KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACIJOS SISTEMŲ KATEDRA

Vidas Trijonis

Vartotojo įvedamos ir išvedamos informacijos specifikavimo konceptualizavimas

Magistro darbas

Darbo vadovas
dr. doc. Rimantas Butleris

KAUNAS, 2005

Turinys

Summary ... 4
Sutrumpinimų sąrašas ... 5
Įvadas ... 6
1 Vartotojo reikalavimų specifikavimo analizė ... 7
1.1 Vartotojo reikalavimų specifikavimo proceso aktualumas ... 7
1.2 Vartotojo reikalavimų specifikavimo analizės metodų, priemonių parinkimas ... 8
1.3 Vartotojų reikalavimų specifikavimo analizės rezultatai ... 8
1.3.1 Vartotojo reikalavimų specifikavimo srities analizė 8
1.3.1.1 R. Barker vartotojo reikalavimų specifikavimo metodika ... 9
1.3.1.2 RUP metodika ... 9
1.3.1.3 Rummler siūloma vartotojo reikalavimų specifikavimo metodika ... 10
1.3.1.4 Formomis paremta objektiškai orientuota atvirkštinės inžinerijos metodika ... 11
1.3.1.5 Veiklos procesų atvirkštinė inžinerija: formomis paremtas požiūris ... 12
1.3.1.6 Vartotojo reikalavimų specifikavimas, paremtas formų analize ... 12
1.3.2 Vartotojo reikalavimų specifikavimo metodikų lyginamoji analizė 13
1.3.3 Vartotojo reikalavimų specifikavimo metodikos kokybės kriterijų apibrėžimas 14
1.4 Vartotojų reikalavimų specifikavimo sistemos varianto parinkimas ... 15
2 Apribojimai vartotojo reikalavimų specifikavimui ... 17
2.1 Vartotojo reikalavimų specifikavimo panaudojimo atvejų modeliai ... 17
2.1.1 Vartotojo reikalavimų specifikavimo veiklos modeliai 19
2.1.2 Vartotojo reikalavimų specifikavimo klasių modelis 21
2.1.3 Vartotojo sąsajos modelis reikalavimų specifikavimui 22
3 Vartotojo reikalavimų specifikavimo metodikos sistemos projektas ... 24
3.1 Vartotojo reikalavimų specifikavimo metodikos panaudojimo atvejų modelis ... 24
3.2 Vartotojo reikalavimų specifikavimo metodikos sekų modeliai ... 25
3.3 Vartotojo reikalavimų specifikavimo ir metodikos klasių modeliai ... 27
3.4 Vartotojo reikalavimų specifikavimo metodikos panaudojimo atvejų realizacijos modelis 28
3.5 Duomenų bazės loginė schema ... 30
3.5.1 Vartotojo reikalavimų specifikavimo metodikos realizacijos modelis 33
4 Programinė vartotojo reikalavimų specifikavimo realizacija ... 35
4.1 Vartotojo reikalavimų specifikavimo įrankio struktūra ... 36
4.2 Realizuotos funkcijos ... 37

4.2.1 Vartotojo reikalavimų specifikavimo įvedamos ir išvedamos informacijos nustatymas 37
4.2.2 Vartotojo reikalavimų specifikavimo sluoksnių realizavimas 39
4.2.3 Grafinių žymėjimų realizacija 40
4.2.4 Vartotojo reikalavimų specifikavimo sluoksnių funkcijų realizacija 41
4.2.5 Vartotojo reikalavimų specifikavimo ryšio su metaduomenų baze realizavimas 46
5 Vartotojo reikalavimų specifikavimo realizacijos įvertinimas 50
5.1 Vartotojo įvedamos ir išvedamos informacijos specifikavimo metodikos įvertinimas 50
5.2 Vartotojo reikalavimų specifikavimo duomenų saugyklos įvertinimas 51
5.3 Vartotojo įvedamos ir išvedamos informacijos specifikavimo prototipo įvertinimas 52
Išvados 54
Literatūros šaltiniai 55
Priedai 57

Summary

The conceptualization of specification of users' requirements to incoming and out coming information

The specification of requirements is one of the most important stages in the development of information system. The correct description is one of the principle goals in preparation of specification of requirements. Well-done specification of information system is a significant support to create software with high functionality, low maintenance and adaptation costs.

Filled in document forms play an important role in the analysis of user requirements. This analysis helps to reduce the gap between the user and the system analyst. The methods of form analysis are oriented in creation of an entity relationship scheme.

Information system department offering user requirements specification process of functional requirements specification method is similar to natural requirements analysis proceeding. This process begins from specification of users' requirements to incoming and out coming information.

This work presents the model and prototype for specification of users' requirements to incoming and out coming information. The model is based on analysis of documents forms given by user. Method is divided into three phases: 1) user incoming and out coming information, 2) data structure analysis, 3) conceptual schema generation.

The meta-model repository of requirements specification is presented. The specification process and prototype of specification of incoming and out coming information of user’s requirements are described.

Sutrumpinimų sąrašas

CASE (Computer Aided System Engineering) – automatizuotas projektavimas

EPRE (Enterprise process reverse engineering) – veiklos procesų atvirkštinė inžinerija

FORE ((Form Driven Object-Oriented Reverse Engineering Methodology) – Formomis paremta objektiškai orientuota atvirkštinė inžinerija

VBA (Visual Basic for Application) – programavimo kalba, skirta praplėsti Microsoft Visio paketo galimybems.

Įvadas

Taikant sukurtas metodikas informacinei sistemai kurti, procesų eiga patogi sistemų analitikui. Daugumoje metodikų pirma atliekama reikalavimų inžinerija, kurios metu įsigilinama į probleminę sritį ir vartotojų reikalavimus. Tačiau taikant jas, informacinei sistemai keliamų reikalavimų specifikacijos eiga yra nenatūrali. Pavyzdžiui, duomenų modelio elementai į specifikaciją įtraukiami anksčiau, nei išvedamos informacijos vieneto (pavyzdžiui, ataskaitos) elementai, kurie motyvuoja duomenų modelio elementų atsiradimą specifikacijoje.

Taikant tradicinius sistemų kūrimo metodus, vartotojo pateikti reikalavimai užrašomi natūralia kalba ir specifikuojami formalizuotai. O perėjimas nuo dalinai struktūrizuotos specifikacijos prie formalizuotos yra paprastesnis, nei perėjimas nuo natūralios kalbos, nes atotrūkis tarp vartotojo ir analitiko yra mažesnis. Egzistuojantys automatizuoti dokumentų formų analizės metodai parodė kaip dalinai struktūrizuota informacija dokumentų formų pavyzdžių pavidale gali būti panaudota kuriamos sistemos koncepcinei schemai išgauti. Tačiau šie metodai nėra plačiai naudojami, nes paruošiamieji darbai atima daug laiko, o visiškos garantijos, jog gauta specifikacija yra teisinga, nėra. Tradiciniai sistemų kūrimo metodai siūlo analizuoti dokumentų formas reikalavimams kompiuterizuotos informacijos sistemos statikai išgauti, tačiau formalizuotų priemonių šiam tikslui jie neturi.

Informacijos sistemos katedros siūlomas funkcinių reikalavimų specifikavimo metodo [3, 4, 5] vartotojo reikalavimų specifikavimo procesas yra artimas natūraliai reikalavimų analizės eigai. Šis procesas prasideda nuo kuriamos kompiuterizuotos informacijos sistemos konteksto apibrėžimo ir reikalavimų įvedamos ir išvedamos informacijos specifikavimo.

Pirmame skyriuje apžvelgtos metodikos vartotojo reikalavimams specifikuoti: visos metodikos pateiktos moksliniuose veikaluose, dalis jų turi eksperimentines realizacijas, o kai kurios ir komercinius CASE paketus. Išanalizuotas dokumentų formų panaudojimas reikalavimų inžinerijoje.

Antrame skyriuje pateiktas vartotojų įvedamos ir išvedamos informacijos specifikavimo analizės modelis, pagrįstas dokumentų formų analize. Aprašytas procesas skirtas vartotojo reikalavimams specifikuoti.

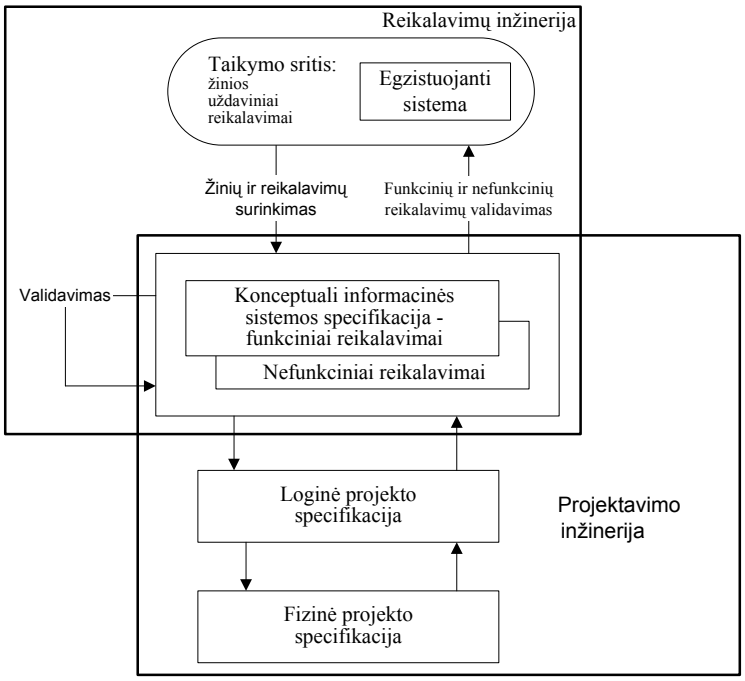
Trečiame skyriuje pateiktas analizuotos metodikos sistemos modelis. Pateikta metaduomenų saugykla, skirta įvedamai ir išvedamai informacijos struktūrai saugoti.

Ketvirtame skyriuje pateiktas realizuotas įvedamos ir išvedamos informacijos specifikavimo įrankio prototipas. Aprašyta prototipo struktūra ir funkcijos. Prototipas realizuotas Microsoft Visio paketu, o metaduomenų saugykla pasirinkta Microsoft Access duomenų bazė.

