų turinio aprašomoji kalba
Metaduomenys	Duomenys apie duomenis
Predikatas	Objekto požymis
MagicDraw	UML modeliavimo kalbos diagramų kūrimo įrankis

Įvadas

Grafinių modelių projektavimas yra plačiai naudojamas tiek mokslo, tiek verslo srityse. Vieni jų – sistemų architektūros projektavimui, kiti – verslo procesų valdymui. Pasaulyje naudojama įvairių kalbų, skirtų tiek sistemų architektūrų, tiek verslo procesų projektavimui.

Aktualumas. Dažnas sistemų architektas, analitikas ar kitas asmuo, susijęs su idėjų perteikimu grafiniais modeliais naudoja įrankius, leidžiančius jiems realizuoti savo idėją. Kiekvienas jų (įrankių) skirtas konkrečios kalbos specifikai (pvz., UML [3]). Specializuoti įrankiai padeda sisteminti, spartinti modelių kūrimo darbą, dažnais atvejais jų pagalba tiesiog išvengiama kritinių klaidų.

Didžioji dalis įrankių turi vartotojo sąsają, kurios pagalba konkrečios sryties specialistas ar eilinis vartotojos gali kontroliuoti bei stebėti įrankio (naudojamos sistemos) veiklą.

Problema. Vienai labiausiai paplitusių kalbų (UML) trūksta metodikos ir įrankių, gebančių korektiškai perteikti natūralia kalba sistemų architektų aprašytus grafinius modelius asmenims, mažai kvalifikuotiems grafinių modelių sudaryme, skaityme. Perteikimas tuo natūralesnis ir labiau suprantamesnis, kuo jis artimesnis natūraliai kalbai.

Darbo tikslas. Sukonstruoti UML (Universal Modeling Language) diagramų, aprašytų MagicDraw įrankiu, formavimo taisykles lietuvių kalba, kurios leistų analitikams ar architektams (taip pat studentams) kuriantiems įvairių sričių veiklos sistemų grafinius modelius UML kalba, patikslinti suformuotą modelį ar jį pateikti naudotojui verbalizuota forma.

Tariamojo darbo siekiamas sprendimas – sukurti grafinio modelio lietuviškų sakinių formulavimo taisykles (žodyną), pagal kurį realizuotas įrankis generuos korektiškai suformuluotus lietuviškus sakinius, apibrėžiančius objektų sąryšius tarp pateikto grafinio modelio.

Sprendžiami uždaviniai.

1. Išanalizuoti pasirinktų grafinių modelių išreikštų UML kalba sąryšių tarp objektų panaudojimą sakinių išreiškimui natūralia kalba;
2. Pateiktų sąryšių verbalizavimo taisyklių sukūrimas;

3. Grafiniame modelyje sąryšiais siejamų objektų verbalizavimas juos apjungiant su sąryšiams naudojamų taisyklių visuma siekiant išgauti korektiškai suformuluotus natūralios kalbos sakinius, aprašančius pateiktą grafinį modelį išreikštą UML kalba;
4. Ištirti lietuvių kalbos linksniavimo panaudojimo galimybes grafinio modelio išreikšto UML kalba išreiškimui natūralia kalba.

Darbo rezultatai. Sudaryta metodika, aprašanti grafinio modelio išreikšto UML kalba performulavimo šablono panaudojimą modelio išreiškimui natūralia kalba. Realizuotas įrankis performuluojantis grafinį modelį (klasių diagramą) išreikštą UML kalba į natūralią kalbą.

Naujumas. Darbe aprašyta metodika grafinio modelio išreikšto UML kalba performulavimui natūralia kalba ir realizuotas įrankis klasių diagramos verbalizavimui objektiniu ir esybių – ryšių požiūriu įtraukiant linksniavimo lietuvių kalboje galimybę.

Darbo struktūra. Analitinėje dalyje aptariama grafinio modelio išreikšto UML kalba performulavimo galimybės natūralia kalba. Aptariamos dažniausiai naudojamas UML diagramos. Atliekama panašių metodikų ir sistemų analizė. Analitinės dalies pateikiamos analizės išvados.

Projektinėje dalyje specifikuojami reikalavimai, kurių pagrindu kuriamas grafinio modelio išreikšto UML kalba performulavimo į natūralią kalbą įrankis.

Realizacijos dalyje aprašomas kaip realizuotas grafinio modelio išreikšto UML kalba performulavimo į natūralią kalbą įrankis ir jo veikimas.

Eksperimentinėje dalyje atliekamas sukurto įrankio ir metodikos grafinio modelio performulavimui į natūralią kalbą eksperimentas.

1. Analizė

1.1. Tyrimo sritis, objektas ir problema

Tyrimo objektas – grafinio modelio išreikšto UML kalba išreiškimas natūralia kalba.

Tyrimo sritis – grafinio modelio išreikšto UML kalba išreiškimo į natūralią kalbą automatizavimas.

Tyrimo problema – trūksta metodikos ir įrankių, galinčių grafinį modelį išreikštą UML kalba performuluoti natūralia kalba.

1.2. Analizės tikslas

Analizės tikslas yra išsiaiškinti kaip galime sukurti arba panaudoti egzistuojančias priemones grafinio modelio išreikšto UML kalba išreiškimo natūralia kalba automatizavimui . Tikslui pasiekti turi būti išspręsti tokie uždaviniai:

1. Ištirti modelio išreikšto UML kalba formulavimo natūralios kalbos sakiniais galimybes;
2. Atlikti technologijų, įrankių, kurie galėtų būti panaudoti modelio išreikšto UML kalba išreiškimui natūralia kalba automatizavimui analizę.

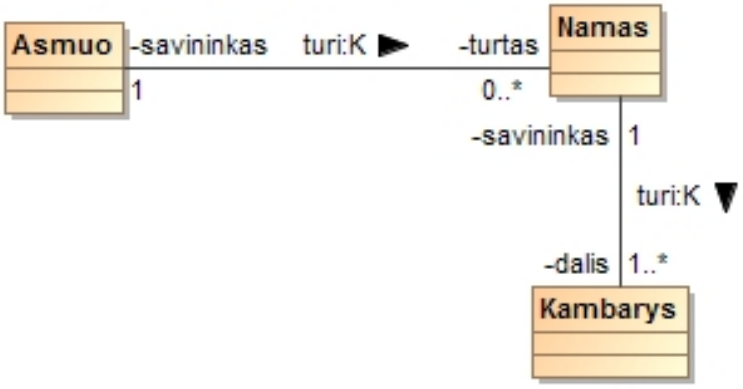
1.3. Modelio išreikšto UML kalba išreiškimo natūralia kalba analizė

Šiame skyriuje aptarsime bendrąsias taisykles grafiniams modeliams verbalizuoti. UML kalboje numatytos tokio tipo diagramos: klasių, panaudos atvejų ir veiklos. UML [3] metodika apima didesnį kiekį diagramų, tačiau tyrime pasirinktos aktualiausios informacinių sistemų (daugiausia registrų) kūrimui naudojamos diagramos.

Pati UML kalba yra priskiriama kalbai, kuri naudojama sistemoms modeliuoti prisilaikant objektinio požiūrio. Objektinio ir struktūrinio požiūrio įtaka dalykinės srities koncepcinių modelių sudarymui liečia tik vieno tipo diagramas – klasių arba esybių – ryšių.

Jei modeliuojant sistemos statinius aspektus laikomasi gryno objektinio požiūrio, o modeliams išreikšti naudojama UML kalba, tai asociacijos elementai įvardijami taip: „savininkas", „turtas" – vaidmenys, „turi:K" – veidmenų susiejimas veiksmažodine forma kilmininko linksnyje.

Objektinis požiūris



1.1 pav. Objektinis požiūris

Žemiau lentelėje (žr. 1.1 lentelė) pateikiamas aukščiau pateikto (žr. 1.1 pav.) modelio pagal objektinį požiūrį išreiškimo į natūralią kalbą rezultatas.

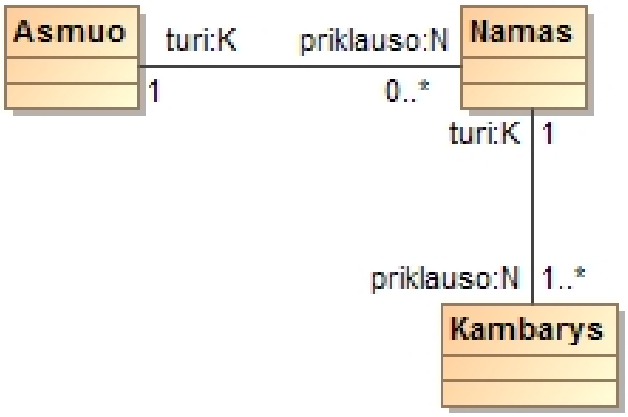
1.1 lentelė. Objektinio požiūrio išreiškimas natūralia kalba

Nr.	Išreiškimas natūralia kalba
1.	Asmuo turi daug namų. Savininkas turi daug turto. Savininkas yra asmuo. Turtas yra namas.
2.	Namas turi daug kambarių. Savininkas turi daug dalių. Savininkas yra namas. Dalis yra kambarys.

Tačiau modelio, sudaryto laikantis šio požiūrio, verbalizavimas yra nepilnavertis – nėra galimybės vienareikšmiškai perskaityti tos pačios asociacijos iš abiejų galų, kadangi trūksta dar vieno asociacijos vardo. Šis vardas gali būti numanomas (žmogaus), bet norint automatizuoti verbalizavimą, reikalinga vienareikšmė informacija (semantika apie dalykinę sritį).

Laikantis struktūrinio požiūrio [1] analogiškas sistemos aspektas gali būti sumodeliuotas laikantis, pavyzdžiui ER (esybių – ryšių) diagramų modeliavimo taisyklių. Šiuo atveju, asociacijos elementai įvardijami taip: „turi:K” – elemento „Asmuo” susiejimas su elementu „Namas” kilmininko linksnyje.

Esybių požiūris



1.2 pav. Esybių - ryšių požiūris

Žemiau lentelėje (žr. 1.2 lentelė) pateikiamas aukščiau pateikto (žr. 1.2 pav.) modelio pagal esybių požiūrį išreiškimo į natūralią kalbą rezultatas.

1.2 lentelė. Esybių - ryšių požiūrio išreiškimas natūralia kalba

Nr.	Išreiškimas natūralia kalba
1.	Asmuo turi daug namų.
2.	Daug namų priklauso asmeniui.
3.	Namas turi daug kambarių.
4.	Kambarys priklauso namui.

Iš pavyzdžio matyti, kad dabar ryšiai gali būti verbalizuoti skaitant juos iš abiejų ryšio galų. Tačiau taip modeliuojant nusižengiama UML modeliavimo taisyklėms, nes klasės vaidmens vardui naudojama veiksmažodinę frazė, kuri turėtų būti naudojama asociacijos vardui.

Modeliuoti UML laikantis tokio požiūrio yra nusižengimas UML standartui, bet tai leidžia perteikti grafinio modelio elementus abipusiškai.

1.3.1. Klasių diagramos verbalizavimas

Skyriuje aptariamas UML klasių diagramos [4] verbalizavimo natūralia kalba ypatybės.

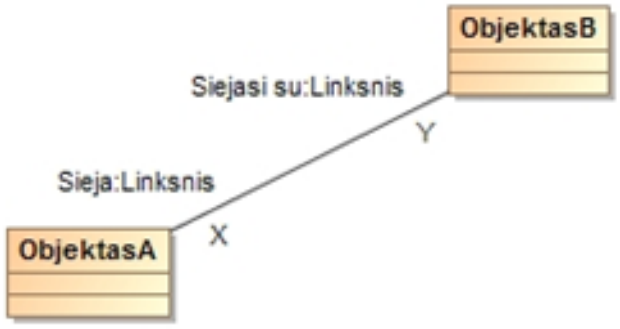
Asociacija

Klasių diagramai braižyti naudojami šie pagrindiniai elementai: klasė, asociacija. Klasių diagramoje verbalizavimo poreikis atsiranda susiejus dalykinės srities klases asociacijomis. Be specialių modeliavimo žinių ne specialistui suprasti ir tinkamai įvertinti sudarytą modelį yra sunku. Todėl asociacijų verbalizavimo funkcija būtų naudinga ir įneštų daugiau aiškumo apie modeliuotojo dalykinės srities supratimą.

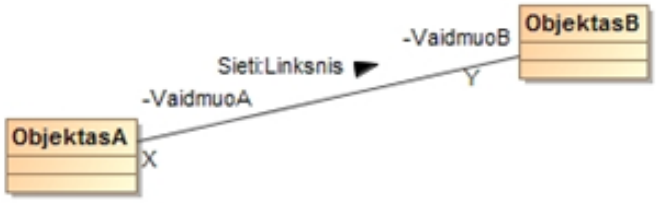
UML klasių diagramoje naudojamos kelių rūšių asociacijos: paprasta binarinė asociacija, agregavimas, kompozicija ir specializavimas. Toliau aptarsime variantus, kaip įvairios asociacijos gali būti verbalizuojamos. Kiekvienam atvejui bus pateiktas abstraktus modelio pavyzdys ir galimi verbalizavimo pavyzdžiai.

Paprasta binarinė asociacija

Žemiau pateiktoje lentelėje (žr. 1.3 lentelė) verbalizuojama grafiko dalis (žr. 1.3 pav. ir 1.4 pav.), kurioje naudojama asociacija.



1.3 pav. Asociacijos verbalizavimas esybių - ryšių požiūriu



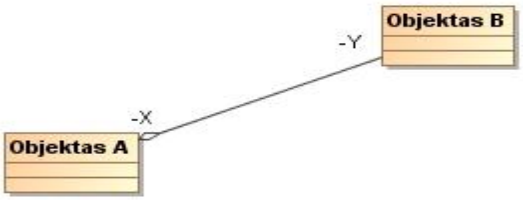
1.4 pav. Asociacijos verbalizavimas objektiniu požiūriu

1.3 lentelė. Asociacijos verbalizavimo ER ir OP požiūriu rezultatas

Nr.	X	Y	Verbalizavimas ER požiūriu	Verbalizavimas OP požiūriu
1	0..1	1	Objektas A sieja Objektą B; Objektas B gali sietis su Objektu A;	Objektas A sieja Objektą B; Vaidmuo A yra Objektas A; Vaidmuo B yra Objektas B;
2	0..1	1..*	Objektas A sieja daug Objektų B; Objektas B gali sietis su Objektu A;	Objektas A sieja daug Objektų B; Vaidmuo A yra Objektas A; Vaidmuo B yra Objektas B;
3	0..*	1	Daug Objektų A sieja Objektą B; Objektas B gali sietis su daugeliu Objektų A;	Daug Objektų A sieja Objektą B; Vaidmuo A yra Objektas A; Vaidmuo B yra Objektas B;
4	0..*	1..*	Daug Objektų A sieja daug Objektų B; Daug Objektų B gali sietis su daugeliu Objektų A;	Daug Objektų A sieja daug Objektų B; Vaidmuo A yra Objektas A; Vaidmuo B yra Objektas B;
5	1	1	Objektas A sieja Objektą B; Objektas B siejasi su Objektu A;	Objektas A sieja Objektą B; Vaidmuo A yra Objektas A; Vaidmuo B yra Objektas B;
6	1	1..*	Objektas A sieja daug Objektų B; Daugelis Objektų B siejasi su Objektu A;	Objektas A sieja daug Objektų B; Vaidmuo A yra Objektas A; Vaidmuo B yra Objektas B;
7	1..*	1..*	Daug Objektų A sieja daug Objektų B; Daug Objektų B siejasi su daugeliu Objektų A;	Daug Objektų A sieja daug Objektų B; Vaidmuo A yra Objektas A; Vaidmuo B yra Objektas B;
8	0..1	0..1	Objektas A gali sieti Objektą B; Objektas B gali sietis su Objektu A;	Objektas A gali sieti Objektą B; Vaidmuo A yra Objektas A; Vaidmuo B yra Objektas B;
9	0..1	0..*	Objektas A gali sieti Objektą B; Objektas B gali sietis su Objektu B;	Objektas A gali sieti Objektą B; Vaidmuo A yra Objektas A; Vaidmuo B yra Objektas B;
10	0..*	0..*	Daug Objektų A gali sieti daug Objektų B; Daug Objektų B gali sietis su daugeliu Objektų A;	Daug Objektų A gali sieti daug Objektų B; Vaidmuo A yra Objektas A; Vaidmuo B yra Objektas B;

Agregavimas

Žemiau pateiktoje lentelėje (žr. 1.4 lentelė) verbalizuojama grafiko dalis (žr. 1.5 pav.), kurioje naudojamas agregavimas.



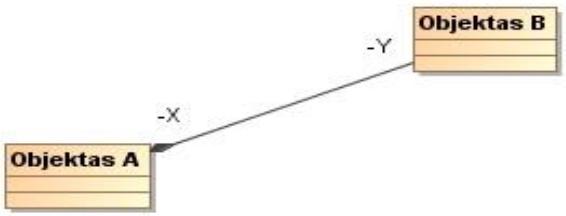
1.5 pav. Agregavimas

1.4 lentelė. Agregavimo verbalizavimo rezultatas

Nr.	X	Y	Verbalizavimas
1	1	0..1	Objektas A gali turėti Objektą B; Objektas B priklauso Objektui A;
2	1	0..*	Objektas A gali turėti daug Objektų B; Objektas B priklauso Objektui A;

Kompozicija

Žemiau pateiktoje lentelėje verbalizuojama grafiko dalis (žr. 1.6 pav.), kurioje naudojama kompozicija (žr. 1.5 lentelė).



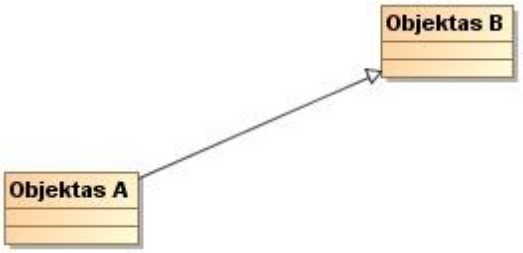
1.6 pav. Kompozicija

1.5 lentelė. Kompozicijos verbalizavimo rezultatas

Nr.	X	Y	Verbalizavimas
1	1	1..*	Objektas A turi daug Objektų B; Daug Objektų B priklauso Objektui A;
2	1	1	Objektas A turi Objektą B; Objektas B priklauso Objektui A;

Apibendrinimas – specializavimas tarp klasių

Žemiau pateiktoje lentelėje (žr. 1.6 lentelė) verbalizuojama grafiko dalis (žr. 1.7 pav.), kurioje naudojama generalizacija.



1.7 pav. Apibendrinimas – specializavimas

1.6 lentelė. Apibendrinimo - specializavimo verbalizavimo rezultatas

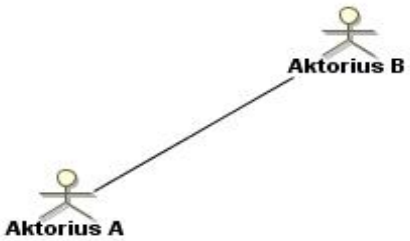
Nr.	Verbalizavimas
1	Objektas A specializuoja Objektą B;
2	Objektas B apibendrina Objektą A;

1.3.2. Panaudos atvejų diagramos verbalizavimas

Skyriuje aptariamas panaudos atvejų diagramos [5] performulavimo natūralia kalba ypatybės.