1 Vartotojo reikalavimų specifikavimo analizė

1.1 Vartotojo reikalavimų specifikavimo proceso aktualumas

Informacijos sistema – tai sistema, kuri surenka, apdoroja, saugo ir platina informaciją, padedančią priimti sprendimus, koordinuoti ir kontroliuoti organizacijos veiklą, analizuoti problemas, vizualizuoti sudėtingus objektus, kurti naujus produktus. Informacijos sistemos schema ir jos gyvavimo ciklo etapai pavaizduoti 1 paveiksle. Informacijos sistemoje galima išskirti du gyvavimo ciklo etapus: reikalavimų inžinerijos ir sistemos inžinerijos [11].



1 pav. Informacijos sistemos schema

Reikalavimų sistemos etape vartotojas pateikia reikalavimus įvedamai ir išvedamai informacijai. Įvedamos informacijos struktūra priklauso nuo to, kokią informacijos struktūrą vartotojas pageidauja gauti sistemos išeigoje. Vartotojas pateikia reikalavimus natūralia kalba, formomis, naudojamomis informacijos sistemomis.

Sukurtos ir naudojamos metodikos vartotojo reikalavimams specifikuoti skirtos analitikui. Sukurti modeliai nėra suprantami vartotojui. Tuomet kyla klausimas: ar teisingai analitikas suspecifikavimo vartotojo keliamus reikalavimus? Šitokiu atveju analitikas turi būti kompetentingas tiriamoje biznio veiklos srityje.

1.2 Vartotojo reikalavimų specifikavimo analizės metodų, priemonių parinkimas

Pirmasis etapas analizuojant organizacijos veiklą yra vartotojo reikalavimų išgavimas. Vartotojas dažniausiai reikalavimus pateikia natūralia kalba ir nestruktūrizuotus, o analitikas turi specifikuoti pateiktus reikalavimus.

Moksliniuose straipsniuose ir eksperimentinėse programose sprendžiama problema, kaip sumažinti atstumą tarp analitiko ir vartotojo. Vienuose straipsniuose analitikas artinamas prie vartotojo ir tiriamos biznio srities, kituose – vartotojas artinamas prie analitiko. Pirmuose straipsniuose pateiktomis metodikomis paremta dauguma komercinių CASE paketų. Antrųjų straipsnių metodikos dar vystomos ir sukurti tik eksperimentiniai projektavimo paketai. Daugumoje jų duomenų įvedimui naudojamos formos, nes jos yra paplitusios bendravimui tarp organizacijų.

Naudodamas sukurtus komercinius produktus analitikas turi galimybę specifikuoti įmonės veiklą ir vartotojo keliamus reikalavimus. Plačiau naudojami paketai yra Oracle Developer Suite [16], Provision Modeling Suite [17], Rational Suite [18], Magic Draw [19], Microsoft Visio [20]. Kiekvienas paketas yra paremtas tam tikra metodika ar net keliomis.

1.3 Vartotojų reikalavimų specifikavimo analizės rezultatai

Norint išrinkti kuo geresnį vartotojo informacijos specifikavimo konceptualizavimui tinkamą sprendimą, atlikta esamų metodikų ir CASE paketų analizė.

Dauguma paketų paremti metodikomis, kuriose analitikas, o ne vartotojas, artinamas prie tiriamos biznio veiklos, kad kuo tiksliau specifikuotų pateikiamus reikalavimus. Kai kuriose kompanijose, turinčiose komercinį CASE paketą, tebedirba metodikų autoriai, kaip Barker [1], Booch [2], Rumbaugh [13] ir kiti.

Vis dar ieškoma patogesnės ir artimesnės vartotojui specifikavimo metodikos, kurioje vartotojas galėtų dalyvauti betarpiškai. Todėl tebėra aktuali problema kuo teisingiau ir tiksliau specifikuoti vartotojo poreikius, kuo daugiau įtraukiant patį vartotoją, nes kuriama sistema turi tenkinti vartotojo poreikius, o ne atvirkščiai. Didėjant verslo procesų sudėtingumui, analitikui vis sudėtingiau juos gerai žinoti ir įsijausti į vartotojo rolę.

1.3.1 Vartotojo reikalavimų specifikavimo srities analizė

Vartotojų poreikių specifikavimas yra informacijos sistemos kūrimo pirmasis etapas. Dauguma sukurtų procesų ir CASE paketų vartotojo poreikius specifikuoja grafiniu pavidalu.

Aptarsiu keleto sukurtų CASE paketų vartotojo reikalavimų specifikavimą.

1.3.1.1 R. Barker vartotojo reikalavimų specifikavimo metodika

R. Barker siūloma metodika [1] realizuota ORACLE Developer Suite [16] pakete. Analizuojant kuriamą informacinę sistemą yra sudaromi šie sistemos modeliai:

- ✓ esybių-ryšių modelyje specifikuojama kuriamos IS duomenų schema. Joje suspecifikuojama įvedama ir išvedama informacija, nurodomi tarpusavio ryšiai.
- ✓ funkcijų hierarchijos modelyje sudedamos kuriamos IS atliekamos funkcijos ir jų hierarchija.
- ✓ procesų modelyje kuriamos IS funkcijos priskiriamos aktoriams, būsimiems IS vartotojams arba išorės organizacijoms, ir nurodomi duomenų srautai tarp funkcijų.
- ✓ duomenų srautų modelyje pateikiamas duomenų judėjimas tarp esybių, procesų ir duomenų saugyklų.

Remiantis esybių – ryšių modeliu yra sugeneruojama duomenų bazės schema, iš funkcijų hierarchijos sugeneruojama kuriamos IS meniu, o kiekviena duomenų srautų diagrama atitinka formą arba ataskaitą.

1.3.1.2 RUP metodika

G. Booch [2] pasiūlyta metodika realizuota Rational Suite [18] pakete. Analizuojant kuriamą informacinę sistemą yra sudaromi šie modeliai:

- ✓ klasių diagrama – struktūrinė diagrama, parodanti klasių, sąsajų, bendradarbiavimų ir jų ryšių visumą. Klasės gali vaizduoti informaciją, produktus, dokumentus ar organizacijas,
- ✓ objektų diagrama - struktūrinė diagrama, parodanti objektų ir jų ryšių visumą. Diagrama atvaizduoja generalizaciją klasių diagramos specifinėms situacijoms,
- ✓ USE CASE diagrama - sistemos elgsenos diagrama, parodanti panaudojimo atvejus, dalyvių ir jų ryšių visumą,
- ✓ sekų diagrama - elgsenos diagrama, parodanti sąveikas, pabrėžianti žinučių tvarkymo laiką,
- ✓ bendradarbiavimo diagrama - elgsenos diagrama, parodanti sąveikas, pabrėžianti struktūrizuotą objektų organizavimą ir žinučių siuntimą,
- ✓ būsenų diagrama - elgsenos diagrama, parodanti galimas klasės (objekto) būsenas. Objektų ir sistemų gyvavimo ciklas parodo būsenas, kurios gali būti klasė (objektas) per visą gyvavimo laiką,

- ✓ veiklos diagrama - elgsenos diagrama, parodanti veiklos tėkmę – priežastinį ryšį nuo veiklos į veiklą. Veiklos diagramos gali būti naudojamos biznio procesų ir programinės įrangos modeliavimui,
- ✓ komponentų diagrama - struktūrinė diagrama, parodanti programinės įrangos komponentų ir jų ryšių visumą,
- ✓ įrangų diagrama - struktūrinė diagrama, parodanti programinės įrangos ir techninės įrangos ryšių visumą.

Remiantis sudarytais modeliais, yra sugeneruojami duomenų bazių ir klasių modeliai. RUP metodu išanalizuojama tiek statinė, tiek dinaminė įmonės veikla.

1.3.1.3 Rummler siūloma vartotojo reikalavimų specifikavimo metodika

Rummler-Brache pasiūlyta metodika realizuota „Proforma“ korporacijos pakete „Provision Workbench“ [17]. Analizuojant biznio veiklą sudaromi šie modeliai:

- ✓ tikslų modelis – vaizduoja biznio veiklos tikslus. Organizacijos tikslai pateikiami hierarchinėje struktūroje, kur aukštesnis lygis detalizuojamas detalesniais tikslais. Todėl norint pasiekti visus tikslus, reikia pasiekti visus žemiausio lygio tikslus, kurie yra elementarūs,
- ✓ organizacijos modelis – pateikia organizacijos hierarchinę struktūrą,
- ✓ vietovių modelis – parodomos vietovės, kurios atitinka veiklos aplinką. Hierarchiškai išdėstomos detalesnės vietovės priklausančios pagrindinei aplinkai,
- ✓ procesų modelis – sudaroma biznio procesų hierarchinė struktūra iš biznio sąveikų ir darbų sekos modelių,
- ✓ sistemos modelis - atvaizduojama hierarchinė sistemų, kurios yra atviros bizniui, struktūra. Sistemos objektai gali būti naudojami darbų sekos modelyje,
- ✓ įvykių modelis – išdėstyta įvykių hierarchija, kuri rodo vykstančius, atsirandančius įvykius biznio sferoje. Įvykis šiame modelyje įpareigoja biznio sferą inicijuoti tam tikrą funkciją. Visi įvykiai apjungti į hierarchiją,
- ✓ biznio sąveikų modelis – atvaizduojama vartotojo biznio veikla strateginėje perspektyvoje, parodoma sąveika tarp vidinių organizacijos objektų ir išorinių organizacijų. Modelis orientuotas ne į organizacinių vienetų apibrėžimą, bet į ryšius bei informacinius ir materialius persiuntimus tarp organizacijų,
- ✓ darbų sekos modelis – atvaizduojami biznio procesai, išreiškiant juos komponentais ir darbų seka tarp tų veiklų. Modelis koncentruotas į darbų seką

nuo biznio pradžios iki galo. Procesui arba darbui gali būti kuriamas detalesnio lygio darbų sekos modelis,

- ✓ panaudojimo atvejų modelis – apibrėžiamos ir analizuojamos svarbesnės biznio sferos, biznio procesų ar veiklos sąveikos tiek organizacijos viduje tiek išorėje. Padeda apžvelgti kaip dauguma biznio zonų yra susijusios su išorinėmis biznio esybėmis veiklomis ir sistemomis,
- ✓ objektų modelis – apibrėžiama detali biznio objektų informacija(savybės): atributai, metodai, funkcijos, būsenos. Objektų modelis sudaro pagrindinį biznio sferos objektų komponentų sąrašą,
- ✓ subtipų modelis – apibrėžiami objektų tipai, kurie gali būti specializacijos arba apibendrinimai priklausomai nuo objekto tipo,
- ✓ būsenų modelis – pateikiamas objektų tipų gyvavimo ciklas. Modelyje būsena gali būti sudėtinė iš kelių jos subbūsenų,
- ✓ sąveikų modelis – atvaizduojami pranešimai, kurie siunčiami tarp biznio objektų. Modelis parodo įvykių ir pranešimų eilės tvarką,
- ✓ metodų modelis – detalizuojamos biznio sferos elgsenos iki įvykių.

Sudarant šiuos modelius išanalizuojama įmonės veikla, joje vykstantys procesai, įmonės tikslai.

Kai kurios formomis paremtos metodikos turi eksperimentines realizacijas arba visai jų neturi. Formos yra plačiai naudojamos bendravimui tarp organizacijų. Plačiausiai naudojami duomenų fragmentai dažniausiai yra surenkami formose.

1.3.1.4 Formomis paremta objektiškai orientuota atvirkštinės inžinerijos metodika

Formomis paremta objektiškai orientuota atvirkštinės inžinerijos (Form Driven Object-Oriented Reverse Engineering Methodology (FORE)) metodika [10] naudoja formas liktinių programų semantikai gauti.

Formos yra plačiausiai naudojami oficialūs bendravimo dokumentai. CODASYL vartotojo palankių sąlygų komiteto pripažino formų svarbumą ir rekomenduoja formomis paremtą požiūrį naudoti kaip pagrindinį, kuriant vartotojo sąsają.

FORE metodas naudoja elektronines formas kaip duomenų šaltinį. Šioje metodikoje nėra įvertinama išvedama informacija – ataskaitos, todėl išvedamai informacijai gali trūkti duomenų.

FORE metodas susideda iš penkių fazių: formos elgesio analizės, formos objektų skaidymo, objektų struktūros modeliavimo, scenarijaus projektavimo ir modelio integracijos.

Formos elgesio analizės etape reikalinga formos struktūra ir vartotojo įsiterpimas. Programa-agentas surenka visą informaciją apie formos struktūrą ir vartotojo sąveiką, tokia kaip įvykiai, operacijos ir formos laukų reikšmės.

1.3.1.5 Veiklos procesų atvirkštinė inžinerija: formomis paremtas požiūris

Įmonių procesų atvirkštinės inžinerijos (Enterprise process reverse engineering (EPRE)) metodika[9] buvo pasiūlyta K.-H. Kim ir Y.-G. Kim [9]. Šis metodas paremtas biznio formų analize. Metodo principas - padidinti sąveiką tarp verslo srities projektų vadovų, tiriamos informacinės sistemos personalo ir galutinio vartotojo analizės ir pakartotinio projektavimo fazėse.

Informacinių sistemų srityje atvirkštinė inžinerija yra projektavimo elementų, tokių kaip programos struktūra ar duomenų schema, ištraukimas iš egzistuojančios sistemos. Tačiau atvirkštinė inžinerija orientuota ištraukti informaciją kuo efektyviau ir teisingiau. Biznio formos, duomenų bazės, duomenų modeliai ir programos kodas yra dažniausiai naudojami šaltiniai.

Formos paprastai yra sukurtos, kad būtų standartizuotos biznio veiklos ir palengvintas jų tarpusavio bendravimas. Tokiu būdu jos teikia daug informacijos apie biznio užduotis, sąlygas ir darbų sekas. Ši charakteristika leidžia formas laikyti kaip šaltinį, generuojant procesų modelį. EPRE metode buvo pasirinktas Choobineh ir kitų pasiūlytas formų apibrėžimas, kad forma yra kintamųjų struktūriška aibė (t.y. formų laukai), kurie tinkamai formuoti duomenų įvedimui ir vaizdavimui. Šis apibrėžimas glaustai susieja tikslą, funkciją ir komponentus, ir tinkamas tiek elektroninėms tiek spausdintoms formoms.

EPRE proceso metodas susideda iš trijų stadijų: formos analizės, proceso modelio formavimo ir procesų pakartotinio projektavimo.

Šiuo metu sukurtoje sistemoje yra keletas trūkumų Pirma, įtraukiant fizines veiklas ir šakojimus, EPRE metodui reikia truputį žmogaus įsikišimo, kad sugeneruotų procesų modelį. Antra, procesų pakartotinis projektavimas pritaikytas tik pirkimo-pardavimo veikloms. Todėl fizinės veiklos ir galimybės susietos su pakartotiniu panaudojimu turi būti susietos su vartotoju. Trečia, pasiūlyta laukų aibės operacijų pakartotinio projektavimo darbų linija nesilaiko pagrindinių pakartotinio projektavimo pasirinkimų, tokių kaip esamos esybės pašalinimas. Pašalinimo vietoje siūlomas sudėtingas, reikšmingas, tęstinas patobulinimas. Ketvirta, dėl identifikuojamų laukų priklausomybių ir tarpusavio ryšių apribojimo, pasiūlytas laukų aibės operacijų pakartotinis projektavimas nutrūksta laukų lygyje, nustatant lauko tipą.

1.3.1.6 Vartotojo reikalavimų specifikavimas, paremtas formų analize

J. Choobineh, M.V. Mannino, J.F. Nunamaker, B. R. Konsynski, V. P. Tseng ir B. Wangler [7, 8, 12, 15] siūlo analizuoti formas ir automatizuotu būdu jas panaudoti koncepcinės schemos kūrimui.

Šios metodikos esmė tai, kad specifikuojant vartotojo reikalavimus sukuriamas duomenų bazių modelis. Remiantis faktu, kad konceptuali schema gaunama iš formalių aprašymų (formų), kurios yra naudojamos kiekvienos įmonės veikloje, o ne pokalbio metu, sumažinamas loginių klaidų kiekis.

Prototipą sudaro dvi sistemos:

- ✓ formų apibrėžimo sistema
- ✓ konceptualaus modelio sistema.

Ši metodika surenka konceptualų modelį modeliavimo kalbos MOLOC aprašu, kuris yra semantiškai turtingesnis, nei klasikinis esybių-ryšių modelis. MOLOC modelyje konceptualus modelis yra apibrėžiamas kaip PROLOG faktų aibė.

Į konceptualų modelį įtraukiami esybių tipai ir subtipai, ryšių tipai ir subtipai, įvykių tipai (išoriniai ir vidiniai), atributai, leksikos tipai ir įvairūs integralumo apribojimai.

1.3.2 Vartotojo reikalavimų specifikavimo metodikų lyginamoji analizė

1 lentelėje pateikiamos sukurtos metodikos, jų duomenų šaltiniai ir formuojami modeliai. Matome, kad daugelio metodikų duomenų šaltiniais laikomos formos. Taikant tradicinius sistemų kūrimo metodus, vartotojo pateikti reikalavimai užrašomi natūralia kalba ir specifikuojami formalizuotai.

1 lentelė

Metodikos ir jų pagrindinės savybės

Duomenų šaltiniai	***Sistemos vaizdas***	***Kuriamas modelis***	***Pagrindinės charakteristikos***	***Metodo pavadinimas***	***Autoriai***
Formos	Duomenys	ER modelis	Formos modelio pateikimas Ekspertinių DB projektavimo sistemų kūrimas	Choobineh	Choobineh ir kiti [7, 8]
			Verslo dokumentų formų naudojimas Pokalbis su vartotoju	Kim	Kim ir kiti [9]
			EER naudojimas terminų žodynui	Batini	Batini ir kiti
	Duomenys	Formų specifikacija, paremta FORMAL	Labai aukšto lygio galimybės (facility) teikimas su didelėmis galimybėmis kompiuterizuoti įvairiausius duomenis apdorojančias veiklas Veiklos procedūrų specifikavimas Procesų modelis ir	Shu	Shu ir kiti

Duomenų šaltiniai	*Sistemos vaizdas*	*Kuriamas modelis*	*Pagrindinės charakteristikos*	*Metodo pavadinimas*	*Autoriai*
			specifikavimas		
	Biznio procesai	EPC modelis	BPR procesų modeliavimo metodo pateikimas: Palaiko tik informaciją apdorojančias veiklas	EPRE	Kim ir kiti
Natūrali kalba	Duomenys	ER modelis	Pirmoji konceptualaus duomenų modeliavimo metodika	Chen	Chen
		EDFD	ERD ir DSD sujungimas	Bailin	Bailin
	Objektai	Objektų, dinaminis, funkcinis modeliai	Paremta leksinės analizės metodu	Booch	Booch
Esamų sistemų šaltiniai	Duomenys ir procesai	Duomenų srautų modelis	Struktūrinio projektavimo metodika	SSADM	Yourdon
		Duomenų, procesų modeliai, duomenų / procesų matrica	Programinės įrangos projektavimo metodika: Įtraukia ISP į struktūrinį metodą	Informacijos inžinerija	Martin
	Objektai	NER ir UPM	Pateikia eksperimentinį objektų modelį	MOODD	Silva
		Objektų modelis	Formalus OO analizės ir projektavimo modelis	OMT UML	Rumbaugh ir kiti [13] Quatrani

1.3.3 Vartotojo reikalavimų specifikavimo metodikos kokybės kriterijų apibrėžimas

Sukurtas prototipas bus naudojamas vartotojo reikalavimų įvedamai ir išvedamai informacijai specifikuoti. Atliekant eksperimentą, siekiami tikslai:

- ✓ atstumo tarp analitiko ir vartotojo sumažinimas,
- ✓ įvedamos informacijos pagrindimas išvedamąja,
- ✓ sumažinti klaidų skaičių specifikuojant vartotojo reikalavimus,
- ✓ duomenų saugojimas SQL metaduomenų bazėje,
- ✓ kuriant duomenų modelį atsižvelgti į išvedamos informacijos reikalavimus.

Sukurtas prototipas turėtų būti funkcinių reikalavimų specifikavimo metodika paremto prototipo dalis, naudojama vartotojo reikalavimams surinkti.

Realizuotas įvedamos ir išvedamos informacijos specifikavimo prototipas turėtų turėti :

- ✓ mažesnį atstumą tarp analitiko ir vartotojo, labiau priartinant vartotoją prie analitiko, nei naudojamos komercinės metodikos,
- ✓ analitiko darbo nepadidinimas arba santykinai nedidelis darbo laiko sugaišimas analizuojant reikalavimus lyginant su esamomis metodikomis.

1.4 Vartotojų reikalavimų specifikavimo sistemos varianto parinkimas

Kuriant prototipą yra dvi alternatyvos:

- ✓ sukurti naują prototipą, realizuojant vartotojo grafinę sąsają ir reikiamas funkcijas,
- ✓ pasirinkti esamą CASE įrankį ir praplėsti jo galimybes, realizuojant reikiamas funkcijas.