Asociacija tarp aktorių

Žemiau pateiktoje lentelėje (žr. 1.7 lentelė) verbalizuojama grafiko dalis (žr. 1.8 pav.), kurioje naudojama asociacija tarp aktorių.



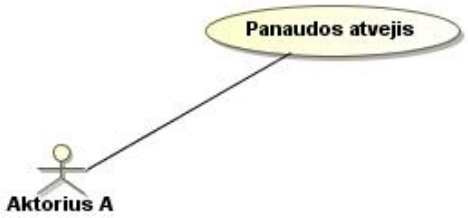
1.8 pav. Asociacija tarp aktorių

1.7 lentelė. Asociacijos tarp aktorių verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Aktorius A siejasi su Aktorius B;
2	Aktorius B siejasi su Aktorius A;

Asociacija tarp aktoriaus ir panaudos atvejo

Žemiau pateiktoje lentelėje (žr. 1.8 lentelė) verbalizuojama grafiko dalis (žr. 1.9 pav.), kurioje naudojama asociacija tarp aktoriaus ir panaudos atvejo.



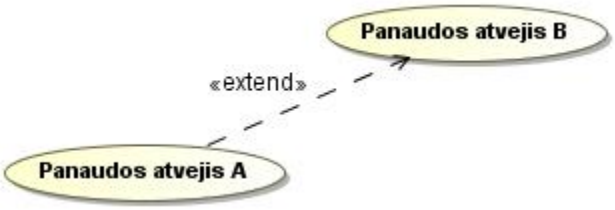
1.9 pav. Asociacija tarp aktoriaus ir panaudos atvejo

1.8 lentelė. Asociacijos tarp aktoriaus ir panaudos atvejo verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Aktorius gali vykdyti Panaudos atvejį;
2	Panaudos atvejis gali būti vykdomas Aktoriaus;

Panaudos atvejo praplėtimas

Žemiau pateiktoje lentelėje (žr. 1.9 lentelė) verbalizuojama grafiko dalis (žr. 1.10 pav.), kurioje naudojamas panaudos atvejo praplėtimas.



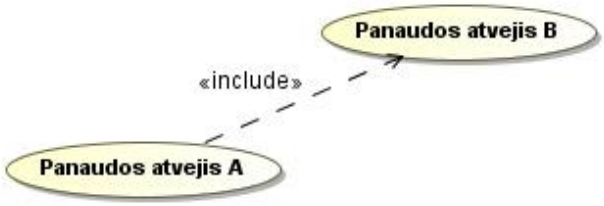
1.10 pav. Panaudos atvejo praplėtimas

1.9 lentelė. Panaudos atvejo praplėtimo verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Panaudos atvejas B praplečia Panaudos atvejį A;
2	Panaudos atvejas A praplečiamas Panaudos atvejo B;

Panaudos atvejo įtraukimas

Žemiau pateiktoje lentelėje (žr. 1.10 lentelė) verbalizuojama grafiko dalis (žr. 1.11 pav.), kurioje naudojamas panaudos atvejo įtraukimas.



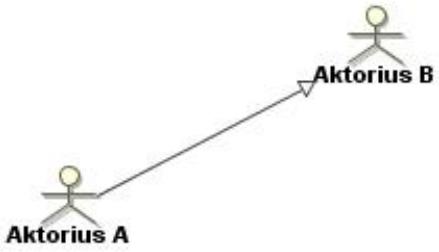
1.11 pav. Panaudos atvejo įtraukimas

1.10 lentelė. Panaudos atvejo įtraukimo verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Panaudos atvejis B apima Panaudos atveją A;
2	Panaudos atvejis A įtraukiamas Panaudos atvejo B;

Apibendrinimas – specializavimas tarp aktorių

Žemiau pateiktoje lentelėje (žr. 1.11 lentelė) verbalizuojama grafiko dalis (žr. 1.12 pav.), kurioje naudojamas apibendrinimas – specializavimas tarp aktorių.



1.12 pav. Apibendrinimas - specializavimas tarp aktorių

1.11 lentelė. Apibendrinimo - specializavimo tarp aktorių verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Aktorius A specializuoja Aktorių B;
2	Aktorius B apibendrinamas Aktoriaus A;

1.3.3. Veiklos diagramos verbalizavimas

Skyriuje aptariamos veiklos diagramos [6] performulavimo natūralia kalba ypatybės.

Veikla po proceso pradžios

Žemiau pateiktoje lentelėje (žr. 1.12 lentelė) verbalizuojama grafiko dalis (žr. 1.13 pav.), kurioje naudojamas veiklos vykdymas pradedant vykdyti veiklos procesus.

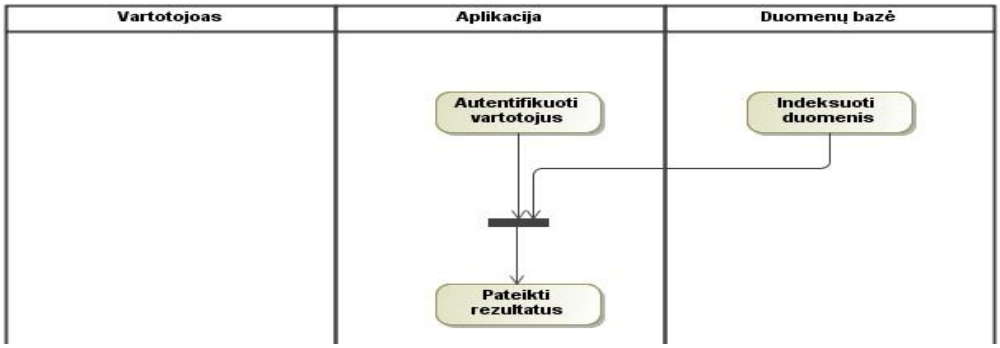
1.13 pav. Veikla po proceso pradžios

1.12 lentelė. Veiklos po proceso pradžios verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Procesas prasideda nuo veiklos „Kreiptis į autentifikaciją“;

Veiklų sujungimas

Žemiau pateiktoje lentelėje (žr. 1.13 lentelė) verbalizuojama grafiko dalis (žr. 1.14 pav.), kurioje naudojamas veiklų sujungimas.



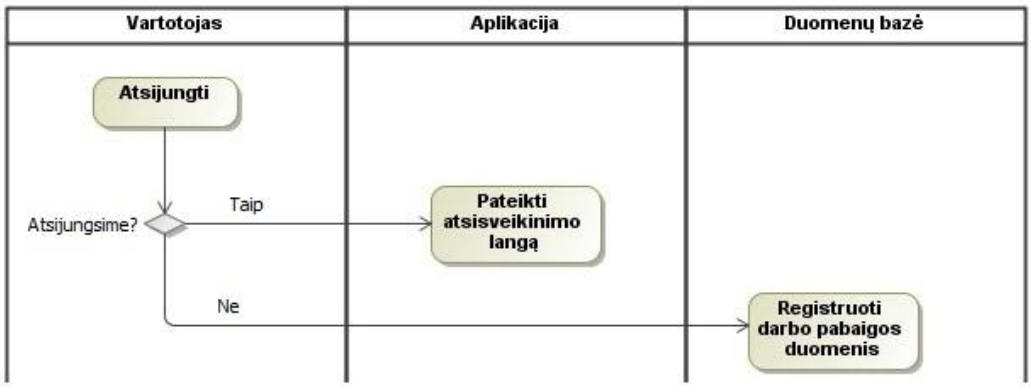
1.14 pav. Veiklų sujungimas

1.13 lentelė. Veiklų sujungimo verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Tik įvykdžius „Autentifikuoti vartotojus“ ir „Indeksuoti duomenis“ pradedama vykdyti „Pateikti rezultatus“;

Veiklos po sąlygos įvykdymo

Žemiau pateiktoje lentelėje (žr. 1.14 lentelė) verbalizuojama grafiko dalis (žr. 1.15 pav.1.14), kurioje naudojamas veiklų vykdymas po įvykdytos sąlygos.



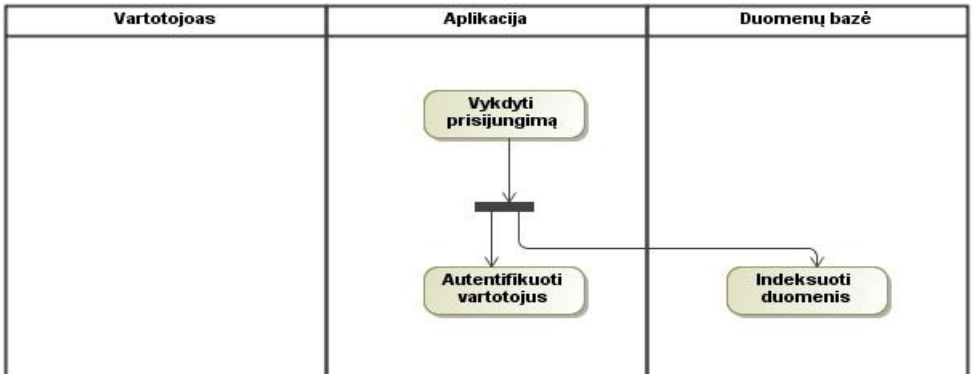
1.15 pav. Veiklos po sąlygos įvykdymo

1.14 lentelė. Veiklos po sąlygos įvykdymo verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Jei „Atsijungsime“, tai įvykdžius veiklą „Atsijungti“ vykdoma „Pateikti atsisveikinimo langą“;
2	Jei ne „Atsijungsime“, tai įvykdžius veiklą „Atsijungti“ vykdoma veikla „Registruoti darbo pabaigos duomenis“;

Veiklos po išsišakojimo

Žemiau pateiktoje lentelėje (žr. 1.15 lentelė) verbalizuojama grafiko dalis (žr. 1.16 pav.), kurioje naudojamas proceso iššakojimas į kelias veiklas.



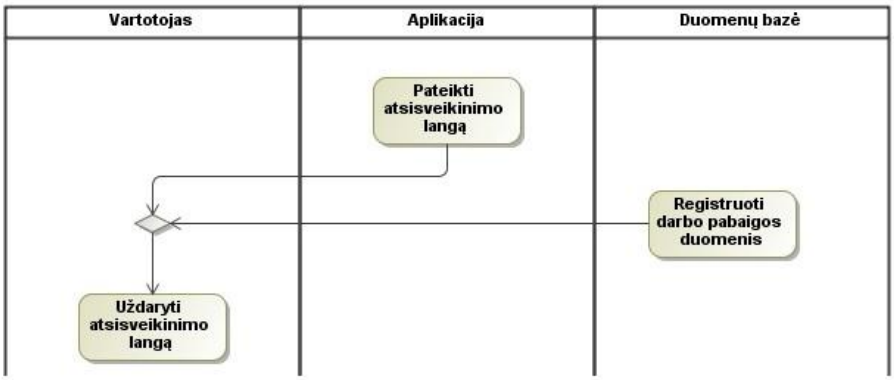
1.16 pav. Veiklos po išsišakojimo

1.15 lentelė. Veiklos po išsišakojimo verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Įvykdžius „Vykdyti prisijungimą“ paraleliai pradedama vykdyti „Autentifikuoti vartotojus“ ir „Indeksuoti duomenis“;

Veiklų suliejimas

Žemiau pateiktoje lentelėje (žr. 1.16 lentelė) verbalizuojama grafiko dalis (žr. 1.17 pav.), kurioje naudojamas proceso veiklų suliejimas į vieną.



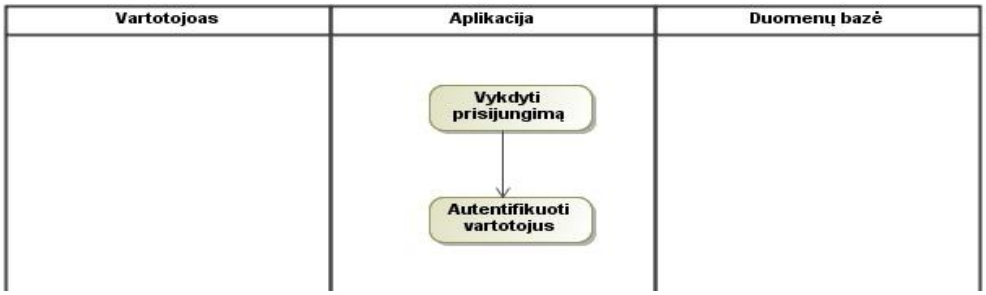
1.17 pav. Veiklos sujungimas

1.16 lentelė. Veikos sujungimo verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Įvykdžius „Pateikti atsisveikinimo langą“ arba „Registruoti darbo pabaigos duomenis“ vykdoma „Uždaryti atsisveikinimo langą“;

Veiklų vykdymo seka

Žemiau pateiktoje lentelėje (žr. 1.17 lentelė) verbalizuojama grafiko dalis (žr. 1.18 pav.), kurioje naudojamas proceso užbaigimas įvykdžius paskutinę veiklą.



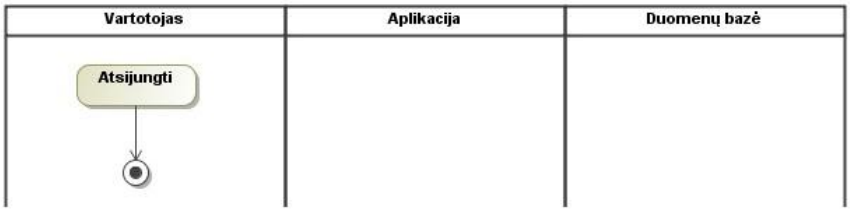
1.18 pav. Veiklų vykdymo seka

1.17 lentelė. Veiklų vykdymo sekos verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Įvykdžius „Vykdyti prisijungimą“ vykdoma „Autentifikuoti vartotojus“;

Užbaigimas po veiklos įvykdymo

Žemiau pateiktoje lentelėje (žr. 1.18 lentelė) verbalizuojama grafiko dalis (žr. 1.19 pav.), kurioje naudojamas proceso užbaigimas įvykdžius paskutinę.



1.19 pav. Užbaigimas po veiklos įvykdymo

1.18 lentelė. Užbaigimo po veiklos įvykdymo verbalizavimo rezultatas

Nr.	Verbalizavimas
1	Įvykdžius „Atsijungti“ procesas baigiasi;

1.4. Vartotojų analizė

Grafinio modelio išreikšto UML kalba išreiškimas į natūralią kalbą aktualus asmenims, kurie semiasi žinių UML kalbos modeliavime ir tiems, kurie aktyviai dalyvauja diskusijose su asmenimis (dalykinės srities žinovai, sistemų užsakovai) silpnai išmanančiais UML kalbos specifiką. Modelio išreikšto UML kalba išreiškimas į natūralią kalbą leistų geriau suprasti analizuojamo modelio išreikšto UML kalba esmę.

Galima būtų išskirti tokias dvi siekiamo sprendimo vartotojų grupes.

1. Analitikai, architektai – asmenys, kurie privalo perteikti modelių esmę išreikštą UML kalba asmenims, kurie silpnai išmano UML kalbos specifiką.
2. Studijuojantys UML kalbą – asmenys gilinantys žinias UML kalba

Vartotojų tikslas – modelio išreikšto UML kalba automatizuoto performulavimo panaudojimas grafinio modelio išreiškimui natūralia kalba. Tuo būdu suteikti galimybę modelius suprasti ir asmenims, kurie neturi žinių, reikalingų modelių skaitymui aprašytų UML kalba.

Pagrindiniai vartotojai yra analitikai, architektai kuriantys įvairių sričių veiklos sistemų grafinius modelius UML kalba. Patyrusių specialistų problema – įvertinti sudarytų modelių teisingumą (t.y. patikrinti, ar jie teisingai suprato gautą informaciją apie sritį), o tai padaryti gali tik padedami dalykinės srities žinovo, kuris dažniausiai neišmano UML kalbos, todėl tenka išreikšti modelius sudarytus UML kalba išreikšti natūralia kalba.

Lentelėje (žr. 1.19 lentelė) pateikiami vartotojų problemų aprašai.

1.19 lentelė. Vartotojų problemų aprašai

Nr.	Problema	Problemos sprendimas be modelio verbalizavimo
1	UML modelio tikslinimas	Modelių išreikštų UML kalba sudarytojai diskutuoja ir analizuoja, ar jų sudarytas modelis tinkamai perteikia įdėją.
2	UML modelio pateikimas	Ypatingai sistemų analitikai, kurie atsakingi už sistemų kūrimo dokumentacijos pildymą, stengiasi apteikti kuo daugiau informacijos natūralia kalba apie aprašytus modelius UML kalba dokumentacijos skaitytojams, kurie neturi gebėjimų skaityti modelių aprašytų UML kalba.
3	UML modelio supratimas	Studentai ar kiti asmenys, kurie gilina žinias modelių sudaryme UML kalba, konsultuojasi su dėstytojais ar asmenimis, suprantančiais modelių sudarymo UML kalba subtilybes. Tik tokiu būdu, jie gali įsitikinti, ar jų aprašytas modelis UML kalba yra pakankamai logiškas.

1.5. Problemos sprendimo metodų literatūros šaltiniuose analizė

Viena iš alternatyvų, kaip galėtų būti verbalizuojamas grafinis modelis į natūralią kalbą yra Mustafa Jarrar [2] siūlomas „ORM Markup Language“. Straipsnyje „Towards Methological Principles for Ontology Engineering“ [1] aprašomas būdas panaudoti logikos teorijų, aksiomatizacijų ir kitų specifikacijų, pvz. verslo taisyklių, verbalizaciją. Sprendimas pritaikytas verbalizuoti ORM [2] pavaizduotus koncepcinius modelius, tačiau norint pagelbėti srities ekspertams, pagrindinius principus galima taikyti ir kitiems modeliams ar formalioms kalboms, pvz. aprašomajai logikai. Taip pat sprendimas pasižymi lankstumu, plėtros galimybėmis, paprastai tvarkomais verbalizacijos šablonais. Sprendimas leidžia verbalizuoti modelius įvairiomis šnekamosiomis kalbomis kiekvienai kalbai sukuriant atskirą verbalizavimo šabloną.

Modelių verbalizavimui sukurta ORM-ML (ORM – Markup Language) metodologija. Ši metodologija paremta XML sintakse ir formuluojama XML – schema.