Šiame darbe pasirinkta praplėsti esamą įrankį. Praplečiant jau sukurta CASE įrankį papildomu funkcionalumu:

- ✓ reikės mažiau laiko resursų, nes nereikės sukurti vartotojo sąsajos, grafinės aplinkos,
- ✓ sukurta įrankio dalis yra ištestuota ir be klaidų. Reikės testuoti tik sukurtą įrankį. Tokiu būdu sumažėja klaidų tikimybė ir skaičius,
- ✓ yra tikimybė, kad bus galima panaudoti kai kurias CASE įrankio funkcijas. Tokiu atveju nereikės kurti kai kurių CASE įrankio funkcijų, be to jos veikia teisingai ir yra pilnai ištestuotos,
- ✓ tikimybė, kad bus nutrauktas komercinio CASE įrankio realizavimas, yra labai maža.

Tačiau praplečiant jau esamą CASE įrankį, galima susidurti su problemomis:

- ✓ CASE įrankis turi savo saugyklą ir prie jos prieiti negalima. Paprastesni CASE įrankiai turi savo metaduomenų saugyklą failinėje struktūroje ir ji žinoma tik paketo kūrėjams. Tokiu atveju būtina realizuota lygiagrečią metaduomenų saugyklą ir duomenis saugoti dviejose saugyklose,
- ✓ dauguma CASE įrankių nėra praplečiami. Šiai dienai vieni didesnių paketų, turintys galimybių praplėtimo savybę, yra Microsoft Visio ir Rational XDE,
- ✓ programavimo priemonės yra ribotų galimybių. Galimybė praplėsti CASE įrankio savybes sumažina programavimo kalbos lankstumą, kadangi būtina naudoti paketo palaikomą programavimo kalbą, dažnai ji būna apribota, būtina naudoti duomenų ir objektų struktūrą..

Prototipo kūrimui nuspręsta pasirinkti Visio 2003 paketą. Jis leidžia praplėsti įrankį naujomis funkcijomis, tačiau neleidžia prieiti prie savo saugyklos. Todėl darbe naudosime savo saugyklą.

Specifikuojant vartotojo reikalavimus viena aktualių temų yra esybių-ryšių vaizdavimas. Juos galima vaizduoti kaip siūlo ORACLE CASE.

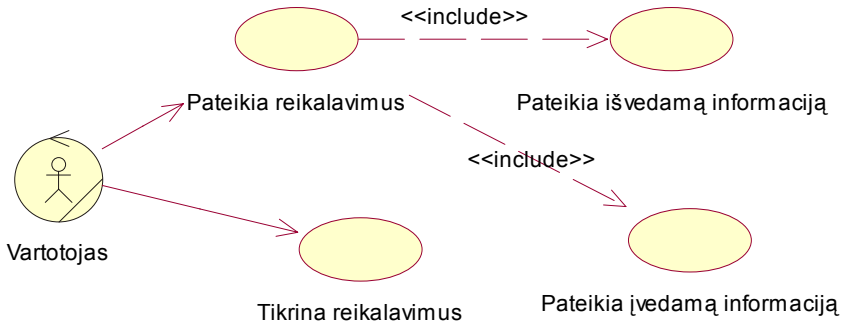
Oracle siūlo esybių – ryšių diagramą vaizduoti, taip kad būtų patogiau analitikui. Ji siūlo esybes, kurios turi daugiausia ryšių vaizduoti dešinėje ir viršuje. Pirma dedama esybė, turinti daugiausia ryšių dešinėje ir po to nuosekliai vaizduojamos susietos esybės kairėn (2 pav.).

Kitas būdas yra išdėstyti esybes ir ryšius taip, kaip jie ekvivalenčiai išdėstyti formoje. Tačiau tokiu būdu analitikui sudėtingiau pamatyti ER schemos teisingumą.

2 Apribojimai vartotojo reikalavimų specifikavimui

2.1 Vartotojo reikalavimų specifikavimo panaudojimo atvejų modeliai

Vartotojo atliekamos funkcijos pateiktos 3 paveiksle.



2 pav. Vartotojo panaudojimo atvejų diagrama

Aktoriai:

Vartotojas – pateikia informaciją specifikavimui ir dalyvauja suvedant reikalavimus į kompiuterį.

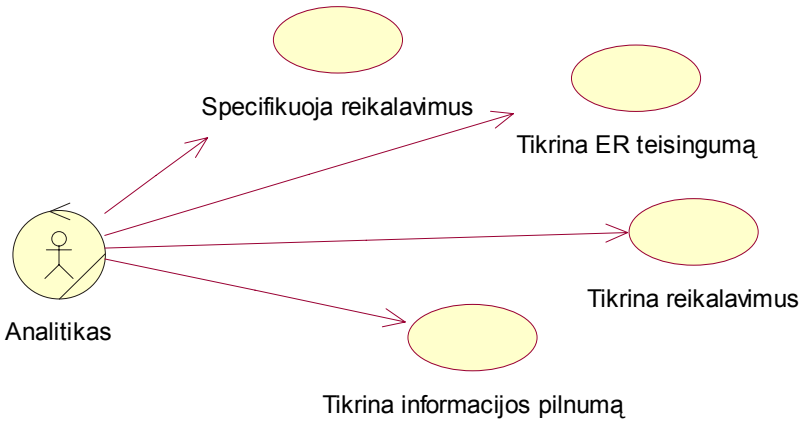
Panaudojimo atvejai:

Pateikia reikalavimus – įvedamos ir išvedamos informacijos pateikimas.

- *Pateikia išvedamą informaciją* – informacijos pateikimas, kurią nori matyti vartotojas kaip sukurtos KIS sistemos rezultatą.
- *Pateikia įvedamą informaciją* – informacijos pateikimas, kurią vartotojas nori arba privalo pateikti, kad sukurta KIS suformuotų norimą rezultatą.

Tikrina reikalavimus – reikalavimų patikrinimas, kompiuterizavus juos.

Analitiko atliekamos funkcijos pateiktos 4 paveiksle.



3 pav. Analitiko panaudojimo atvejų diagrama

Aktoriai:

Analitikas – specifikuoją pateiktą informaciją, generuoja koncepcinę schemą ir tikrina jos teisingumą.

Panaudojimo atvejai:

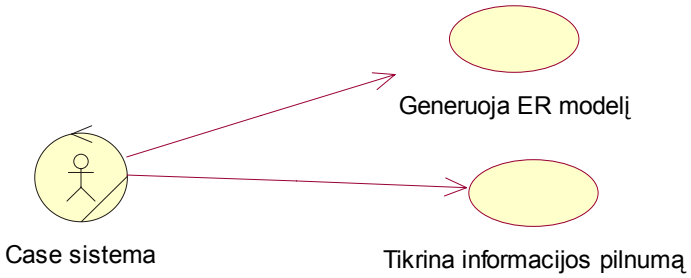
Specifikuoja reikalavimus– išskyrimas objektų, jų grupių iš vartotojo pateiktos informacijos.

Tikrina ER teisingumą – koncepcinės schemos tikrinimas, sugeneravus eksperimentiniam įrankiui esybių – ryšių modelį.

Tikrina reikalavimus – reikalavimų tikrinimas, išskiriant objektus ir jų grupes. Esant būtinybei – pareikalavimas tikslinti vartotojo pateiktus reikalavimus.

Tikrina informacijos pilnumą – tikrinimas, kad kiekvienas išeinančios informacijos elementas būtų pagrįstas įeinančiosios arba būtų išskaičiuojamos.

Eksperimentinės realizacijos atliekamos automatizuotos funkcijos pateiktos 5 paveiksle.



4 pav. Prototipo panaudojimo atvejų diagrama

Aktoriai:

CASE sistema – atlieka automatizuotus veiksmus – koncepcinės schemos generavimą.ir informacijos pilnumo tikrinimą

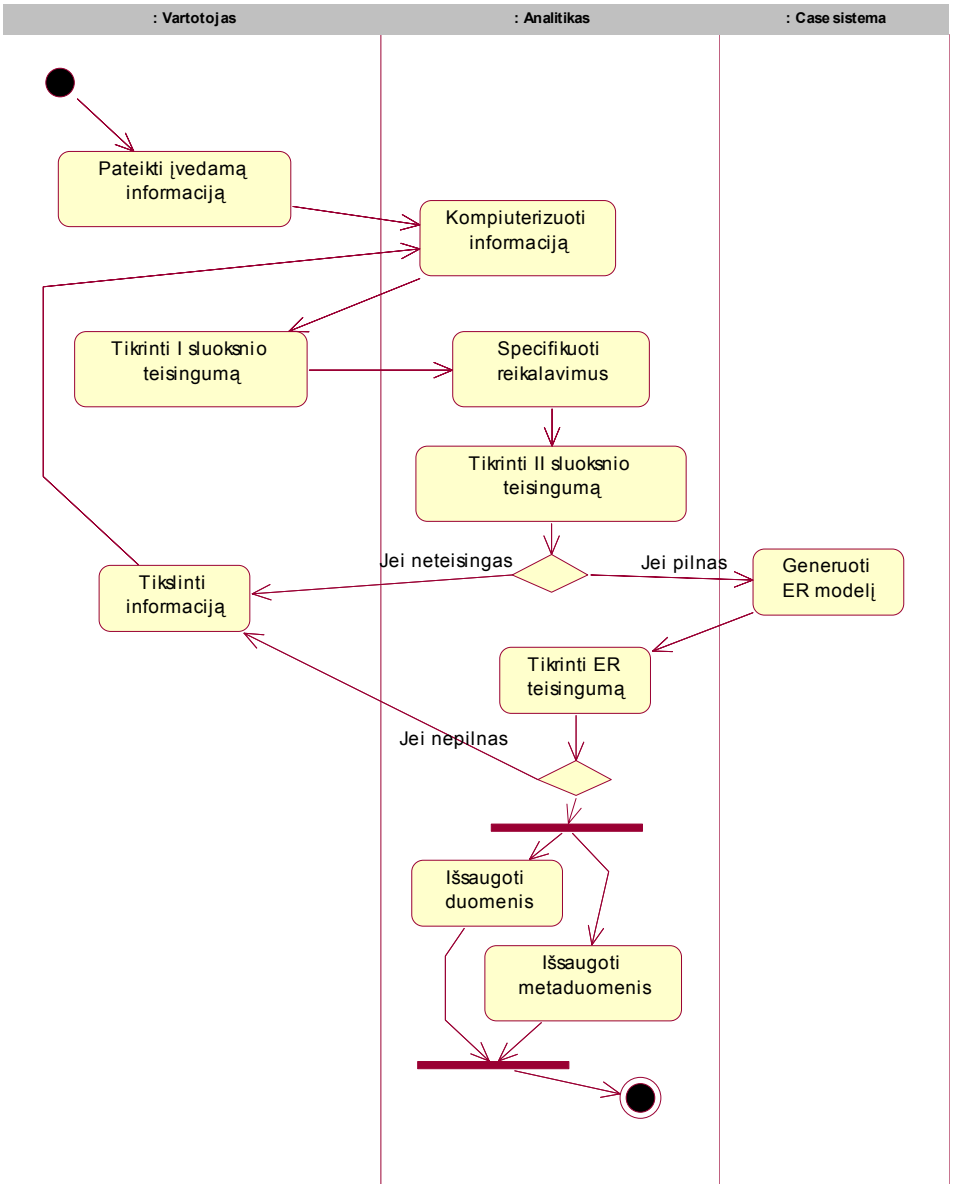
Panaudojimo atvejai:

Generuoja ER modelį – generuoja koncepcinę schemą iš analitiko išskirtų objektų ir jų grupių.

Tikrina informacijos pilnumą - tikrinimas, kad kiekvienas išeinančios informacijos elementas būtų pagrįstas įeinančiosios arba būtų išskaičiuojamos.

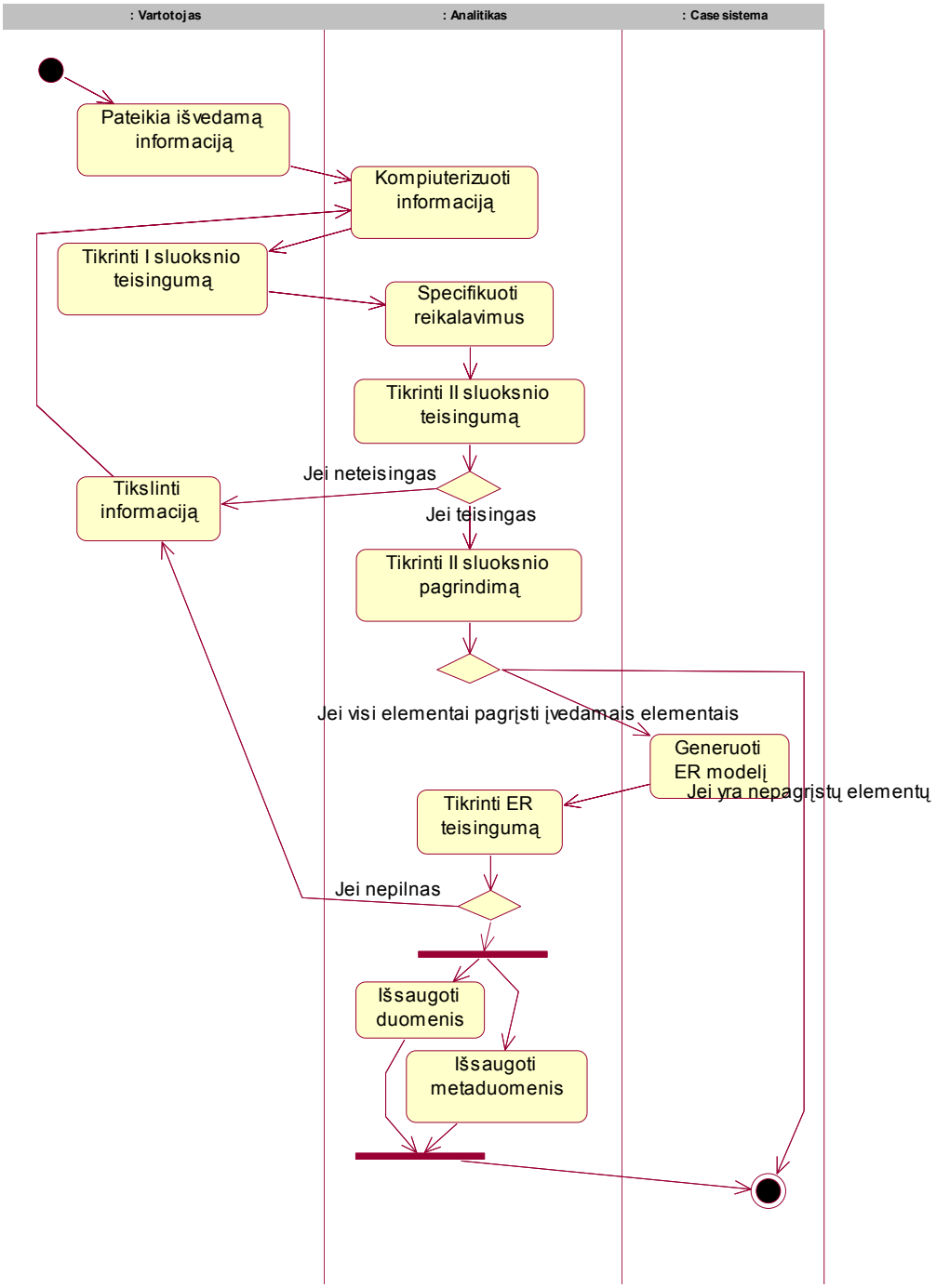
2.1.1 Vartotojo reikalavimų specifikavimo veiklos modeliai

Vartotojo įvedamos informacijos specifikavimas pateiktas 5 paveiksle.



5 pav. Įvedamos informacijos konceptualizavimo veiklos diagrama

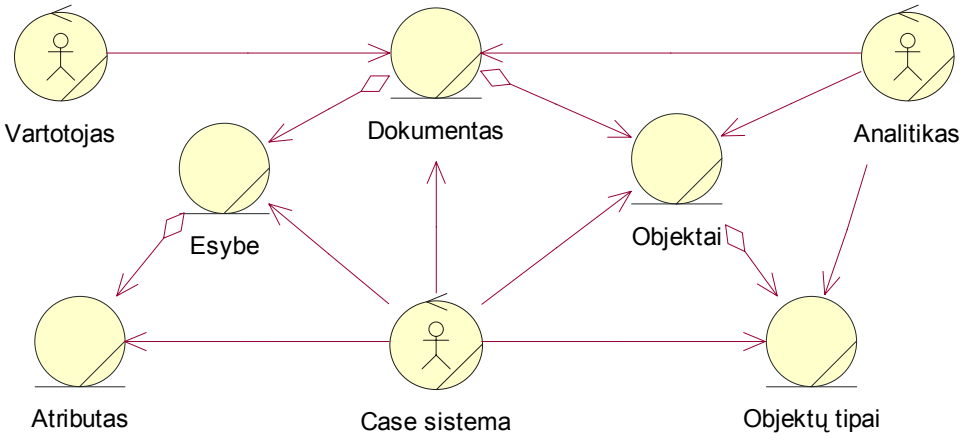
Išvedamos informacijos specifikavimas pateiktas 6 paveiksle.



6 pav. Išvedamos informacijos konceptualizavimo veiklos diagrama

2.1.2 Vartotojo reikalavimų specifikavimo klasių modelis

Dalykinės srities klasių diagrama pateikta 7 paveiksle.



7 pav. Dalykinės srities klasių diagrama

Esybės:

Dokumentas – vartotojo pateikti reikalavimai ir, jei reikia, struktūrizuoti.

Objektas – analitiko išskirtas elementas iš vartotojo reikalavimų, kaip svarbus objektas tolesniems KIS kūrimo etapams.

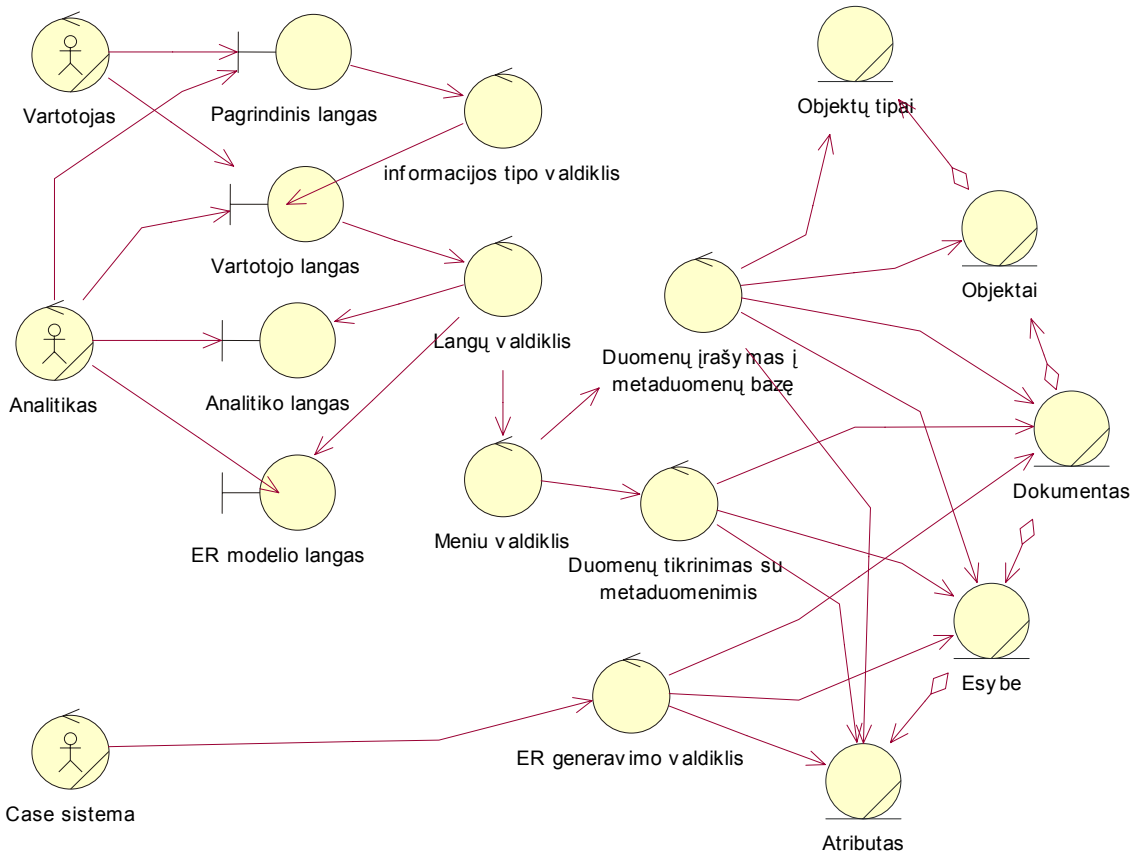
Objektų tipai – objekto tipų aibė, kuri nurodo objekto galimas savybes.