ORM dokumentas yra pateikiamas kaip viena žymė ir vadinama ORMSchema, kuri savyje turi dar du mazgus (angl. „node“): ORMMeta ir ORMBody. Žemiau pateikiamas ORMSchema dokumento (elemento) pavyzdys (žr. 1.20 pav.).

```
<?xml version="1.0" encoding="UTF-8" ?>
 <ORMSchema xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://starlab.vub.ac.be/ORMML/ormml.xsd"
xmlns:dc="http://purl.org/dc/elements/1.1/">
  <ORMMeta>
     ...
  </ORMMeta>
  <ORMBody>
     ...
  </ORMBody>
</ORMSchema>
```

1.20 pav. ORM schema

ORMMeta

„ORMMeta metadata“ yra tarsi ORM-ML dokumento antraštė, kuri savyje saugo informaciją apie ORM dokumentą. Pvz., „Pavadinimas“, „Kūrėjas“, „Versija“ ir kt. Kiekvienas meta elementas turi du elementus: pavadinimas ir turinys. Pagrindinė šios elementarios struktūros idėja yra suteikti lankstumo egzistuojančių metaduomenų standartų priėmimui. Žemiau pateikiamas ORMMeta žymės pavyzdys (žr. 1.21 pav.).

```
....
<ORMMeta>
  <Meta name="DC.Title" content="Complaint Problems"/>
  <Meta name="DC.Creator"  content="Mustafa Jarrar"/>
  <Meta name="DC.Language"  content="English"/>
  <Meta name="DC.Description" content="An axiomatization of
complaint problems categories..."/>
  ...
</ORMMeta>
 ....
```

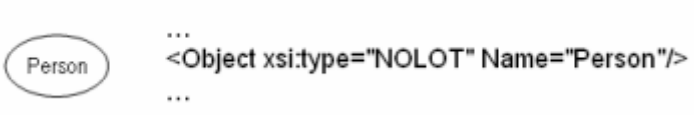
1.21 pav. ORM meta

ORMBody

„ORMBody“ žymė susideda iš penkių skirtingų elementų: objekto tipo, elemento potipio, predikato, predikato objekto ir apribojimo (angl. „constraint“).

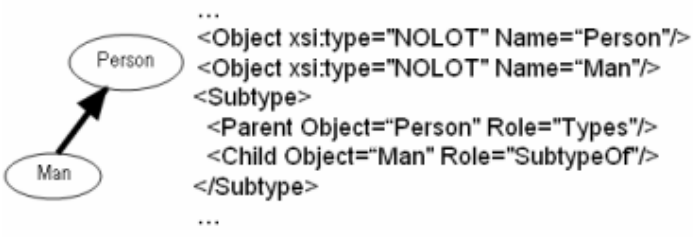
Aptarkime kiekvieną aukščiau minėtų elementų:

1. Objekto tipas – tai yra elementas, kuris turi atributus „xsi:type“ ir „Name“. Pateikiamas pavyzdys, kaip atrodo šis elementas ORM struktūroje (žr. 1.22 pav.)



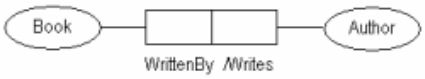
1.22 pav. ORM objektas

2. Potypis – potipio žymės yra naudojamos ryšių tarp objektų tipų pateikimui. Potipio žymė privalo turėti du elementus: „tėvą“ ir „vaiką“. Pateikiamas pavyzdys, kaip atrodo šis elementas ORM struktūroje (žr. 1.23 pav.)



1.23 pav. ORM potypis

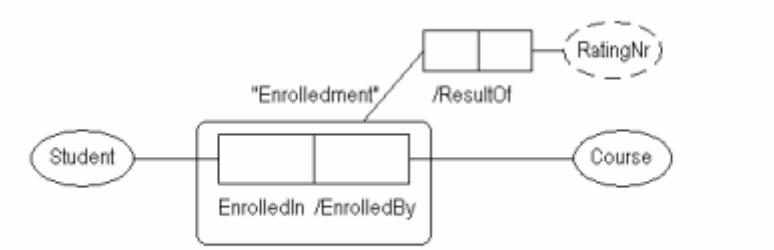
3. Predikatas – predikatą sudaro bent vienas „Object_Role“ elementas. Šis elementas yra nuoroda tam tikrą objektą ir gali turėti tam tikrą vaidmenį. ORM schemoje šie elementai atstovauja stačiakampius. Pateikiamas pavyzdys, kaip atrodo šis elementas ORM struktūroje (žr. 1.24 pav.)




```
...
<Object xsi:type="NOLOT" Name="Book"/>
<Object xsi:type="NOLOT" Name="Author"/>
<Predicate>
   <Object_Role ID="ID1" Object="Book"
       Role="WrittenBy"/>
   <Object_Role ID="ID2" Object="Author"
       Role="Writes"/>
</Predicate>
...
```

1.24 pav. ORM predikatas

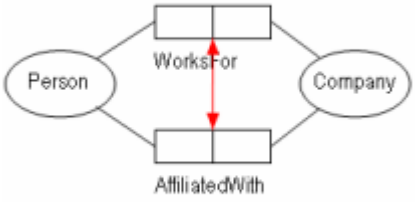
4. Predikato objektai – šie elementai saugo predikatų elementus ir turi atributą „Predicate_Name“. Pateikiamas pavyzdys, kaip atrodo šis elementas ORM struktūroje (žr. 1.25 pav.)




```
...
<Object xsi:type="NOLOT" Name="Student"/>
<Object xsi:type="NOLOT" Name="Course"/>
<Object xsi:type="LOT" Name="RatingNr"/>
<Predicate_Object Predicate_Name="Enrollment">
  <Predicate>
    <Object_Role ID="ID18" Object="Student" Role="EnrolledIn"/>
    <Object_Role ID="ID19" Object="Course" Role="EnrolledBy"/>
  </Predicate>
</Predicate_Object>
<Predicate>
  <Object_Role ID="ID20" Object="Enrollment" Role=""/>
  <Object_Role ID="ID21" Object="RatingNr" Role="ResultOf"/>
</Predicate>
...
```

1.25 pav. ORM predikatas struktūroje

5. Apribojimas – tai elementas vykdantis įvairių tipų apribojimus, pvz., privalomumas, unikalumas, lygybė ir kt. Žemiau pateikiamas lygybės apribojimo pavyzdys (žr. 1.26 pav.)




```
...
<Constraint xsi:type="Equality ">
  <First>
    <Object_Role>ID1</Object_Role>
    <Object_Role>ID2</Object_Role>
  </First>
  <Second>
    <Object_Role>ID3</Object_Role>
    <Object_Role>ID4</Object_Role>
  </Second>
</Constraint>
...
```

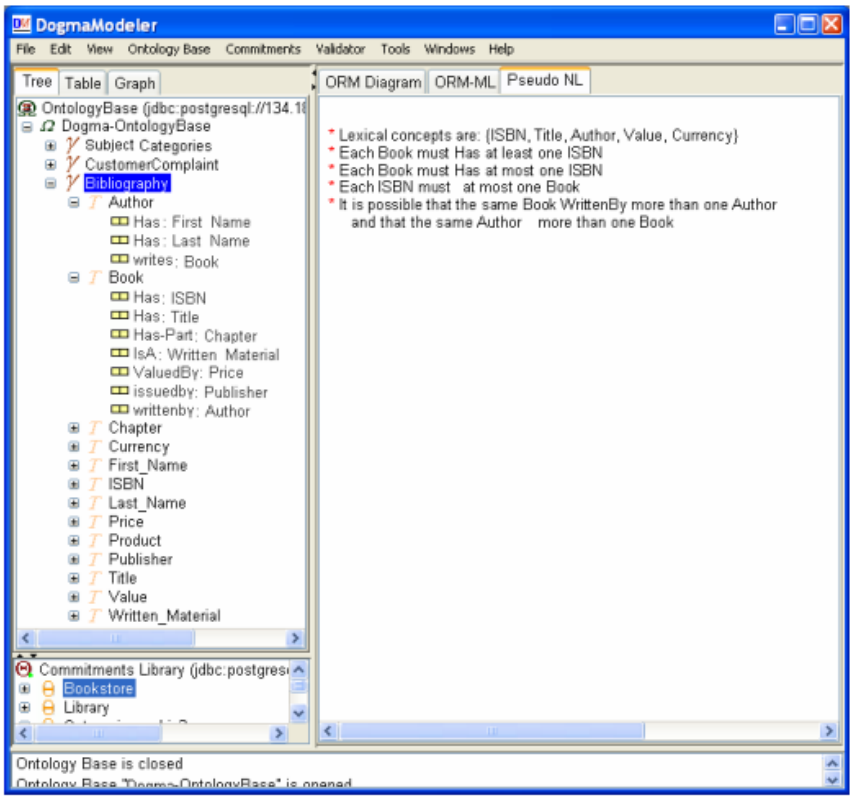
1.26 pav. Lygybės apribojimas

1.6. Verbalizavimo sprendimų analizė

Šiame skyriuje pateiksiu ORM-ML metodologija paremto realizuoto įrankio „DogmaModeler Ontology Engineering Tool“ [3] aprašymą.

Diagrama (žr.) demonstruoja ORM diagramos verbalizavimą pseudo natūralia kalba (fiksuotos sintaksės anglų kalbos sakiniai), kurie yra automatiškai generuojami „DogmaModeler“ įrankio pagalba. Įrankis vykdo verbalizavimą remdamasis iš anksto aprašytą verbalizavimo šabloną ORM-ML dokumente.

Pateikiamas „DogmaModeler“ įrankio ekrano vaizdas (žr. 1.27 pav.).



1.27 pav. Dogma Modeler ekrano vaizdas

Pateikiamas pavyzdinis verbalizacijos šablonas su būtinuoju apribojimu. Įrankiui pateikus šį šabloną (žr. 1.28 pav.), verbalizavimo rezultatas yra:

“Each Bookmust Hasat least one ISBN“

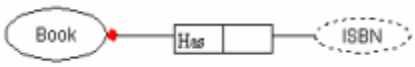
Pateikiamas vertimas lietuvių kalba:

„Kiekviena knyga privalo turėti bent vieną ISBN“

```
<Constraint xsi:type="Mandatory">
   <Text> Each</Text>
   <Object index="0"/>
   <Text> must</Text>
   <Role index="0"/>
   <Text> at least one</Text>
   <Object index="1"/>
</Constraint>
```

1.28 pav. ORM šablonas

Grafinis ORM-ML dokumento fragmentas pateiktas diagramoje žemiau (žr. 1.29 pav.)



1.29 pav. ORM-ML dokumento fragmentas

1.7. Architektūros ir galimų įgyvendinimo priemonių variantų analizė

Grafinio modelio verbalizacijos į natūralią kalbą architektūros projektavimui ir realizacijai technologijos įrankių pasirinkimas itin platus.

Architektūros projektavimui, tinkami „Magic Draw“, „IBM Rational Sotware Architect“ ar net „Microsoft Visio“ įrankiai. Remiantis UML tipo diagramomis, kurias palaiko jau minėti įrankiai, galimas detalus architektūros suplanavimas nuo pat kuriamos sistemos griaučių (sistemos komponentų diagrama) iki pat smulkiausių, apibrėžiančių detalias sekos diagramas.

Visos nagrinėtos architektūros projektavimui naudotinos projektavimo sistemos yra komercinės. Taip pat, jos nėra atviro kodo sistemos.

Žemiau pateiktoje lentelėje lyginami minėti įrankiai UML diagramų realizavimui (žr. 1.20 lentelė)

1.20 lentelė. Įrankiai, diagramų išreikštų UML kalba realizavimui

Nr.	Įrankio pavadinimas	Ypatybės
1	Magic Draw UML	Kūrėjas: No Magic
		Platforma/OS: Įvairios sistemos
		Pasirodė (metai): 1998
		Programavimo kalba: Java
2	IBM Rational Sotware Architect	Kūrėjas: IBM
		Platforma/OS: Windows, Linux
		Pasirodė (metai): 1990
		Programavimo kalba: Java/C++
3	Microsoft Visio	Kūrėjas: Microsoft
		Platforma/OS: Windows
		Pasirodė: 1992
		Programavimo kalba: Nežinoma

Realizacijai renkamasi iš programavimo kalbų, su kuriomis įgyti tvirčiausi įgūdžiai. Rinktasi iš populiariausių programavimo kalbų – C# ir Java.

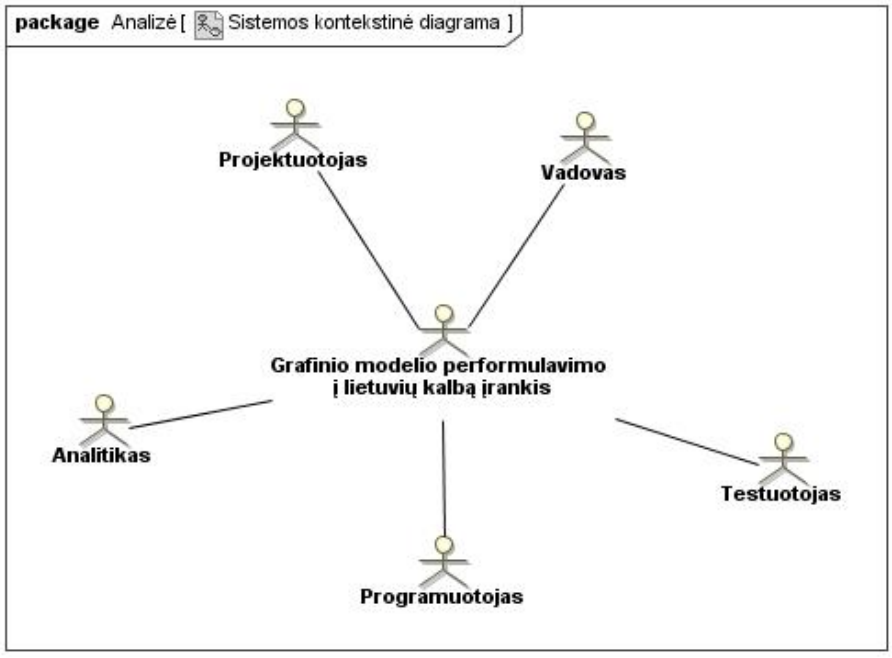
Java – atvira, nemokama programavimo kalba, kuri veikia populiariausiose pasaulyje operacinėse sistemose. Populiariausi įrankiai – Eclipse / NetBeans.

C# - programavimo kalba, kuri veikia tik „Windows” operacinėje sistemoje. Naudojamamas populiariausias įrankis programavimui su C# kalba – “Visual Studio”.

1.8. Siekiamos sistemos apibrėžimas

Siekiamos sistemos naudotojai (vartotojai) apibrėžiami žemiau pateiktoje sistemos kontekstinėje diagramoje (žr. 1.30 pav.). Naudotojui poaibis koncentruotas į IT įmonės veiklą (pareigybės, atsakomybės). Sistemos kontekstinėje diagramoje būtent minėtų IT įmonės darbuotojų perspektyvoje pateikiamas jų sąveika su sistema. Pateikiami naudotojų aprašai:

- Vadovas – asmuo, tvirtinantis priduodamos sistemos dokumentus, norintis turėti galimybę peržvelgti suprojektuotos sistemos grafinių modelių teisingumą;
- Projektuotojas – asmuo, atliekantis projektavimo darbus ir siekiantis patikslinti savo projektuotų diagramų teisingumą;
- Analitikas – asmuo, perteikiantis projektuotojo pateiktus grafinius modelius užsakovui suprantama kalba. Pildantis dokumentaciją, kurioje talpina tiek grafinius modelius, tiek jų aprašus;
- Programuotojas – asmuo, realizuojantis sistemą pagal pateiktus grafinius modelius bei jų aprašus;
- Testuotojas – asmuo, vykdantis realizuotos sistemos funkcionalumo testavimą pagal pateiktą sistemos aprašą.



1.30 pav. Sistemos kontekstinė diagrama

1.9. Darbo tikslas ir siekiami privalumai

Tiriamojo darbo tikslas – modelio išreikšto UML kalba išreiškimo natūralia kalba automatizavimu padėti modelių UML kalba kūrėjams ir dokumentuotojams bei UML kalbą studijuojantiems sudaryti sąlygas efektyviau tikrinti sukurtų ar kuriamų modelių teisingumą, detaliau aprašinėti sukurtus modelius sistemų dokumentacijose.

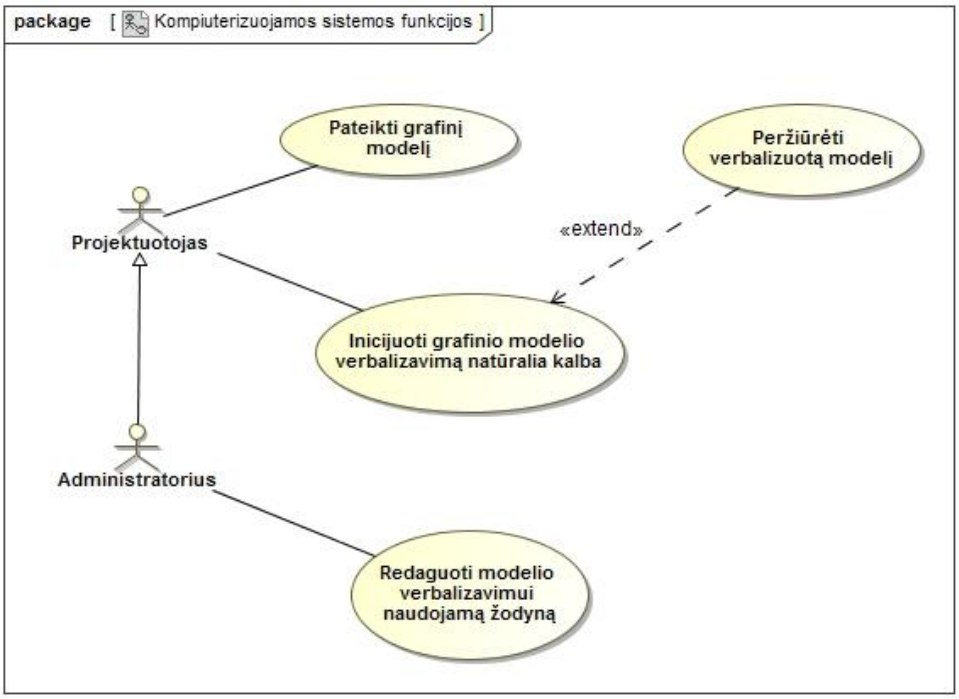
Tariamojo darbo siekiamas sprendimas – sukurti grafinio modelio lietuviškų sakinių formulavimo taisykles (žodyną), pagal kurį realizuotas įrankis generuos korektiškai suformuluotus lietuviškus sakinius, apibrėžiančius objektų sąryšius tarp pateikto grafinio modelio.

Darbo tikslui pasiekti turi būti išspręsti šie pagrindiniai uždaviniai:

5. Išanalizuoti pasirinktų grafinių modelių išreikštų UML kalba sąryšių tarp objektų panaudojimą sakinių išreiškimui natūralia kalba;
6. Sukurti pateiktų sąryšių verbalizavimo taisykles;
7. Ištirti lietuvių kalbos linksniavimo panaudojimo galimybes grafinio modelio išreikšto UML kalba išreiškimui natūralia kalba;
8. Sukurti verbalizavimo programą ir eksperimentiškai ją įvertinti.

1.10. Kompiuterizuojamos sistemos funkcijos

Grafinio modelio išreikšto UML kalba performulavimo natūralia kalba kompiuterizuojamos funkcijos (žr. 1.31 pav.) turi leisti vartotojui (priklausomai nuo to, koks jo tikslas) gauti suprojektuoto modelio aprašą natūralia kalba.



1.31 pav. Kompiuterizuojamos sistemos funkcijos

Kuriamos sistemos funkcijomis besinaudojantys vartotojai išskiriami į du tipus: naudotojas ir administratorius.