Esybė – eksperimentinės sistemos sugeneruotos esybės ir analitiko išskirtų ir nustatytų su tam tikrais požymiais objektai.

Atributas – eksperimentinės sistemos sugeneruoti atributai ir analitiko išskirtų ir nustatytų su tam tikrais požymiais objektai.

2.1.3 Vartotojo sąsajos modelis reikalavimų specifikavimui

Vartotojo navigavimo planas pateiktas 8 paveiksle.



8 pav. Vartotojo sąsajos modelis

Ribinės klasės:

Pagrindinis langas – esamų dokumentų pasirinkimo arba naujų dokumentų kūrimo langas.

Vartoto langas – vartotojo kompiuterizuotos informacijos vaizdavimo ir redagavimo langas

Analitiko langas – objektų išskyrimo ir jų tarpusavio ryšių nustatymo langas.

ER modelio langas – eksperimentinės sistemos sugeneruotos koncepcinės schemos vaizdavimo langas.

Meniu valdikliai:

Informacijos tipo valdiklis – pasirinkimas tarp įvedamos ir išvedamos informacijos specifikavimo.

Langų valdiklis – valdiklis, realizuojantis navigaciją tarp langų.

Meniu valdiklis – valdiklis, realizuojantis meniu.

ER generavimo valdiklis – esybių – ryšių modelio generavimo valdiklis. Jį pasirinkus, sistema generuos ER modelį.

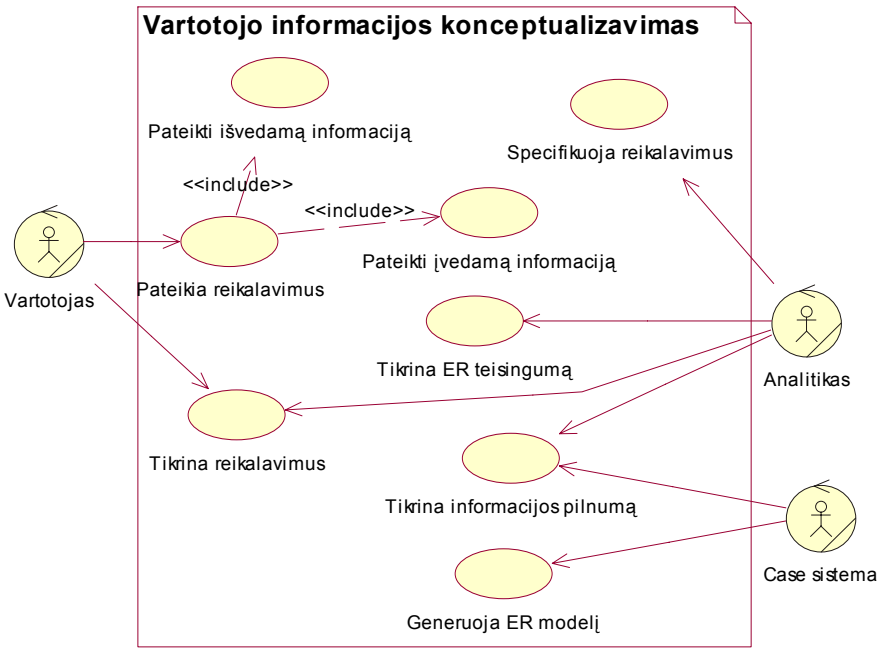
Duomenų įrašymas į metaduomenų bazę – duomenų įrašymo valdiklis. Pasirinkus duomenys bus įrašomi į metaduomenų bazę.

Duomenų tikrinimas su metaduomenimis – esamo dokumento tikrinimas su duomenimis metaduomenų bazėje ir klaidų pateikimas analitikui.

3 Vartotojo reikalavimų specifikavimo metodikos sistemos projektas

3.1 Vartotojo reikalavimų specifikavimo metodikos panaudojimo atvejų modelis

Sistemos panaudojimo atvejų diagrama pateikta 9 paveiksle.



9 pav. Sistemos panaudojimo atvejų diagrama

Aktoriai:

Vartotojas – pateikia informaciją specifikavimui ir dalyvauja suvedant reikalavimus į kompiuterį.

Analitikas – specifikuoją pateiktą informaciją, generuoja koncepcinę schemą ir tikrina jos teisingumą.

CASE sistema – atlieka automatizuotus veiksmus – koncepcinės schemos generavimą.ir informacijos pilnumo tikrinimą

Panaudojimo atvejai:

Specifikuoja reikalavimus– išskyrimas objektų, jų grupių iš vartotojo pateiktos informacijos.

Tikrinti ER teisingumą – koncepcinės schemos tikrinimas, sugeneravus eksperimentinei realizacijai esybių – ryšių modelį.

Tikrinti reikalavimus – reikalavimų tikrinimas, išskiriant objektus ir jų grupes. Esant būtinybei – pareikalavimas tikslinti vartotojo pateiktus reikalavimus.

Tikrinti informacijos pilnumą – tikrinimas, kad kiekvienas išeinančios informacijos elementas būtų pagrįstas įeinančiosios arba būtų išskaičiuojamos.

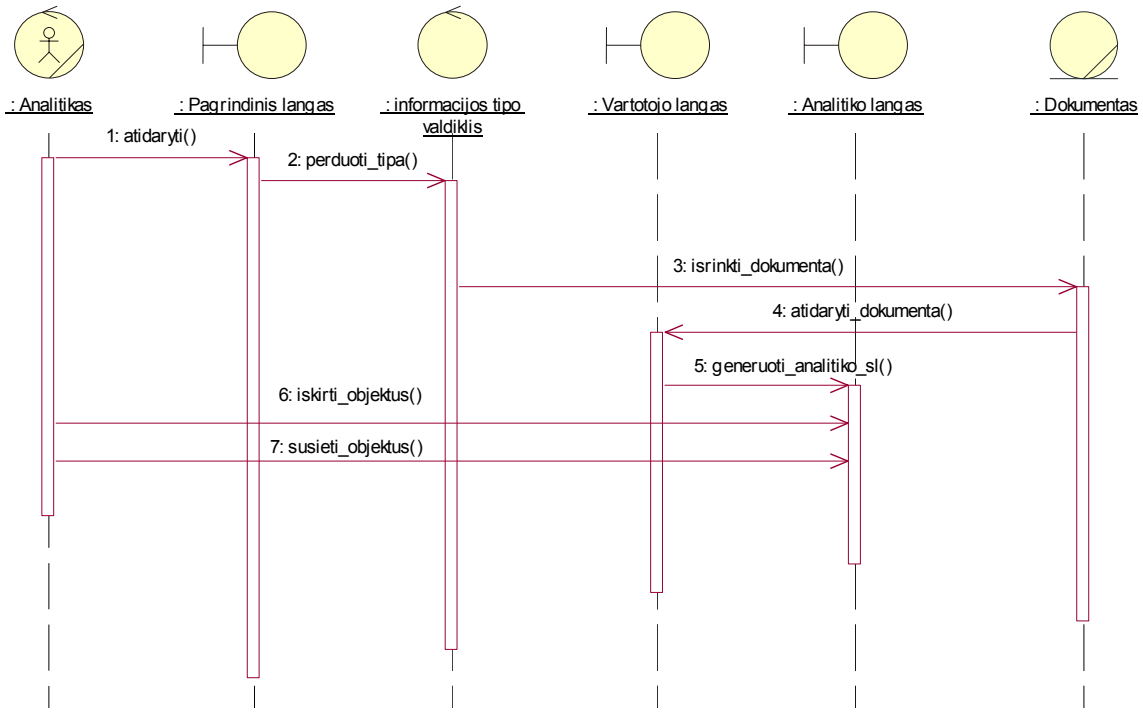
Pateikia reikalavimus – įvedamos ir išvedamos informacijos pateikimas.

- *Pateikia išvedamą informaciją* – informacijos pateikimas, kuria nori matyti vartotojas kaip sukurtos KIS sistemos rezultatą.
- *Pateikia įvedamą informaciją* – informacijos pateikimas, kurią vartotojas nori arba privalo pateikti, kad sukurta KIS suformuotų norimą rezultatą.

Generuoti ER modelį – generuoja koncepcinę schemą iš analitiko išskirtų objektų ir jų grupių.

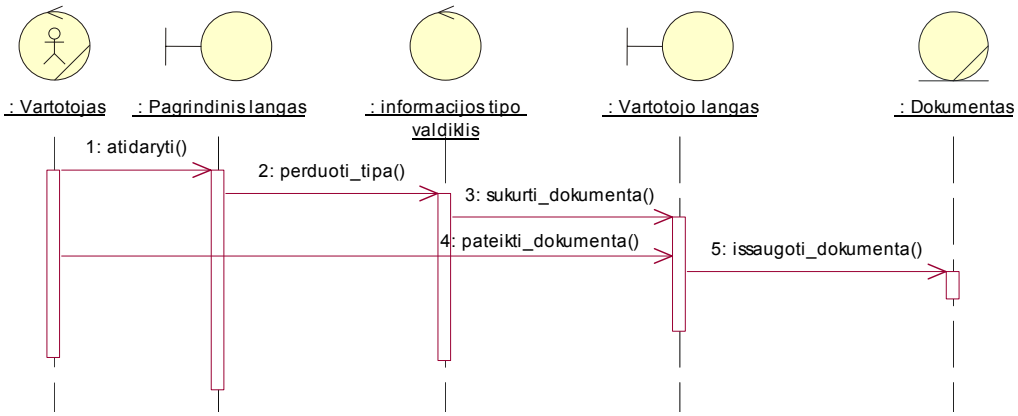
3.2 Vartotojo reikalavimų specifikavimo metodikos sekų modeliai

Panaudojimo atvejo „specifikuoti reikalavimus“ sekų diagrama pateikta 10 paveiksle.