Tik sistemos administratorius turi turėti galimybę redaguoti ir pildyti grafinio modelio verbalizavimui naudojamą žodyną, kuriame saugomos numatytosios grafinio modelio asociacijų reikšmės (pavadinimai). Taip pat, administratorius turės ir eiliniam naudotojui suteikiamas teises, todėl jei sistemos administratoriaus teisės yra suteikiamas projektuotojui, vienas asmuo (naudotojas) turės visas galimybes tiek keisti sistemos žodyną, tiek dirbti su realizuotomis funkcijomis.

Kompiuterizuojamų funkcijų aprašas pateiktas lentelėje (žr. 1.21 lentelė).

1.21 lentelė. Kompiuterizuojamų funkcijų aprašas

Nr.	Kompiuterizuojamos funkcijos pav.	Aprašymas
1	Pateikti grafinį modelį	Grafinio modelio išreikšto UML kalba ir aprašyto naudojant „MagicDraw“ įrankį pateikimas
2	Inicijuoti grafinio modelio verbalizavimą natūralia kalba	Pateikto modelio išreikšto UML kalba išreiškimo natūralia kalba automatizavimo vykdymas
3	Peržiūrėti verbalizuotą modelį	Pateikto modelio išreikšto natūralia kalba rezultatų peržiūra
4	Redaguoti modelio verbalizavimui naudojamą žodyną	Modelio verbalizavimui naudojamo žodyno redagavimas

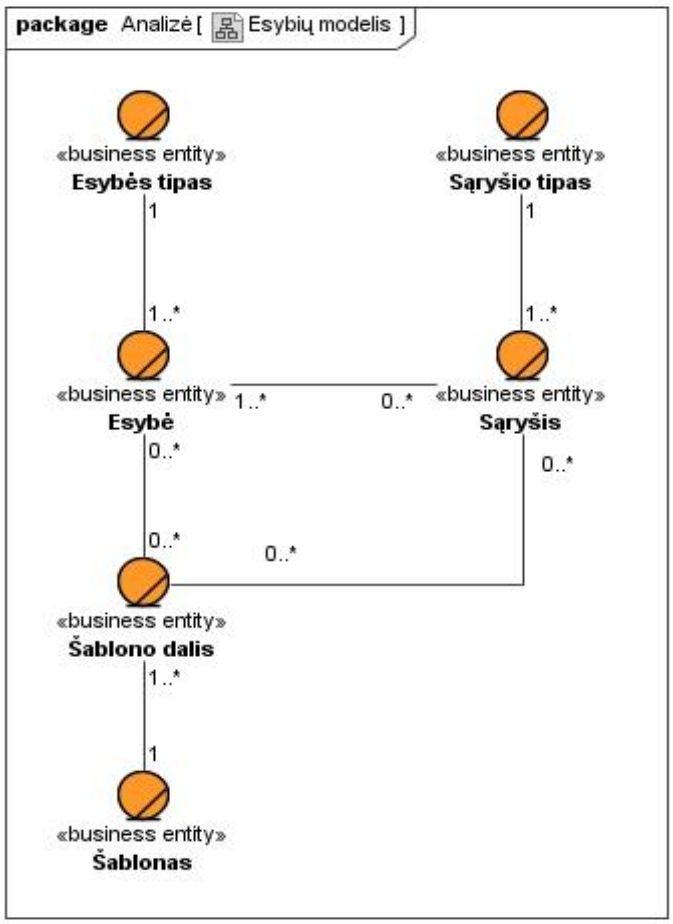
Sistemos naudotojas tikėdamasis gauti verbalizuotą į natūralią kalbą grafinį modelį, visų pirma privalės grafinio modelio priskyrimo funkcijos pagalba nurodyti norimą grafinį modelį, kuris bus verbalizuojamas. Šis funkcionalumas pateiktoje panaudojimo atvejų diagramoje vadinamas „Pateikti paruoštą grafinį modelį“.

Įvykdžius pirmąjį žingsnį (pateikus grafinį modelį) sistemos naudotojas gali inicijuoti sekantį funkcionalumą – grafinio modelio verbalizavimas. Pateiktoje diagramoje šis funkcionalumas vadinamas „Inicijuoti grafinio modelio performulavimą“.

Sistemai sėkmingai įvykdžius grafinio modelio verbalizavimą, naudotojui pateikiamas (pagal pasirinktą būdą, programos lange arba išsaugoma fiziniame faile) rezultatas.

1.11. Reikalavimai duomenims

Pateiktame esybių modelyje (žr. 1.32 pav.) detalizuojamas pateikto grafinio modelio objektų ir sąryšių susietumas su verbalizuojamo šablono dalimis. Grafinio modelio dalys išskaidytos privalės būti perteiktos kaip verbalizavimo rezultatas, todėl kiekvienas itin svarbu, siekiant korektiško rezultato, visus grafiniuose modeliuose pateiktus objektus apjungti logiškais sąryšiais. Priešingu atveju, verbalizavimas į natūralią kalbą netenka prasmės.



1.32 pav. Esybių modelis

Pateikto grafinio modelio duomenys ir rezultatas nukreipiamas į šias esybes:

1. Grafinio modelio elemento tipas – kiekvienas pateiktame grafiniame modelyje egzistuojantis objektas turi savo tipą. Pagal UML standartus, verbalizuodami „Magic Draw“ įrankiu sukurtus grafinius modelius numatyti šie tipai: klasė (esybė), aktorius, panaudos atvejas;
2. Grafinio modelio elementas – konkretus objektas, kuris egzistuoja pateiktame grafiniame modelyje, priskirtas tam tikram tipui bei turintis savo savybes (pavadinimą, unikalų identifikatorių)

3. Sąryšio tipas – kiekvienas pateiktame grafiniame modelyje egzistuojantis sąryšis turi savo tipą. Pagal UML standartus, verbalizuodami „Magic Draw“ įrankiu sukurtus grafinius modelius numatyti šie tipai: asociacija, generalizacija, kompozicija, agregavimas;
4. Sąryšis – konkretus pateiktame grafiniame modelyje egzistuojantis ryšys tarp dviejų objektų, priskirtas tam tikram tipui ir turintis savo savybes (bendra pavadinimą, ryšio galų pavadinimus, identifikatorių);
5. Šablono dalis – naudojama šablono dalis, kuri apibrėžia sąryšio ar objekto įterpimą į verbalizavimui naudojamą šabloną;
6. Šablonas – konkretaus sąryšio tipo šablonas, kuris subendrins sąryšio siejamus objektus į natūralios kalbos sakinį.

1.12. Nefunkciniai reikalavimai ir apribojimai

Projektuojamai sistemai keliami nefunkciniai reikalavimai:

1. Sistemos grafinė sąsaja turi būti:
    a. Lietuviška;
    b. Valdoma naudojant pelę;
2. Įrangai:
    a. Veikimas Windows XP+ operacinėje sistemoje;
    b. Suderinamumas su XML formato failais;
    c. Microsoft .NET 4.0+ versijos palaikymas.
3. Patikimumui:
    a. Sistema negali sutrikti (sustoti veikus), kuomet pateiktas; grafinis modelis yra nekorektiškas;
    b. Sistema turi užtikrinti, jog pateiktas grafinis modelis įvykdžius verbalizavimą nebus modifikuotas.
4. Rezultatui:
    a. Kalbos natūralumas – grafinio modelio elementai turi būti linksniuojamas vardan sakinių natūralumo lietuvių kalboje;
    b. Pateikimo aiškumas – rezultatai turi būti pateikti naudotojui patogia forma. Kiekvieno diagramos sąryšio išreiškimas turi būti grupuojamas su kitais sąryšio išreikštais natūralios kalbos sakiniais.

1.13. Rizikos faktorių analizė

Rizikos faktoriai taikytini siekiamo realizuoti įrankio veikimui įtakoja tiek sistemos naudotojo veiksmus, tiek komunikacijas, kurios būtinos kokybiškam įrankio veikimui. Žemiau pateikiama, išvardijami rizikos faktoriai, kurie yra kertiniai įrankio veikimui, vykdant grafinio modelio verbalizavimą:

- **Prieiga prie interneto**. realizuotas įrankis gebės formuluoti sakinius su korektiškai suformuotomis veiksmažodžių galūnėmis. Veiksmažodžio linksniavimui bus naudojamas linksniavimo servisas, kuris bus nutolęs, t.y. jis bus pasiekiamas vykdant transakcijas internetu.
- **Grafinio modelio korektiškumas.** Įrankiui pateikiamas grafinis modelis privalo būti tinkamai suformuotas. Jo XML struktūra turi atitikti visus „Magic Draw“ įrankio numatytus standartus formuojant grafinius modelius. Grafinio modelio verbalizavimo įrankis, aptikęs nesutapimą pateiktame grafiniame modelyje (jo XML struktūroje), įrankio naudotojui pateiks pranešimą, jog nėra gebama verbalizuoti pateikto grafinio modelio, dėl jo nekorektiškos struktūros ir naudotojas privalės pateikti kitą, tinkamai suformuotą grafinio modelį.
- **Numatytojo šablono žodyno neegzistavimas.** Jei pateikto grafinio modelio verbalizuojamose diagramose bus pateikta sąryšių tarp objektų be sąryšio galuose aprašytų veiksnių ir sąryšis neturės bendrinio pavadinimo, jo verbalizacijai į natūralią kalbą privaloma turėti žodyne aprašytą veiksmažodinę formą aktyvaus sąryšio verbalizavimui. Žodyne neegzistuojant minėtam sąryšio aprašymui, negalimas jo verbalizavimas. Tuo atveju, vartotojui bus pateiktas pranešimas, kuris informuos apie būtinybę papildyti šablono žodyną.

1.14. Rezultato kokybės kriterijai

Grafinio modelio išreikšto UML kalba išreiškimo į natūralią kalbą automatizavimo rezultatas turi būti tikslus ir išbaigtas, kadangi menkiausias skirtumas tarp rezultato ir pateikto grafinio modelio logikos iškreipia grafinio modelio projektuotojo perteikiamą idėją, mintis. Žemiau pateikiami esminiai kriterijai į kuriuos reikia atsižvelgti siekiant maksimalios rezultato kokybės:

- **Išbaigtumas**. Būtina išrinkti ir išreikšti į natūralią kalbą visus egzistuojančius sąryšius iš pasirinktos, verbalizuojamos diagramos. Naudotojo pateikto grafinio modelio išreikšto UML kalba išreiškimas į natūralią kalbą neapimant visų sąryšiu gali visiškai iškreipti projektuotojo, kūrusio pateiktą grafinį modelį, idėją.
- **Kalbos natūralumas**. Grafinio modelio elementai turi būti linksniuojami vardan sakinių natūralumo lietuvių kalboje.
- **Pateikimo aiškumas**. Rezultatai turi būti pateikti naudotojui patogia forma. Kiekvieno diagramos sąryšio išreiškimas turi būti grupuojamas su kitais sąryšio išreikštais natūralios kalbos sakiniais.

1.15. Analizės išvados

1. Grafinio modelio išreikšto UML kalba performulavimas natūralia kalba leidžia UML modelių sudarytojams ir analizuotojams detaliau suprasti ne tik modelius aprašančius statiką, bet ir modelius aprašančius dinamiką, taip prisidendant prie pradinių modelių kokybės gerinimo;
2. Išanalizavus egzistuojančias metodikas ir įrankius, kurie geba performuluoti tik grafinius modelius aprašančius statiką esybių – ryšių požiūriu, bet neaptikta gebančių performuluoti modelius išreikštus UML kalba aprašančius statiką objektiniu požiūriu į natūralią kalbą;
3. Realizuojamų veiklos modelių patvirtinimo iš dalykinių sričių žinovų būtinybė ir atlikus analizę neaptikus analogiškų sprendimų priimta, jog sistemos kūrimas yra reikalingas.

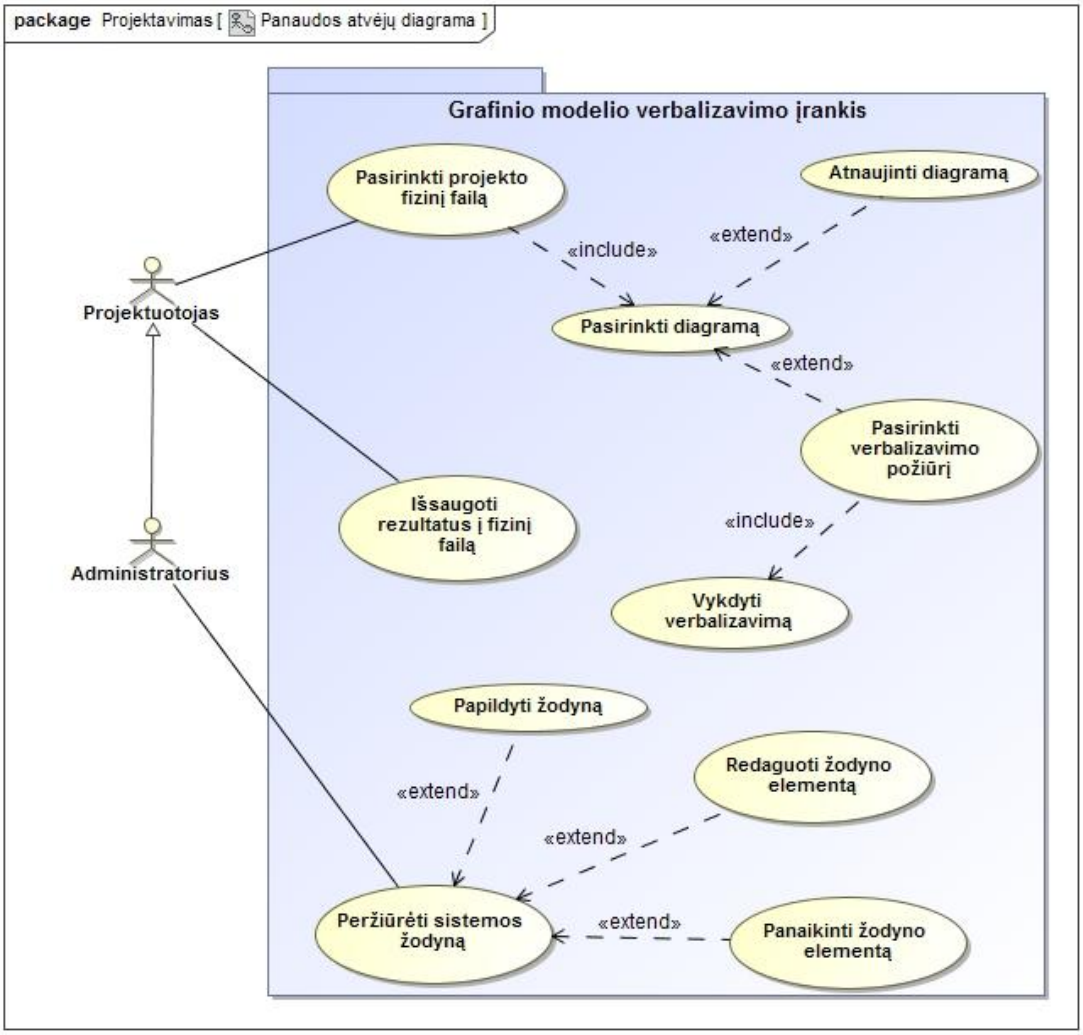
2. Grafinio modelio performulavimo įrankio reikalavimų specifikacija ir analizė

1.16. Reikalavimų specifikacija

Skyriuje pateikiami kuriamai sistemai keliami funkciniai reikalavimai, kurie apjungia panaudojimo atvejus bei dalykinės srities modelį.

1.16.1. Įrankio panaudojimo atvejų diagrama

Žemiau pateikiama įrankio panaudojimo atvejų diagrama (žr. 2.1 pav.). Sistemos administratorius (pagr. funkcija – redaguoti sistemos žodyną) turi teises vykdyti visas sistemos funkcijas, kurias turi ir projektuotojas.



2.1 pav. Panaudojimo atvejų diagrama

1.16.2. Įrankio panaudojimo atvejų specifikacijos

Pateikiamas įrankio panaudojimo atvejų specifikacijos (žr. 2.1 lentelė ir 2.2 lentelė).

2.1 lentelė. Grafinio modelio pateikimo specifikacija

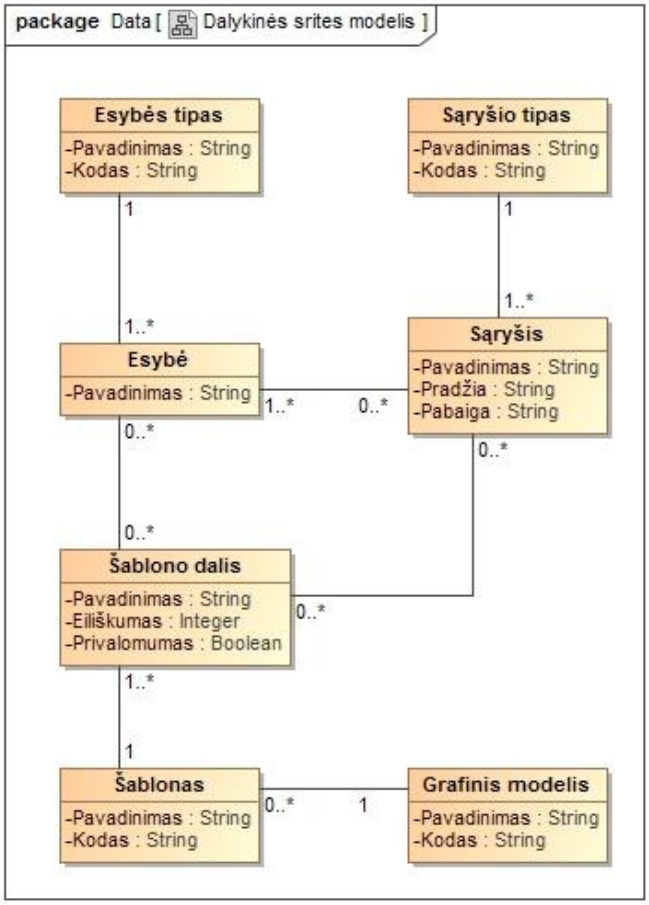
<table>
<tr><th colspan="3">PA „Pateikti grafinį modelį“</th></tr>
<tr><td colspan="3">Tikslas: verbalizavimo įrankiui pateikti grafinį modelį, kuris bus verbalizuojamas</td></tr>
<tr><td colspan="2">Prieš sąlyga</td><td>Naudotojas prisijungęs prie sistemos</td></tr>
<tr><td colspan="2">Aktorius</td><td>Registruotas sistemos naudotojas</td></tr>
<tr><td colspan="2">Sužadinimo sąlyga</td><td>Grafinio modelio pateikimo funkcijos iniciavimas</td></tr>
<tr><td rowspan="3">Susiję panaudojimo atvejai</td><td>Išplečia PA</td><td>-</td></tr>
<tr><td>Apima PA</td><td>-</td></tr>
<tr><td>Specializuoja PA</td><td>-</td></tr>
<tr><td colspan="2">Pagrindinis įvykių srautas</td><td></td></tr>
<tr><td colspan="2">1. Paspaudžiamas mygtukas „Pateikti grafinį modelį“</td><td>Pateikiamas grafinio modelio pasirinkimo dialogo langas</td></tr>
<tr><td colspan="2">2. Dialogo lange pasirenkamas grafinis modelis</td><td>Fiksuojamas pasirinkto grafinio modelio adresas saugykloje (naudotojo kompiuteryje)</td></tr>
<tr><td colspan="2">Po sąlyga:</td><td>Įrankyje užkraunamas pasirinktas grafinis modelis</td></tr>
<tr><td colspan="3">Alternatyvūs scenarijai</td></tr>
<tr><td colspan="2">1a. Atšaukiamas grafinio modelio pasirinkimas</td><td>Grįžtama į pagrindinį įrankio langą</td></tr>
</table>

2.2 lentelė. Grafinio modelio vykdymo verbalizavimo specifikacija

PA „Vykdyti grafinio modelio verbalizavimą“		
Tikslas: verbalizuoti pateiktą grafinį modelį pagal sistemoje registruotą verbalizavimo šabloną		
Prieš sąlyga		Naudotojas prisijungęs prie sistemos
Aktorius		Registruotas sistemos naudotojas
Sužadinimo sąlyga		Grafinio modelio performulavimo funkcijos iniciavimas
Susiję panaudojimo atvejai	**Išplečia PA**	-
	Apima PA	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		
1. Paspaudžiamas mygtukas „Verbalizuoti“		Vykdomas grafinio modelio performulavimas
2. Spaudžiamas mygtukas „Peržiūrėti rezultatus“		Rezultatų lange pateikiamas performuluotas grafinis modelis
Po sąlyga:		Įrankyje arba fiziniame rezultatų faile pateikiamas performuluotas grafinis modelis

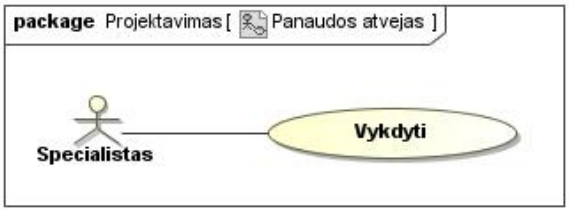
1.17. Dalykinės srities modelis

Žemiau pateikiamas dalykinės srities modelis su išrašytais esybių atributais (žr. 2.2 pav.). Išskirti grafinio modelio išskaidymo sritis bei sistemos naudotojo bei grafinio modelio registravimas.