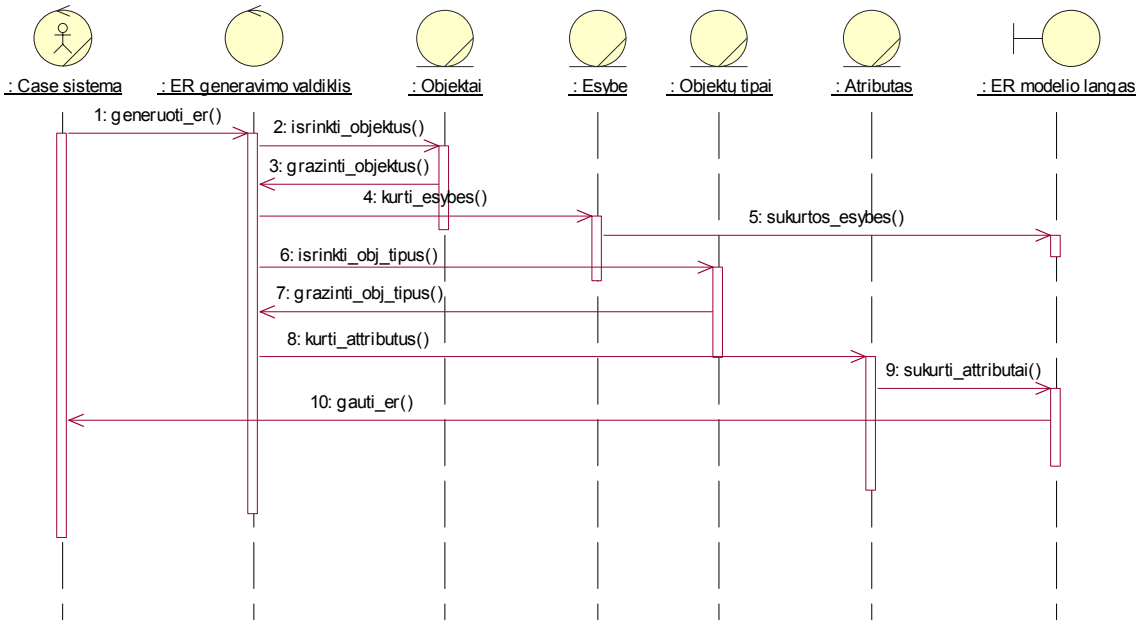
10 pav. Reikalavimų specifikavimo sekų diagrama

Panaudojimo atvejo „pateikti reikalavimus“ sekų diagrama pateikta 11 paveiksle.



11 pav. Reikalavimų pateikimo sekų diagrama

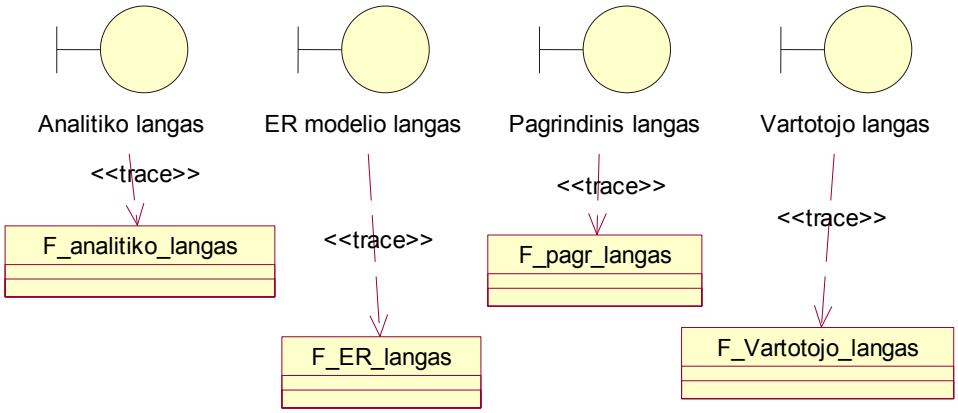
Panaudojimo atvejo „generuoti er“ sekų diagrama pateikta 12 paveiksle.



12 pav. Esybių – ryšių modelio generavimo sekų diagrama

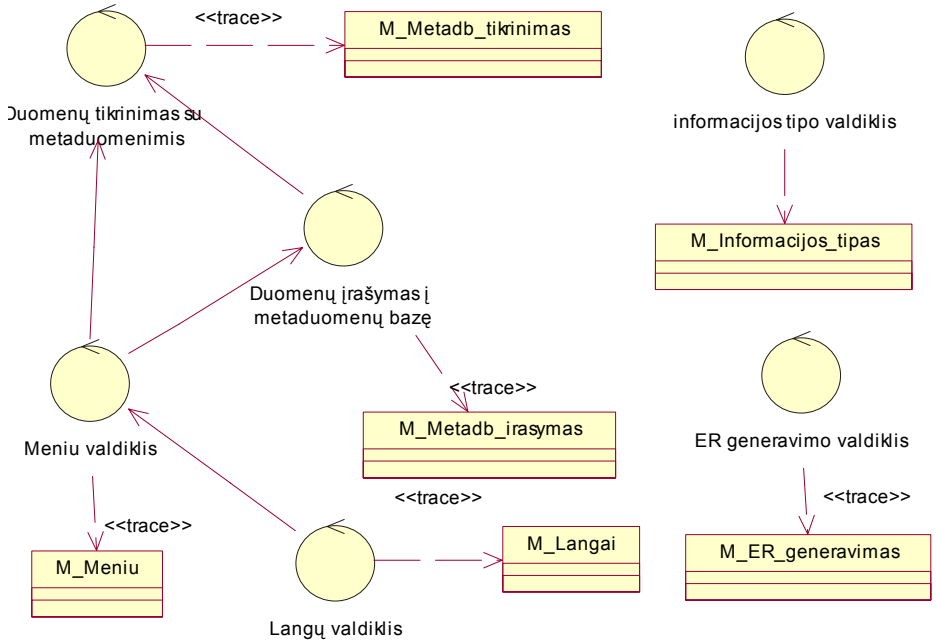
3.3 Vartotojo reikalavimų specifikavimo ir metodikos klasių modeliai

Ryšys tarp reikalavimų ir sistemos dokumentų formų klasių pavaizduotas 13 paveiksle.



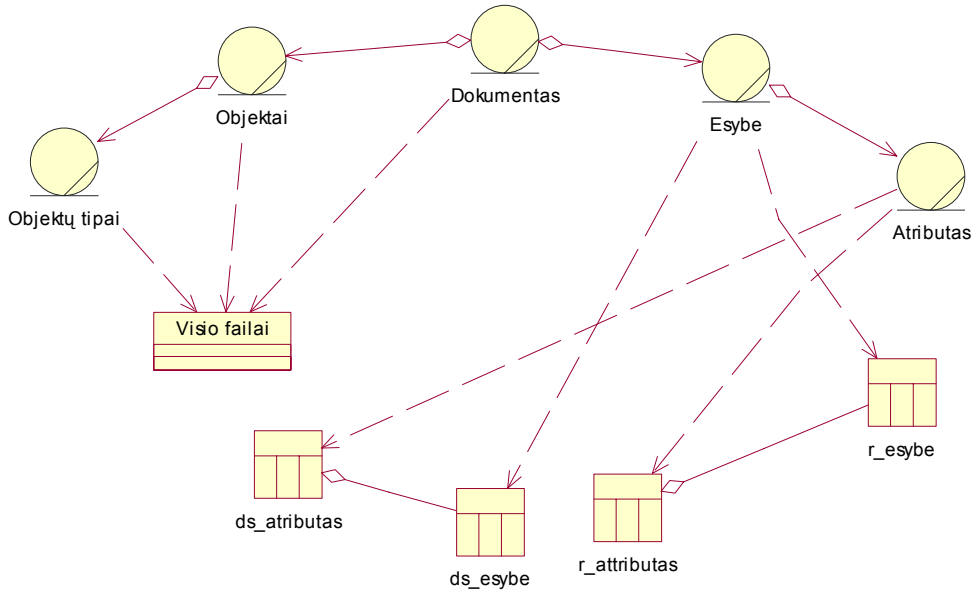
13 pav. Formų klasių trasų diagrama

Ryšys tarp reikalavimų ir sistemos dokumentų meniu klasių pavaizduotas 14 paveiksle.



14 pav. Meniu klasių trasų diagrama

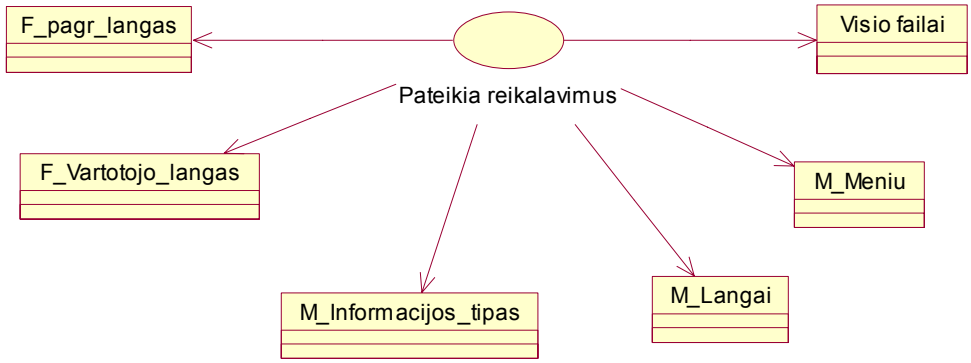
Ryšys tarp reikalavimų ir sistemos dalykinės srities klasių pavaizduotas 15 paveiksle.



15 pav. Esybių trasų diagrama

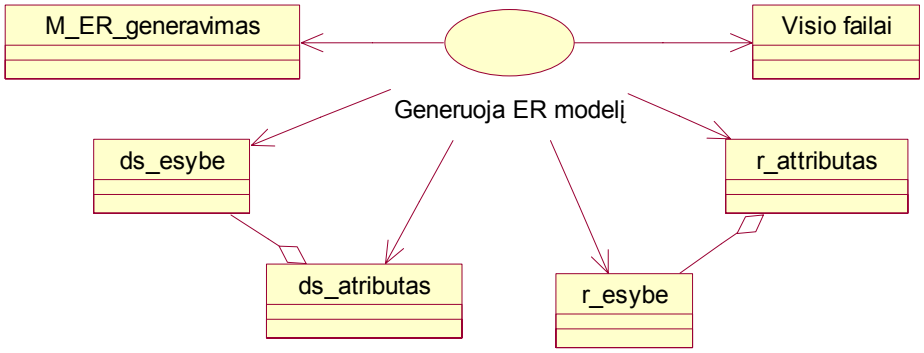
3.4 Vartotojo reikalavimų specifikavimo metodikos panaudojimo atvejų realizacijos modelis

Reikalavimų pateikimo realizacija pateikta 16 paveiksle.



16 pav. Reikalavimų pateikimo realizacija

Koncepcinės schemos generavimo realizacija pateikta 17 paveiksle.