2.2 pav. Dalykinės srities modelis

Remiantis panaudojimo atvejų diagrama (žr. 2.3 pav.), grafinis modelis būtų išskaidytas šitaip:



2.3 pav. Panaudos atvejo diagramos pavyzdys

Esybės tipas (Pavadinimas): „Aktorius“, „PanaudosAtvejas“

Esybė (Pavadinimas): „Specialistas“, „Vykdyti“

Sąryšio tipas (Pavadinimas): „Asociacija“

Sąryšis (Pavadinimas): „Gali vykdyti“, „Gali būti vykdomas“, „Siejasi“

Žemiau pateikiami visų dalykinės srities modelių (žr. 2.2 pav.) esybių aprašai. Kiekvienas jų saugo atitinkamą informaciją iš išskaidyto pateikto grafinio modelio ir sistemoje nurodytų (išsaugotų) verbalizavimo šablonų.

2.3 lentelė. Dalykinių srities modelių aprašas

Nr.	Esybės pavadinimas	Aprašymas
1	Esybės tipas	Saugomas išskaidytas esybės tipas (pvz., klasė)
2	Esybė	Saugoma išskaidytos esybės informacija
3	Sąryšio tipas	Saugomas išskaidytas sąryšio tipas (asociacija, generalizacija, aggregacija, kt.)
4	Sąryšis	Saugoma išskaidyto sąryšio informacija (pavadinimas, sąryšio pradžios veiksnys, sąryšio pabaigos veiksnys)
5	Šablono dalis	Nurodoma numatoma šablono dalis, kuriai bus taikomas išskaidytas elementas
6	Šablonas	Verbalizavimo šablonas, kuris taikytinas išskaidytai grafinio modelio daliai
7	Grafinis modelis	Pateikto grafinio modelio informacija

3. Automatizavimo įrankio projektas

1.18. Įrankio pagrindimas ir esmės išdėstymas

Skyriuje pateikiami specifikuotus reikalavimus tenkinančio įrankio sukurimui reikalingi architektūriniai sprendimai.

Pateikiamas grafinio modelio išreikšto UML kalba išreiškimo iš „MagicDraw“ įrankiu aprašyto projekto natūralia kalba abstraktus darbo sekos aprašas:

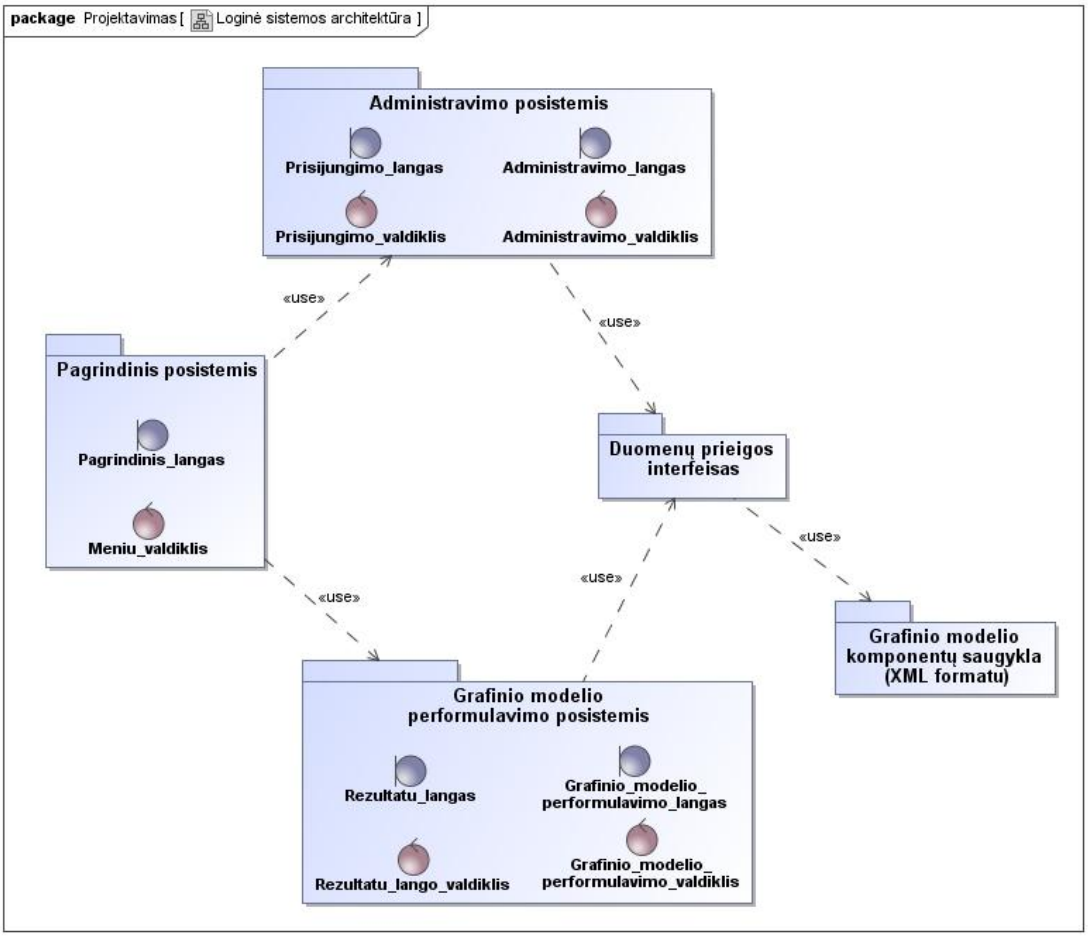
1. Pateiktas grafinis modelis išreikštas UML kalba ir aprašytas „MagicDraw“ įrankiu konvertuojamas į XML struktūros failą
2. Grafinio modelio išreikšto UML kalba automatizavimo įrankis remdamasis pateiktu XML failų skaido išsirenka visus sąryšius atitinkamai pagal naudotojo pasirinktą diagramą
3. Išskirti sąryšiai saugomi duomenų saugyklose iš surenkama kiekvienų jų detali informacija (sąryšio pavadinimas, tipas, vaidmuo tiek sąryšio pradžioje, tiek gale)
4. Pagal išrinktus sąryšius ieškomi su jais susietų esybių pateiktoje XML struktūroje
5. Susietų su sąryšiais esybių detali informacija taip pat išsaugoma
6. Baigus modelio ir aprašyto „MagicDraw“ įrankiu elementų skaidymą pagal pasirinktą diagramą, visi sąryšiai ir susietos esybės formuojami natūralios kalbos sakiniais atsižvelgiant į esybių kardinalumą
7. Tiek esybių pavadinimai, tiek vaidmenys linksniuojami pagal atitinkamas taisykles

1.19. Grafinio modelio performulavimo natūralia kalba įrankio architektūra

Sistemos architektūra apima keturias esminės sistemos veikimo sritis: grafinio modelio pateikimas, pateikto grafinio modelio verbalizavimas, pateikto grafinio modelio esybių ir sąryšių linksniavimas ir rezultatų pateikimas sistemos naudotojui patogiu būdu.

1.19.1. Loginė visos sistemos architektūra

Žemiau pateikiama loginė sistemos architektūra (žr. 3.1 pav.), kurioje išskirtos visos sistemos funkcionalumui reikalingos posistemės bei jų valdikliai.



3.1 pav. Loginė sistemos architektūra

Sistemai grafinis modelis pateikiamas XML formatu, kuris suformuojamas iš suprojektuotos diagramos „MagicDraw“ įrankiu. Pateiktojo grafinio modelio XML formato failas saugo detalią informaciją apie kiekvieną UML projekto elementą. Viename projekte, kas be ko, galima daugiau nei viena diagrama. Kiekvienas diagramos elementas (esybė, sąryšis) taip pat apibūdinamas pagal tam tikrą šabloną pateiktame XML dokumente. Viena esminiu (svarbiausia) sąlygų sistemos (įrankio) veikimui yra pateikto XML dokumento tvarkingumas ir teisingumas. Negalimas joks nukrypimas nuo standartinio, „MagicDraw“ įrankio suformuojamo standarto. Taigi, diagramos XML failo kūrimas taip pat galimas ir XML formatu, tačiau reikalaujamas XML failas turi atitikti „MagicDraw“ įrankio siūlomą struktūrą be priekaištų.

Žemiau lentelėje (žr. 3.1 lentelė) pateikiamas visų loginėje architektūroje naudojamų posistemių aprašymas.

3.1 lentelė. Loginės architektūros posistemių aprašas

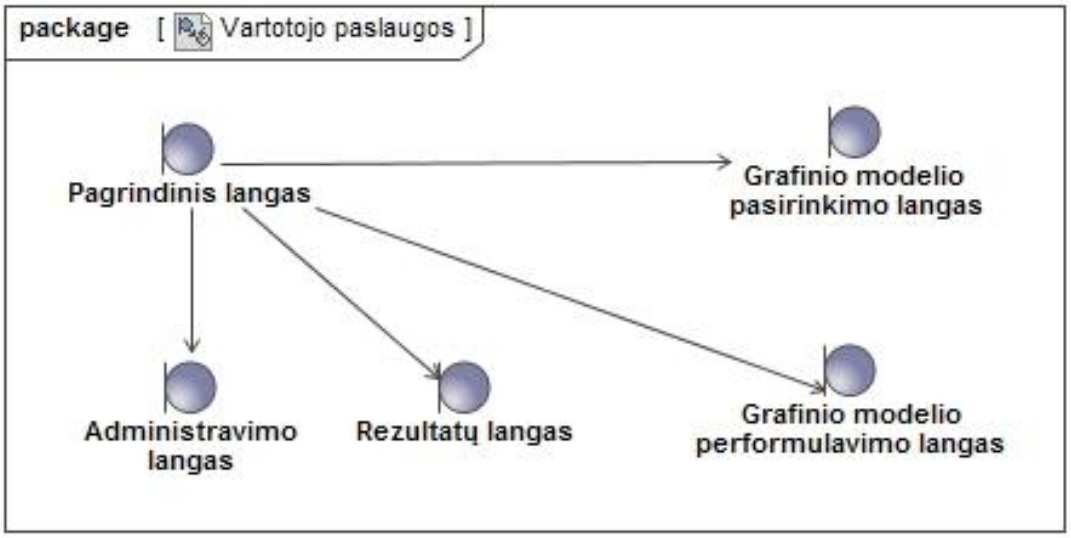
Nr.	Pavadinimas (posistemio)	Aprašymas
1	Administravimo	Naudojama administruoti verbalizacijos žodyno pildymui, redagavimui.
2	Pagrindinis	Atsakingas už pagrindinio lango ir meniu valdymą. Šis posistemis apjungia administravimo ir grafinio modelio išreikšto UML kalba išreiškimo į natūralią kalbą posistemius.
3	Grafinio modelio performulavimo	Posistemis atsakingas už pateikto grafinio modelio išreikšto UML kalba išreiškimą į natūralią kalbą organizavimą.
4	Duomenų prieigos	Duomenų prieigos sąsaja, kuria naudojasi tiek administravimo posistemis (numatytojo žodyno saugojimui), tiek grafinio modelio pateikto UML kalba išreiškimui į natūralią kalbą.
5	Grafinio modelio komponentų saugykla	Pateikto grafinio modelio išreikšto UML kalba jame egzistuojančių diagramų ir jų elementų saugykla.

Visuose posistemiuose veikla kontroliuojama valdiklių pagalba. Kiekvienas jų (valdiklių) yra atsakingas už tam tikrą dalį sistemoje veikiančių procesų. Žemiau pateikiami visų valdiklių aprašai.

- Administravimo posistemis
  - Prisijungimo valdiklis – atsakingas už naudotojo autentifikaciją, kuris siekia redaguoti sistemos žodyną
  - Administravimo valdiklis – atsakingas už žodyno korekcijos funkcijas, leidžiančias redaguoti numatytąjį žodyną
- Pagrindinis posistemis
  - Meniu valdiklis – kontroliuoja pagrindinio lango veiklą
- Grafinio modelio performulavimo posistemis
  - Rezultatų lango valdiklis – pagrindiniame lange esančio rezultatų pateikimo lango valdiklis, pateikiantis sistemos siūlomus rezultatus ir juos išvalantis poreikį
  - Grafinio modelio performulavimo valdiklis – atsakingas už pateikto grafinio modelio išreikšto UML kalba išreiškimą į natūralią kalbą ir pateikimą rezultatų posistemiui

1.19.2. Vartotojo paslaugos

Vartotojo paslaugos apibrėžia vartojo navigaciją po kuriamą įrankį. Kiekvienas iš žemiau pateiktos diagramos (žr. 3.2 pav.) elementų apimą tam tikrą dalį sistemos funkcijų, kurios pasiekiamos remiantis įrankio navigacija.



3.2 pav. Vartotojo paslaugos

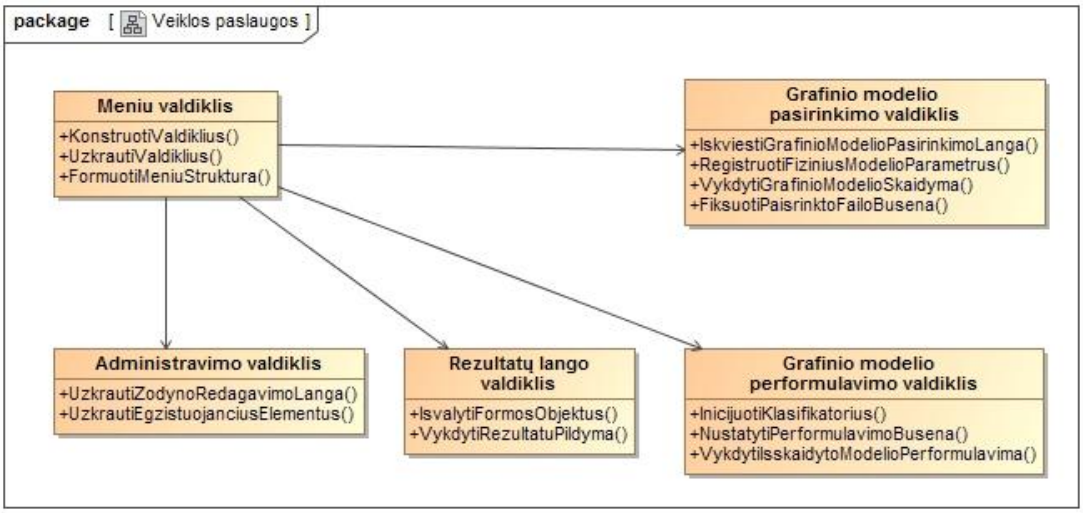
Žemiau lentelėje (žr. 3.2 lentelė) pateikiamas kiekvieno sistemos lango aprašymas (žr. 3.2 pav.).

3.2 lentelė. Sistemos langų aprašas

Nr.	Pavadinimas	Aprašymas
1	Pagrindinis langas	Suteikia kontrolę pradiniame sistemos paleidimo etape. Langas, iš kurio pasiekiami likusieji sistemos langai.
2	Administravimo langas	Įrankyje registruoto žodyno redagavimui ir pildymui reikalingas langas.
3	Grafinio modelio pasirinkimo langas	Langas, kurio pagalba naudotojui suteikiama galimybė pasirinkti iš naudojamo kompiuterio norimą verbalizuoti grafinį modelį aprašytą UML kalba.
4	Grafinio modelio performulavimo langas	Pateikto grafinio modelio išreikšto UML kalba performulavimui skirtas langas, kuriame galime inicijuoti modelio išreiškimą į natūralią kalbą.
5	Rezultatų langas	Pateikto grafinio modelio išreikšto UML kalba išreiškimo į natūralią kalbą rezultatų pateikimo langas.

1.19.3. Veiklos paslaugos

Žemiau pateiktoje veiklos paslaugų diagramoje (žr. 3.3 pav.) matyti pagrindinių valdiklių veikimo (užkrovimo) tvarka bei baziniai metodai, kuriais remiantis bus realizuojamas sistemos funkcionalumas.



3.3 pav. Veikos paslaugos

Žemiau lentelėje pateikiamas visų veiklos paslaugų klasių aprašymai (žr. 3.3 lentelė)

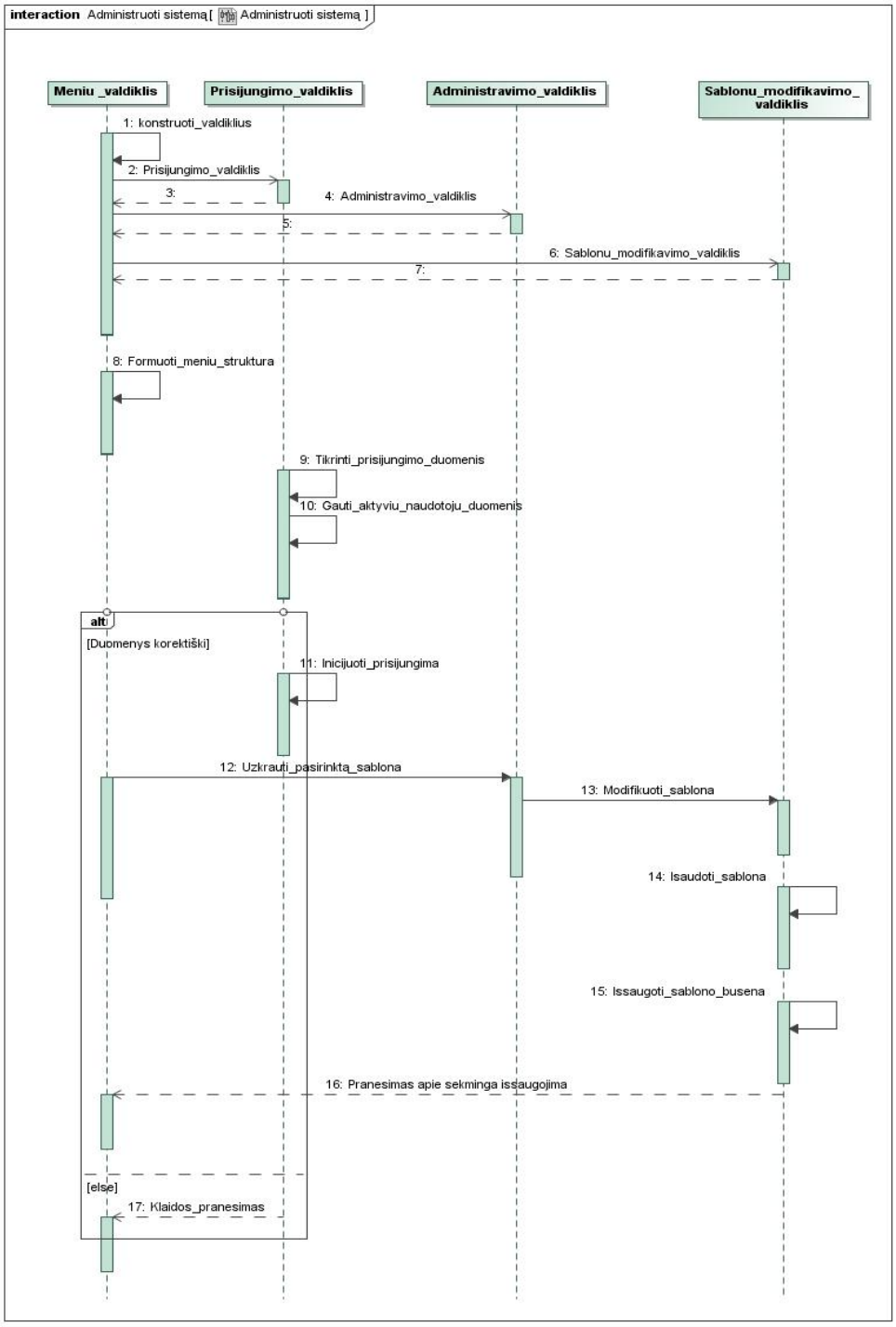
3.3 lentelė. Veiklos paslaugų klasių aprašymas

Nr.	Pavadinimas	Aprašymas
1	Meniu valdiklis	Atsakingas už tinkamą pagrindinio lango funkcionavimą. Tai valdiklis konsoliduojantis visų žemesnio lygmens valdiklių darbą.
2	Administravimo valdiklis	Atsakingas už egzistuojančio žodyno užkrovimą ir pateikimą administravimo lange.
3	Rezultatų lango valdiklis	Atsakingas už rezultatų lango išvalymą ir rezultatų pateikimą jame po sėkmingai užbaigto modelio aprašyto UML kalba išreiškimo į natūralią kalbą.
4	Grafinio modelio performulavimo valdiklis	Valdiklis, kuris atsakingas už pateikto grafinio modelio išreikšto UML kalba išskaidymą pagal jo meta duomenis ir išreiškimą pagal juos į natūralią kalbą.
5	Grafinio modelio pasirinkimo valdiklis	Valdiklis kontroliuojantis grafinio modelio pasirinkimo iš kompiuterio lango veikimą.

1.20. Grafinio modelio performulavimo į natūralią kalbą elgsenos modelis

1.20.1. Grafinio modelio verbalizavimo valdiklių bendravimas

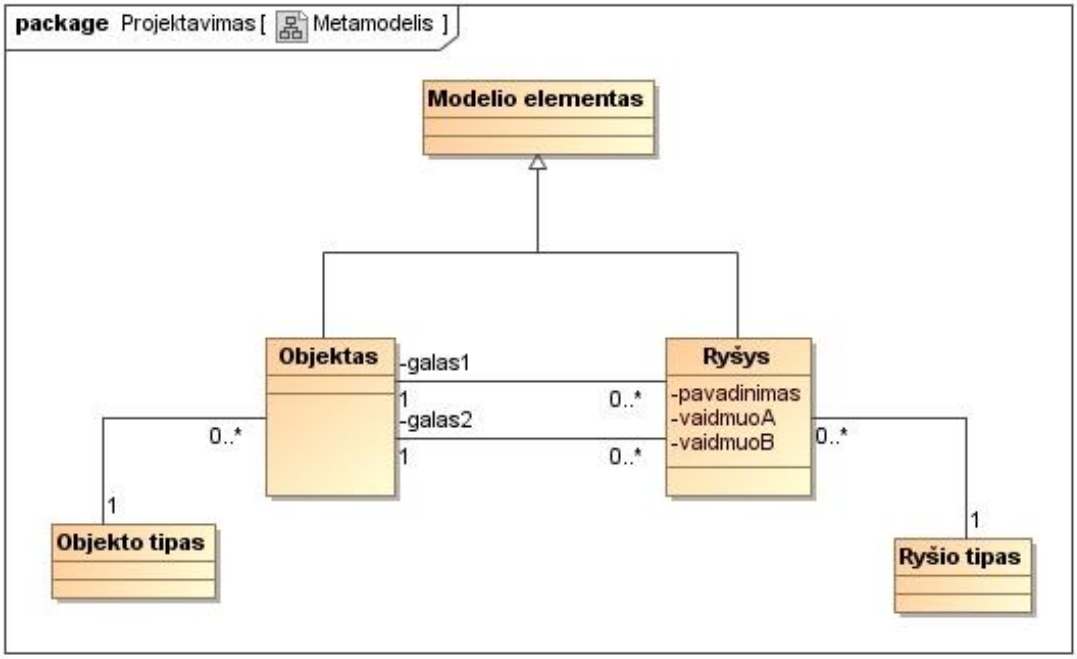
Žemiau pateikiama sekų diagrama, kurioje vaizduojamas administravimo posistemės valdiklių bendravimas (žr. 3.4 pav.)



3.4 pav. Grafinio modelio verbalizavimas

1.21. Detalus projektas

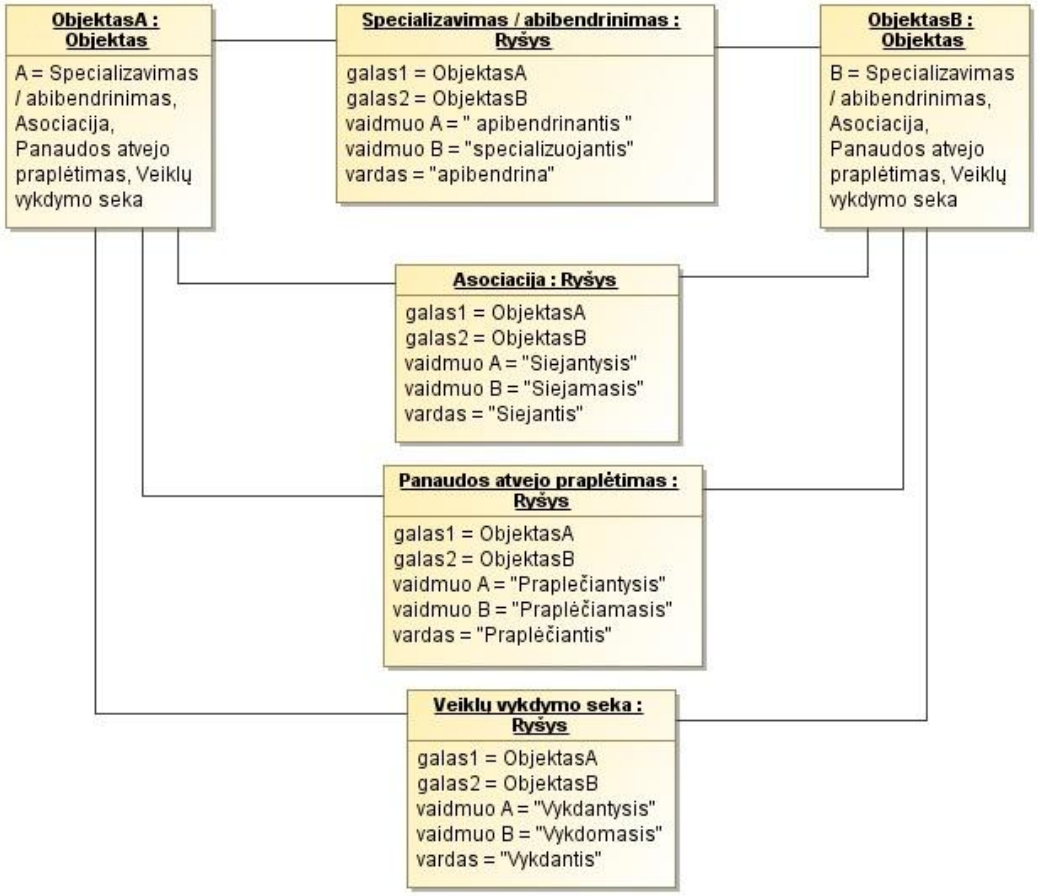
Žemiau pateikiamas grafinio modelio išskaidymui taikytinas metamodelis (žr. 3.5 pav.). Jį sudaro „Modelio elementas“, kuris apibendrina modelyje egzistuojančius objektus ir sąryšius. Kiekvienas objektas ir sąryšis taip pat skirstomi į tipus (pvz., ryšiai - „Asociacija“ ar „Specializavimas“, objektai – „Aktorius“, „Panaudos atvejas“)



3.5 pav. Meta modelis

Žemiau pateikiama grafinio modelių egzempliorių diagrama (žr. 3.6 pav.), kuria perteikiamas grafinio modelio išskaidymas. Egzempliorius apibrėžia šiuos sąryšius:

- Specializavimas – apibendrinimas
- Asociacija
- Panaudos atvejo praplėtimas
- Veiklų vykdymo seka

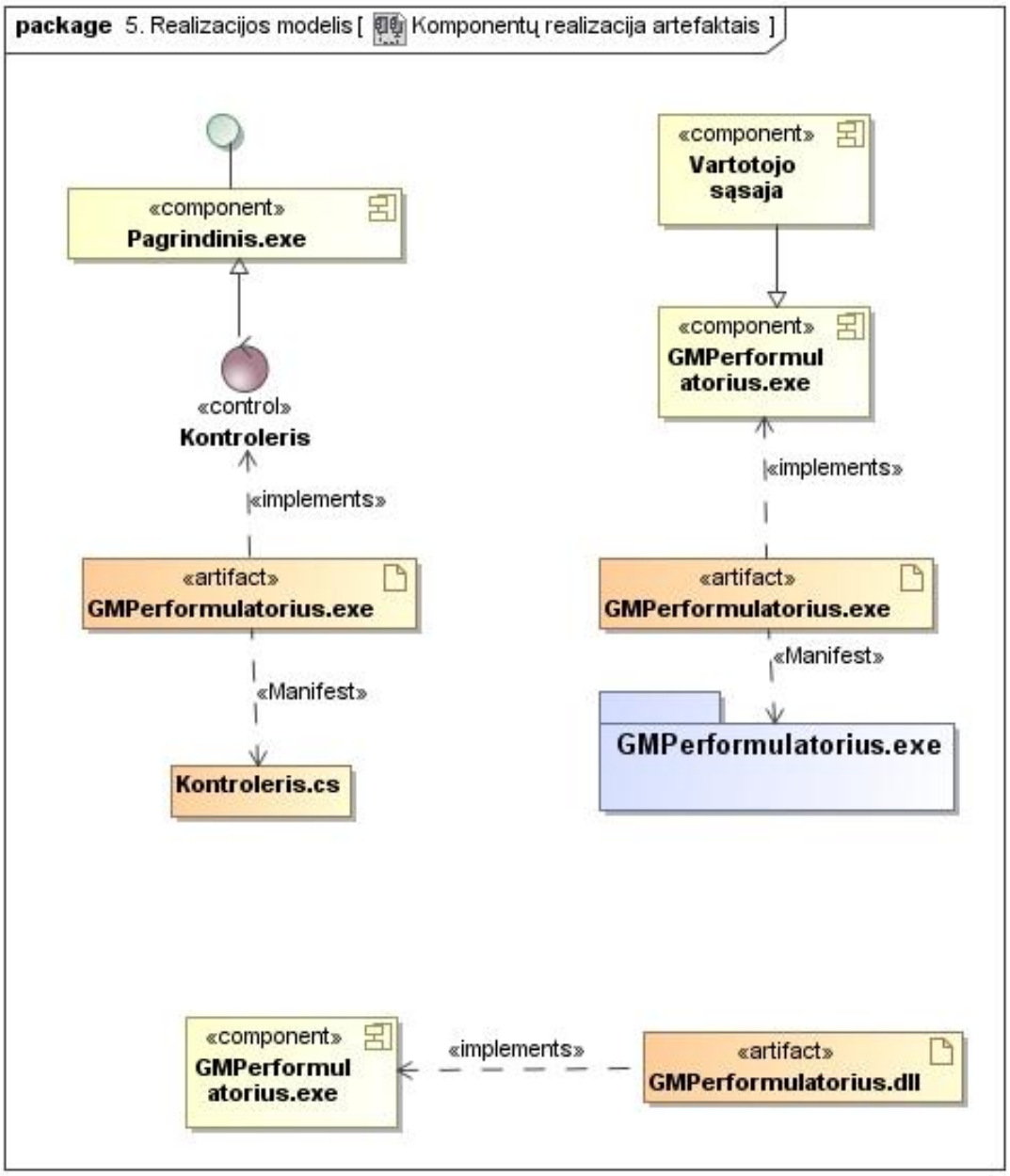


3.6 pav. Grafinio modelio egzempliorių diagrama

1.22. Realizacijos modelis

1.22.1. Programinių komponentų architektūra

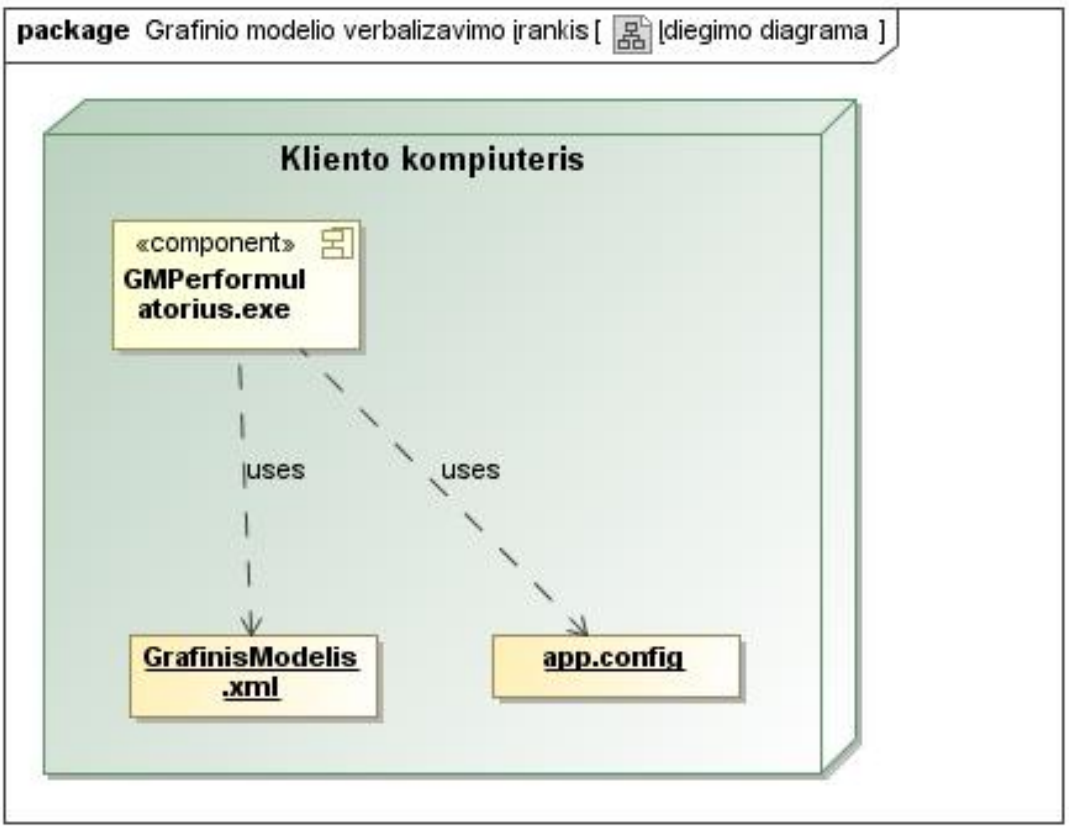
Skyrius skirtas pagrindinių sistemos komponentų sąsajos pateikimui. Žemiau pateikiamas sistemos komponentų modelis (žr. 3.7 pav.), kuriame pateikiami komponentai bei sąryšiai tarp jų komponentų, kurių pagrindu funkcionuoja grafinio modelio performulavimo įrankis.



3.7 pav. Komponentų realizacija artefaktais

1.22.2. Diegimo modelis

Grafinio modelio performulavimo į lietuvių kalbą įrankiui egzistuoti reikalingi komponentai pateikti paveikslėlyje žemiau (žr. 3.8 pav.). Visi komponentai egzistuoja kliento kompiuteryje. Duomenų saugykla XML formatu formuojama automatiškai (vartotojo įsikyšimas nėra reikalingas).



3.8 pav. Įdiegimo diagrama

4. Automatizuoto įrankio realizacija

1.23. Realizacijos ir veikimo aprašymas

Poskyryje pateikiamas grafinio modelio išreikšto UML kalba išreiškimas natūralia kalba automatizavimo įrankio aprašas. Realizuotas įrankis buvo pavadintas „Grafinio modelio verbalizatorius“. Pagrindinė įrankio funkcija – pateikto grafinio modelio išreikšto UML kalba išreiškimas natūralia kalba. Realizuotas įrankis geba performuluoti klasių diagramos tipo grafinį modelį (aprašytą objektiniu ar esybių – sąryšių požiūriu) išreikštą UML kalba į natūralią kalbą.

1.23.1. Veikimo aprašymas

Remiantis darbo projektine dalimi, buvo sukurtas įrankis, leidžiantis jo vartotojui pateikti grafinį modelį išreikštą UML kalba ir aprašyta „MagicDraw“ įrankiu verbalizuoti natūralia kalba. Realizuotas įrankis visiškas išbaigtas pateikiant klasių diagramos grafinį modelį, kuris sudarytas iš klasių ir binarinės asociacijos elementų. Diagramą galima pateikti tiek esybių – ryšių požiūriu, tiek objektiniu požiūriu.

Norint pateiktą grafinio modelį išreikšto UML kalba pateikti natūralia kalba, vykdomi šie žingsniai:

1. Sukuriamas grafinis modelis išreikštas UML kalba naudojant „MagicDraw“ įrankį;
2. Sukurtas grafinis modelis išsaugomas XML formatu taip pat naudojant „MagicDraw“ įrankį. Pastaba: išsaugojus grafinį modelį XML formatu, atliekami pakeitimai „MagicDraw“ įrankyje bus saugomi būtent išsaugotame XML projekte, t.y. originalus MdZip failas nebus koreguojamas;
3. Išsaugotas grafinis modelis XML formatu pateikiamas modelio išreikšto UML kalba išreiškimo natūralia kalba verbalizatoriui;
4. Modelio verbalizatorius skaido pateikto grafinio modelio elementus ir pagal apibrėžtas taisykles formuoja natūralios kalbos sakinius;

Klasių diagramos išreiškimas natūralia kalba taikant esybių – ryšių modelių verbalizavimo taisykles ir klasių diagramos verbalizavimo taisykles (objektinis požiūris) vykdomas skirtingai. Žemiau pateikiama rezultatų formavimo idėja abiejų požiūrių verbalizavime: objektinis požiūris ir esybių – ryšių požiūris.

Objektinis požiūris

Objektiniame požiūryje dominuoja asociacijos skaitymo kryptis. Tai, kuria kryptimi įrankis privalo remtis skaitydamas modelį, informuoja krypties rodyklė uždėta ant asociacijos.

Žemiau pateikiamas objektinio požiūrio klasių diagramos natūralios kalbos sakinio šablonas (žr. 4.1 lentelė), kuris formuojamas pagal diagramoje (žr. 4.1 pav.) pateiktų elementų pavadinimus.

4.1 lentelė. Objektinio požiūrio verbalizavimo šablonas

Nr.	Šablonas
1	[sąryšio pradžios modelio elemento **pavadinimas**] + [sąryšio **pavadinimas**] + [sąryšio galo **kardinalumas**] + [sąryšio galo modelio elemento **pavadinimas linksnyje**].
2	[sąryšio pradžios vaidmens **pavadinimas**] + *yra* + [sąryšio pradžios modelio elemento **pavadinimas**]
3	[sąryšio galo vaidmens **pavadinimas**] + *yra* + [sąryšio galo modelio elemento **pavadinimas**]

Pateikto pavyzdžio (žr.) natūralios kalbos sakiniai:

Asmuo turi daug namų.

Savininkas yra asmuo.

Turtas yra namas.

4.1 pav. Objektinio požiūrio realizavimas

Esybių – ryšių požiūris

Sudarius modelį laikantis esybių – ryšių modeliavimo taisyklių, skirtingai nei klasių diagramoje, turime galimybę perskaityti sąryšį tarp esybių iš abejomis kryptimis.

Žemiau pateikiamas esybių – ryšių principu sudarytos klasių diagramai verbalizuoti skirtas natūralios kalbos sakinio šablonas (žr. 4.2 lentelė), kuris formuojamas pagal diagramoje (žr. 4.2 pav.) pateiktų elementų pavadinimus.

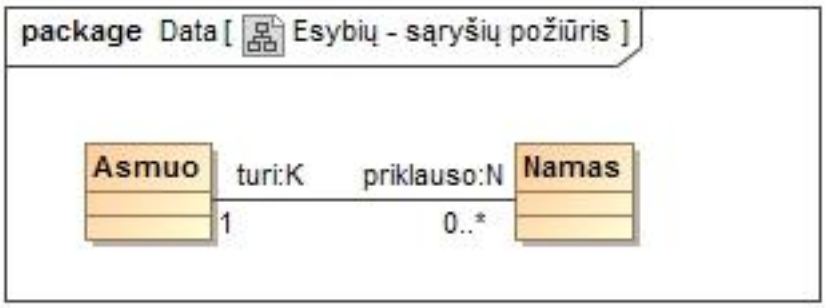
4.2 lentelė. Esybių - ryšių verbalizavimo šablonas

Nr.	Šablonas
1	[sąryšio pradžios modelio elemento **pavadinimas**] + [sąryšio pradžios vaidmens **pavadinimas**] + [sąryšio galo **kardinalumas**] + [sąryšio galo modelio elemento **pavadinimas linksnyje**].
2	[sąryšio galo modelio elemento **pavadinimas**] + [sąryšio galo vaidmens **pavadinimas**] + [sąryšio pradžios **kardinalumas**] + [sąryšio pradžios modelio elemento **pavadinimas linksnyje**].

Pateikto pavyzdžio (žr.) natūralios kalbos sakiniai:

Asmuo turi daug namų.

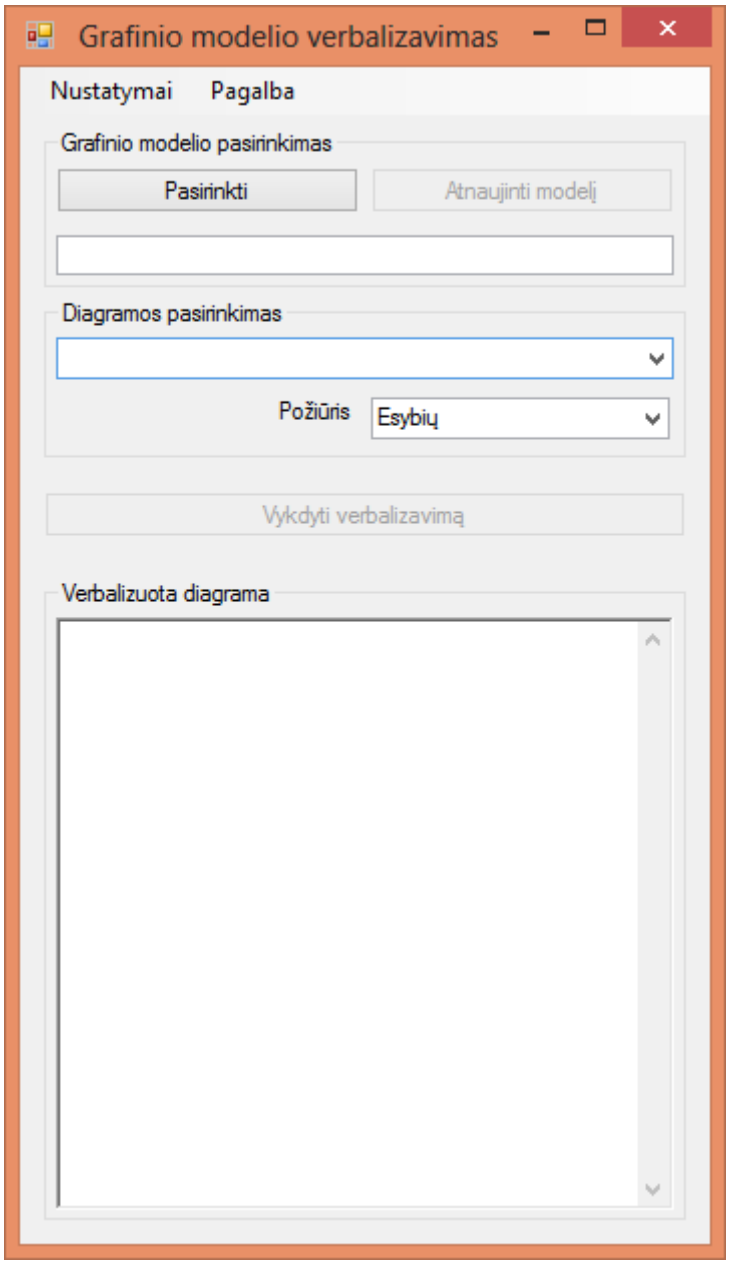
Namas priklauso asmeniui.



4.2 pav. Esybių - ryšio požiūrio realizavimas

1.23.2. Grafinio modelio performulavimo įrankio grafinė sąsaja

Žemiau pateikiamas ir aptariamas pagrindinis įrankio langas (žr. 4.3 pav.). Pagrindinio lango komponentų aprašymas pateiktas lentelėje (žr. 4.3 lentelė).

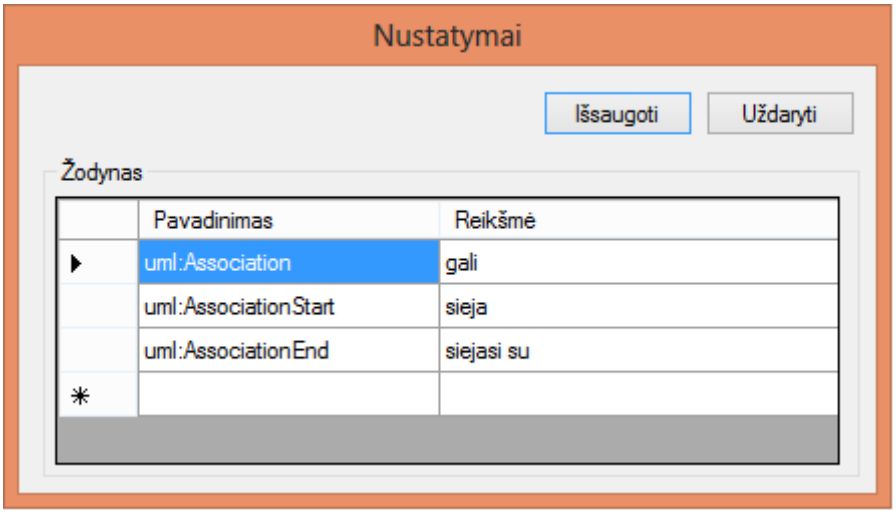


4.3 pav. Grafinė sąsaja

4.3 lentelė. Pagrindinio lango komponentų aprašymas

Nr.	Komponentas	Aprašymas
1	Meniu	Meniu pagalba pasiekiamas nustatymų modulis, kuriame galime koreguoti įrankio verbalizavimo žodyną.
2	Grafinio modelio pasirinkimas	Komponentas iškviečia grafinio modelio pasirinkimo iš lokalaus kompiuterio langą.
3	Diagramos pasirinkimas	Iš pateikto modelio išskaidytų diagramų galima pasirinkti norimą verbalizuoti diagramą.
4	Verbalizuojamas požiūris	Pasirenkamas verbalizavimo požiūris: **esybių – sąryšių** arba **objektinis**.
5	Vykdymas ir rezultatų langas	Inicijuojamas modelio verbalizavimas ir rezultatų pateikimas.

Tiesiogiai iš meniu punkto „Nustatymai“ pasiekiamas nustatymų langas, kuriame turime galimybę pildyti ir redaguoti sistemos žodyną, kuris naudojamas grafinį modelį išreikštą UML kalba išreiškiant natūralia kalba. Žemiau pateikiamas nustatymų langas (žr. 4.4 pav.) ir žodyno lentelės stulpelių aprašas (žr. 4.4 lentelė).



4.4 pav. Grafinė sąsaja nustatymų komponentui

4.4 lentelė. Nustatymų komponentų aprašymas

Nr.	Stulpelio pav.	Aprašymas
1	Pavadinimas	Grafinio modelio elemento pavadinimas pagal „MagicDraw“ įrankio sugeneruotą pavadinimą.
2	Reikšmė	Pateikiama modelio elemento vertimo reikšmė.

1.24. Testavimo modelis

Įrankio testavimas buvo vykdomas remiantis jai keliamais funkciniais ir nefunkciniais reikalavimais.

Kadangi projektavimo procesas buvo pagrįstas įteratyviuoju projektavimo metodu, po kiekvienos didesnės funkcijos ar funkcijos grupių įgyvendinimo, buvo atliekamas išsamus įrankio veikimo testavimas pagal funkcinius reikalavimus. Tokio testavimo metu, būdavo patikrinamas kiekvieno panaudos atvejo veikimo korektiškumas.

Įrankio nefunkciniai reikalavimai buvo testuojami įgyvendinus visą įrankio funkcionalumą. Nefunkcinių reikalavimų testavimas įtraukė sistemos greitaveikos ir kitus parametrus, kurių veikimas gali būti patvirtintas tik tuomet, kai jau visos sistemos funkcijos yra veikiančios ir pilnai darančios įtaka veikimo laikui.

1.25. Testavimo duomenys ir rezultatai (kontrolinis pavyzdys)

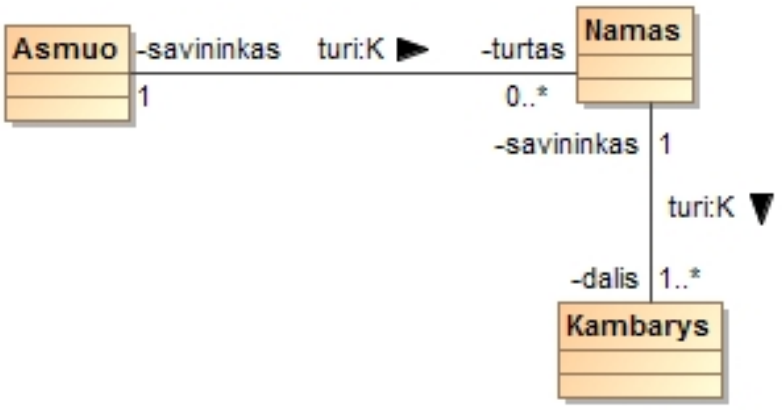
Ruošiant testavimo duomenis buvo atsižvelgta į grafinio modelio išreikšto UML kalba išreiškimą natūralia kalba objektiniu požiūriu ir esybių – sąryšiu požiūriu.

Klasių diagramos su asociacijos sąryšiais tarp objektų grafinis modelis yra tobulas pavyzdys kaip skiriasi grafinio modelio išreikšto UML kalba išreiškimas natūralia kalba objektiniu ir esybių – sąryšių požiūriais.

Žemiau pateikiami testavimo duomenys ir rezultatai pagal skirtingus požiūrius.

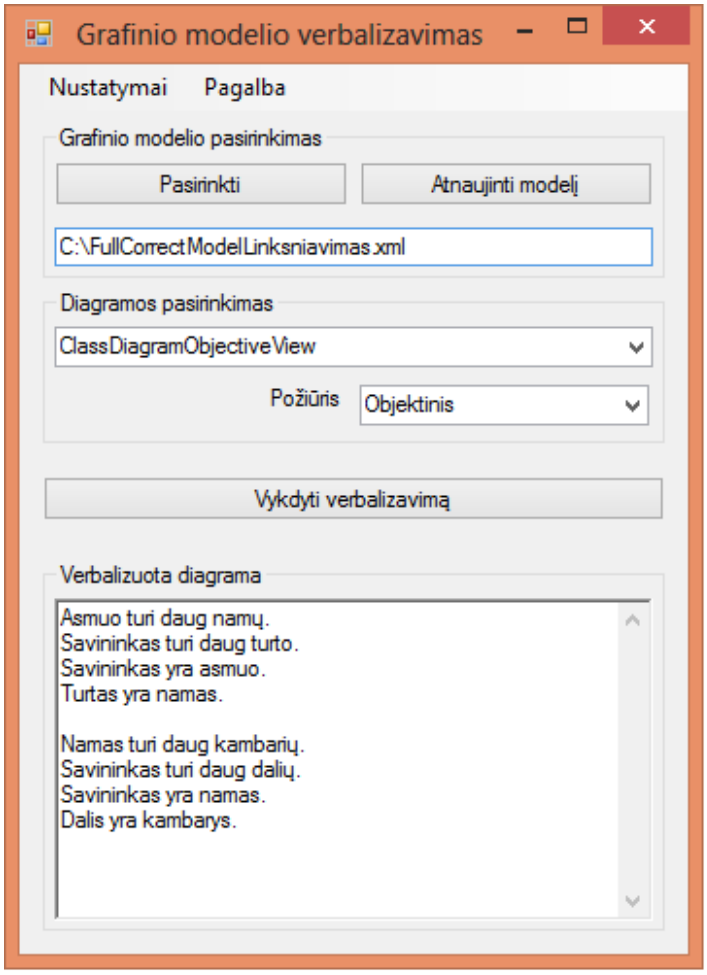
Objektinis požiūris

Žemiau pateikiamas grafinis klasių diagramos modelis išreikštas UML kalba pateiktas objektiniu požiūriu (žr. 4.5 pav.).



4.5 pav. Objektinio požiūrio pavyzdys

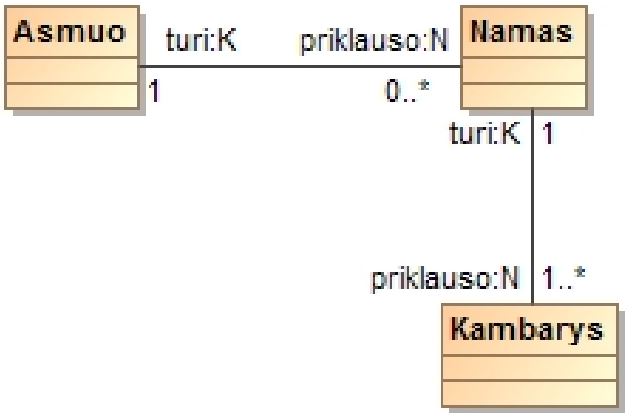
Žemiau pateikiamas įrankio suformuotas rezultatas pagal pateiktą grafinį modelį išreikštą UML kalba ir pateiktą objektiniu požiūriu (žr. 4.6 pav.).



4.6 pav. Objektinio požiūrio rezultatas

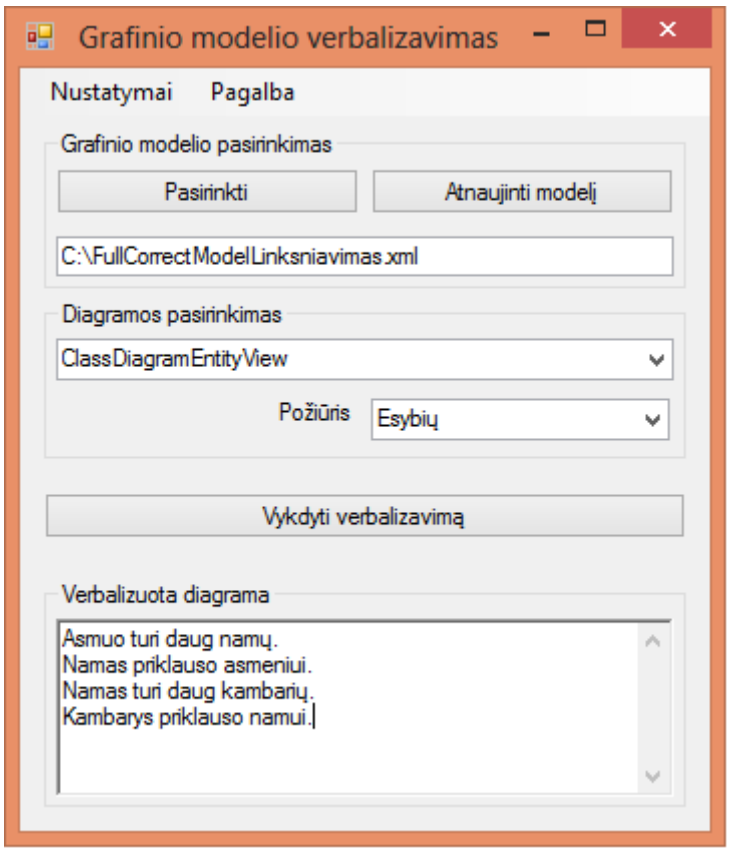
Esybių – sąryšių požiūris

Žemiau pateikiamas grafinis klasių diagramos modelis išreikštas UML kalba pateiktas esybių – sąryšių požiūriu (žr. 4.7 pav.).



4.7 pav. Esybių požiūrio pavyzdys

Žemiau pateikiamas įrankio suformuotas rezultatas pagal pateiktą grafinį modelį išreikštą UML kalba ir pateiktą esybių – sąryšių požiūriu (žr. 4.8 pav.)



4.8 pav. Esybių požiūrio rezultatas

Pateikto grafinio modelio išreikšto UML kalba ir aprašyto „MagicDraw“ įrankiu meta duomenų struktūros fragmentas pateikiamas XML formatu (žr. 4.9 pav. ir 4.10 pav.).

Pateiktuose grafinio modelio išreikšto UML kalba įrankyje „MagicDraw“ pateikiamuose XML fragmentuose matyti pateiktame grafiniame modelyje egzistuojančių diagramų pavadinimai („ClassDiagramEntityView“, „ClassDiagramObjectiveView) ir identifikatoriai.

Taip pat matome klasių diagramose naudojamų esybių pavadinimus ir identifikatorius (Asmuo, Namas, Kambarys), vaidmenis (savininkas, turtas, dalis) bei sąryšio pavadinimus (turi, priklauso).

```
<xmi:Extension extender='MagicDraw UML 17.0.3'>
    <modelExtension>
        <ownedDiagram xmi:type='uml:Diagram' xmi:id='_17_0_3_1_461365_2981' name='ClassDiagramEntityView'
        <ownedDiagram xmi:type='uml:Diagram' xmi:id='_17_0_3_1_282400_3002' name='ClassDiagramObjectiveView'
    </modelExtension>
</xmi:Extension>
```

4.9 pav. Metaduomenų struktūros fragmentas (pagrindinė modelio informacija)

```
<packagedElement xmi:type='uml:Class' xmi:id='_17_0_3_1_ab602d7_1368211536382_439841_3024' name='Asmuo'>
    <ownedAttribute xmi:type='uml:Property' xmi:id='_17_0_3_1_ab602d7_1368211553820_673953_3090' name='turtas'
        <lowerValue xmi:type='uml:LiteralInteger' xmi:id='_17_0_3_1_ab602d7_1368215838124_238776_2892'/>
        <upperValue xmi:type='uml:LiteralUnlimitedNatural' xmi:id='_17_0_3_1_ab602d7_1368215838124_680902_2893' value='*'/>
    </ownedAttribute>
    <ownedAttribute xmi:type='uml:Property' xmi:id='_17_0_3_1_ab602d7_1368216009529_958401_3238' name='priklauso:N'
        <lowerValue xmi:type='uml:LiteralInteger' xmi:id='_17_0_3_1_ab602d7_1368216037968_930972_3282'/>
        <upperValue xmi:type='uml:LiteralUnlimitedNatural' xmi:id='_17_0_3_1_ab602d7_1368216037968_821762_3283' value='*'/>
    </ownedAttribute>
</packagedElement>
<packagedElement xmi:type='uml:Class' xmi:id='_17_0_3_1_ab602d7_1368211541976_161488_3045' name='Namas'>
    <ownedAttribute xmi:type='uml:Property' xmi:id='_17_0_3_1_ab602d7_1368211553836_330372_3091' name='savininkas'
    <ownedAttribute xmi:type='uml:Property' xmi:id='_17_0_3_1_ab602d7_1368211556258_521942_3111' name='dalis'
    <ownedAttribute xmi:type='uml:Property' xmi:id='_17_0_3_1_ab602d7_1368216009529_787845_3239' name='turi:K'
    <ownedAttribute xmi:type='uml:Property' xmi:id='_17_0_3_1_ab602d7_1368216012185_104895_3259' name='priklauso:N'
</packagedElement>
<packagedElement xmi:type='uml:Class' xmi:id='_17_0_3_1_ab602d7_1368211545757_849681_3066' name='Kambarys'>
<packagedElement xmi:type='uml:Association' xmi:id='_17_0_3_1_ab602d7_1368211553820_234082_3089' name='turi:K'>
<packagedElement xmi:type='uml:Association' xmi:id='_17_0_3_1_ab602d7_1368211556258_302965_3110' name='turi:K'>
```

4.10 pav. Metaduomenų struktūros fragmentas (modelio elementai)

5. Eksperimentinis sistemos tyrimas

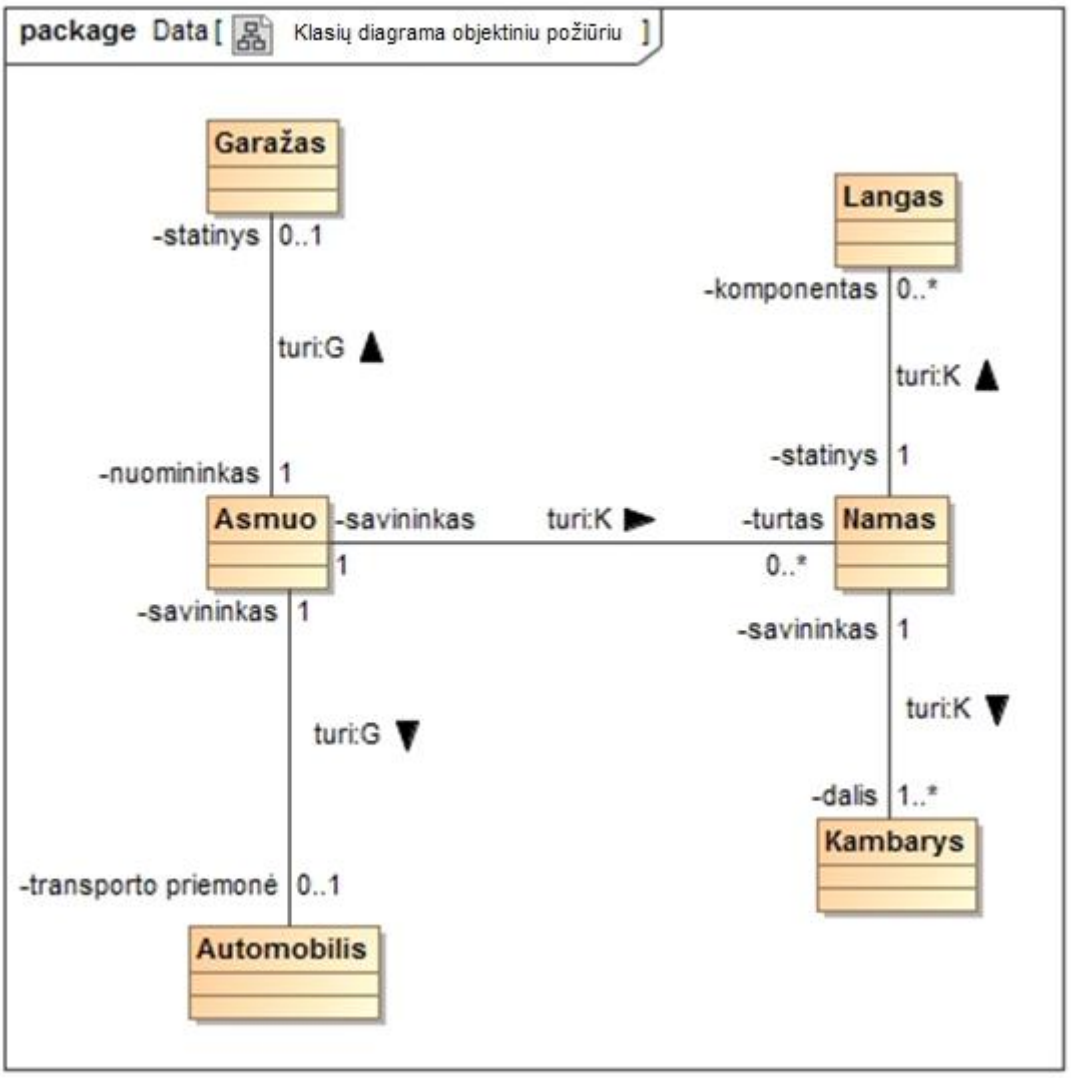
1.26. Eksperimento planas

Eksperimento tyrimui atlikti buvo sukurtas nesudėtingas grafinis modelis išreikštas UML kalba ir aprašymas „MagicDraw“ įrankiu. Eksperimento eiga bus vykdoma pagal realizacijos skyriuje apibrėžto įrankio veikimo aprašymo eigą (žr. punktą 5.1.1).

1.27. Eksperimentas

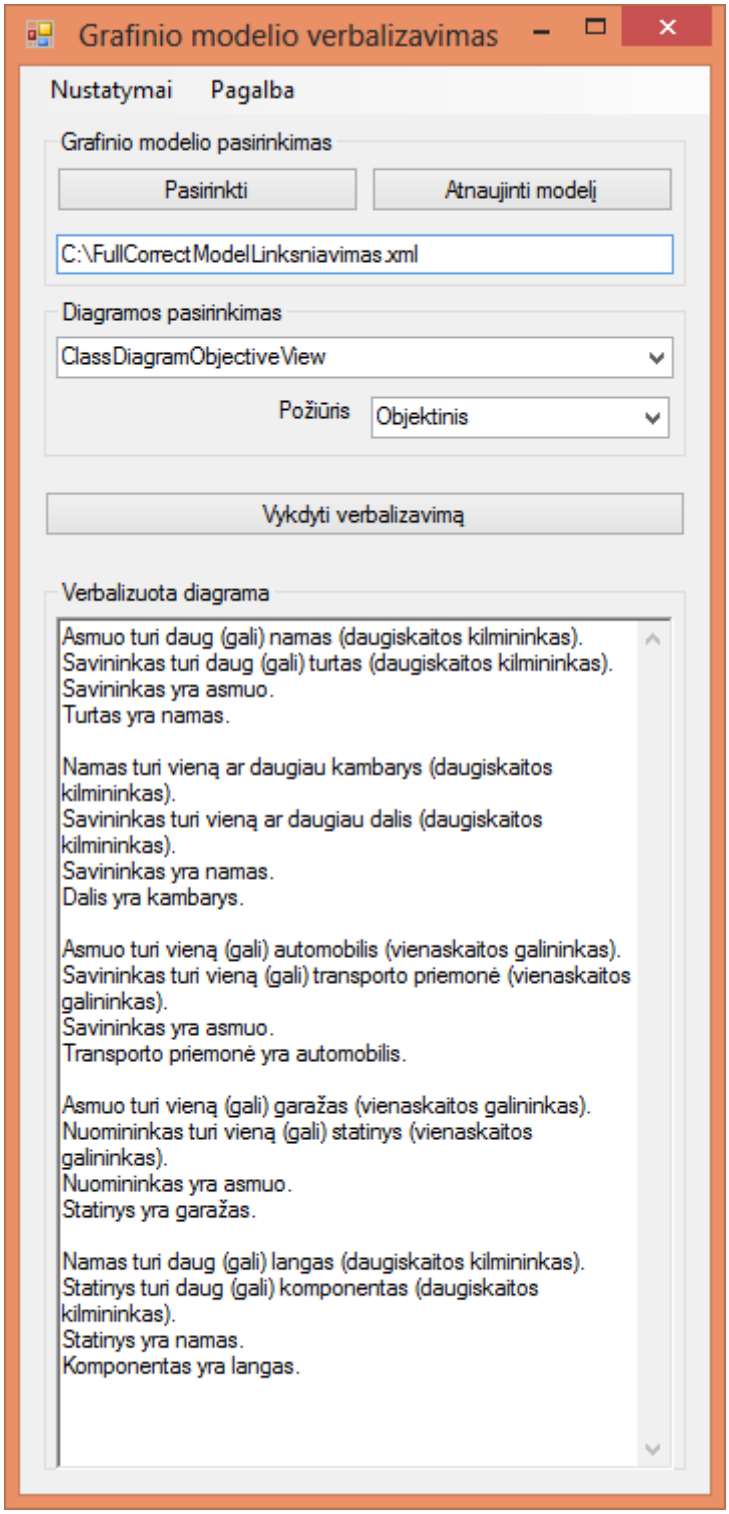
1.27.1. Didelės apimties korektiškas modelis objektiniu požiūriu

Pateikiamas didelės apimties korektiškai suformuluotas grafinis modelis (žr. 5.1 pav.) išreikštas UML kalba, formuluojamas objektiniu požiūriu ir aprašytas „MagicDraw“ įrankiu. Siekiamas rezultatas – sėkmingas modelio performulavimas natūralia kalba.



5.1 pav. Klasių diagrama objektiniu požiūriu

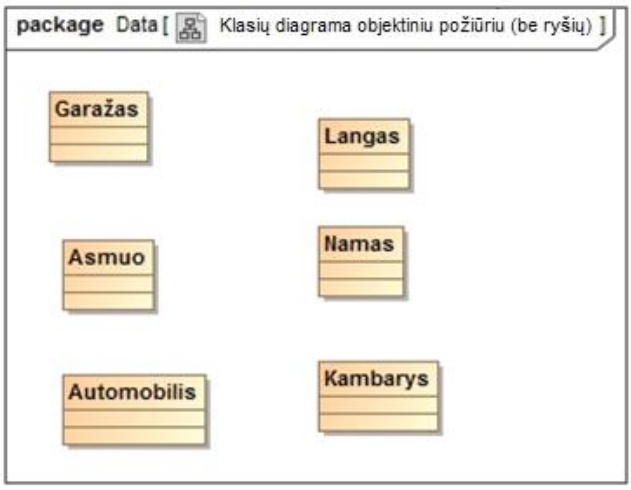
Žemiau pateikiamas įrankio rezultatas kuris atitinka įrankiui ir modeliui išreikšto UML kalba keliamus reikalavimus (žr. 5.2 pav.)



5.2 pav. Klasių diagramos objektiniu požiūriu rezultatas

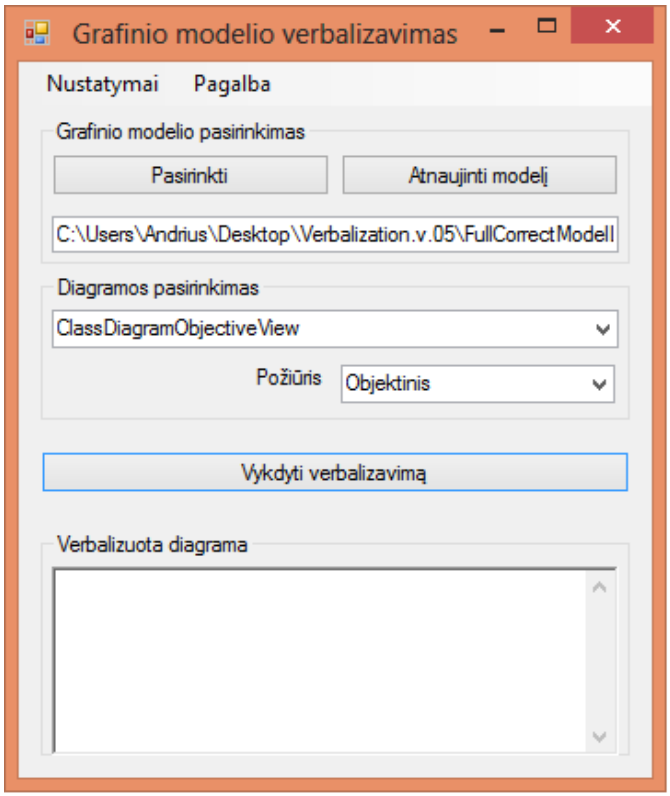
1.27.2. Nekorektiškas grafinis modelis

Pateikiamas nekorektiškai suformuluotas grafinis modelis (žr. 5.3 pav.). Siekiama, kad įrankis nesutriktų ir toliau gebėtų sėkmingai atlikti naujas vartotojo pateiktas užduotis.



5.3 pav. Klasių diagrama objektiniu požiūriu (be ryšių)

Žemiau pateikiamas įrankio rezultatas kuris atitinka įrankiui ir modeliui išreikšto UML kalba keliamus reikalavimus (žr. 5.4 pav.). Kaip matome, įrankis rezultatų nepateikė.



5.4 pav. Klasių diagramos objektiniu požiūriu (be ryšių) rezultatas

1.28. Eksperimento rezultatai

Eksperimento metu pavyko išgauti pateikto grafinio modelio išreikšto UML kalba natūralios kalbos sakinius.

Įrankis sėkmingai dirba tik su klasės diagrama, kurioje egzistuoja klasės tipo elementai ir paprastieji binariniai sąryšiai (asociacijos). Diagrama gali būti tiek esybių – sąryšių, tiek objektinio požiūrių.

Išmėginta su įvairių dydžių (elementų kiekis vienoje diagramoje) modeliais. Taip pat eksperimentuota su nekorektiškai suformuotais modeliais.

6. Išvados

1. Atlikus analizę, nustatyta, kad nėra įrankių galinčių grafinį modelį išreikštą UML kalba performuluoti natūralia kalba, jei grafinio modelio kūrimui laikomasi objektinio požiūrio. Laikantis skirtingų požiūrių sistemų kūrimo metodai naudoja skirtingas taisykles statiniams sistemos aspektams pavaizduoti grafiniame modelyje, t.y. klasių diagramoje ir esybių – ryšių diagramoje;
2. Dalykinių sričių žinovų poreikis detaliau aprašyti grafinius modelius išreikštus UML kalba ir atlikus poreikių analizę neaptikus analogiškų sprendimų nustatyta, kad įrankio kūrimas yra reikalingas;
3. Pagal sudarytą grafinio modelio išreikšto UML kalba performulavimui natūralia kalba įrankio algoritmą nuspręsta, kad jis tinka grafinio modelio performulavimo įrankio kūrimui;
4. Atliktas eksperimentinis sukurto įrankio įvertinimas parodė, kad įrankis leidžia grafinį modelį išreikštą UML kalba, kuris aprašo modelį laikantis esybių – ryšių, tiek objektiniu požiūriu, tinkamai performuluoti į natūralią kalbą.

7. Literatūra

1. Rita Butkienė: „Towards a verbalization of the conceptual model expresses in lithnuanian language“.

2. Mustafa Jarrar: „Towards Methodological Principles for Ontology Engineering”.

3. The Unified Modeling Language (UML) [žiūrėta 2013-04-28]. Prieiga per internetą: http://www.sparxsystems.com/platforms/uml.html

4. UML 2 Class Diagram [žiūrėta 2013-04-28]. Prieiga per internetą: http://www.sparxsystems.com/resources/uml2_tutorial/uml2_classdiagram.html

5. UML 2 Use Case Diagram [žiūrėta 2013-04-28]. Prieiga per internetą: http://www.sparxsystems.com/resources/uml2_tutorial/uml2_usecasediagram.html

6. UML 2 Activity Diagram [žiūrėta 2013-04-28]. Prieiga per internetą: http://www.sparxsystems.com/resources/uml2_tutorial/uml2_activitydiagram.html