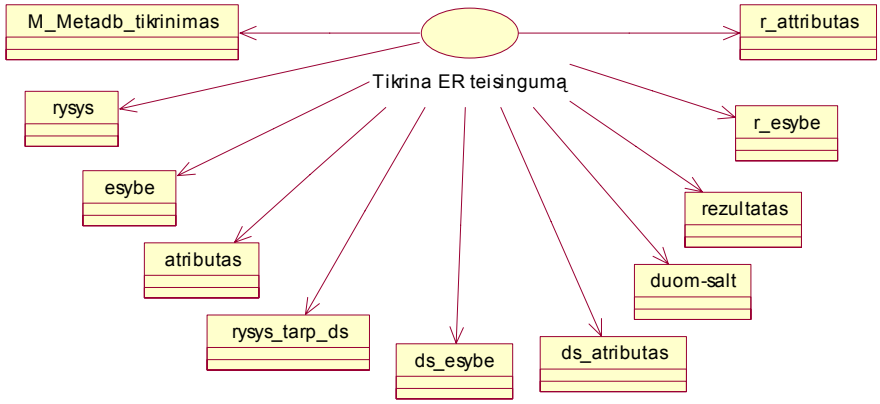


17 pav. Esybių – ryšių modelio generavimo realizacija

Sukurtos taisyklės, kuriomis remiantis eksperimentinė realizacija generuoja koncepcinę schema iš objektų modelio:

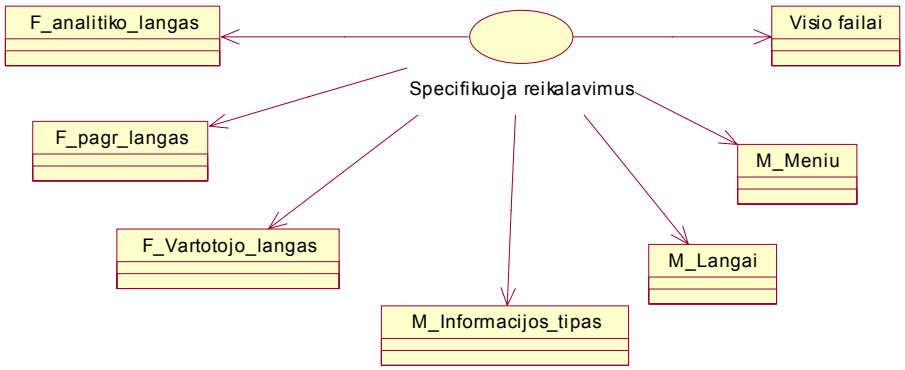
- kiekvienam duomenų šaltiniui specifikuojama viena esybė,
- kiekvienai atributų grupei specifikuojama viena esybė,
- kiekvienam duomenų šaltiniui priklausančiam atributui, kuris nepriklauso jokiai atributų grupei, specifikuojamas esybės, specifikuotos duomenų šaltiniui, atributas.
- kiekvienam atributui, priklausiančiam atributų grupei, specifikuojamas esybės, specifikuotos atributų grupei, atributas,
- jei į duomenų šaltinį įtraukta atributų grupė ir specifikuotas ryšys tarp duomenų šaltinio tipą nusakančio elemento ir grupės, tai tarp esybės, specifikuotos duomenų šaltiniui ir esybės specifikuojamas tokio pačio kardinalumo ryšys, kaip ir ryšio,
- jei į atributų grupę įtraukta kita atributų grupė, t.y. specifikuotas ryšys, tai tarp esybės, specifikuotos atributų grupei ir esybės, specifikuotos atributų grupei, specifikuojamas tokio pačio kardinalumo ryšys.

Koncepcinės schemos teisingumo tikrinimo realizacija pateikta 18 paveiksle.



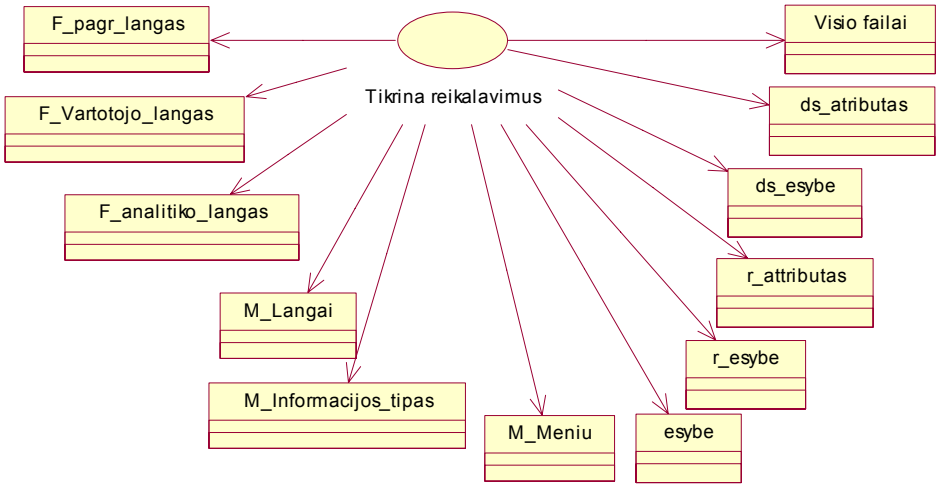
18 pav. Esybių – ryšių modelio tikrinimo realizacija

Reikalavimų specifikavimo realizacija pateikta 19 paveiksle.



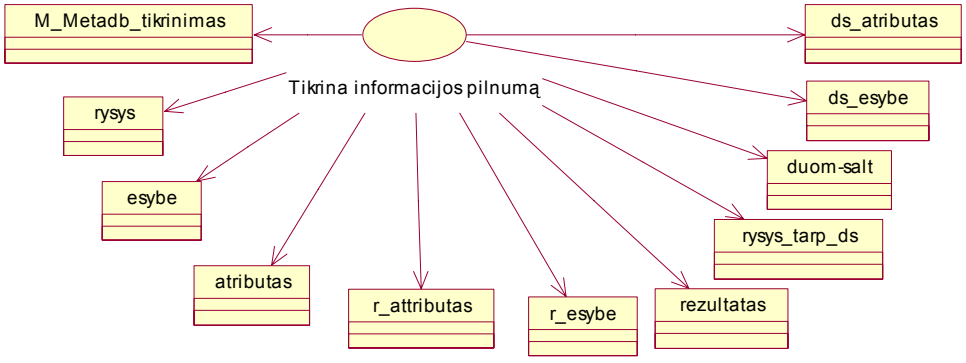
19 pav. Reikalavimų pateikimo realizacija

Reikalavimų tikrinimo realizacija pateikta 20 paveiksle.



20 pav. Reikalavimų tikrinimo realizacija

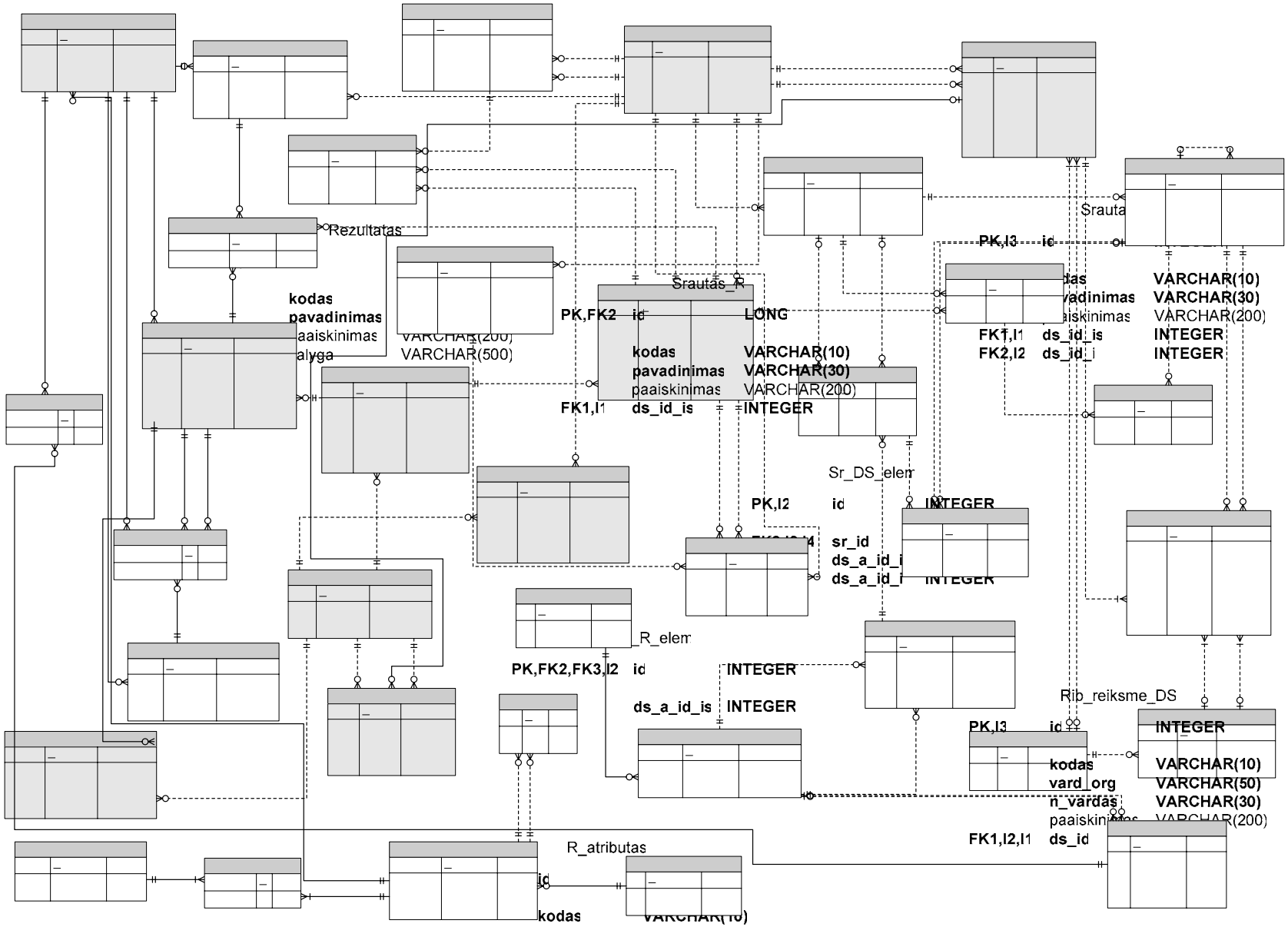
Išvedamos informacijos pagrindimo įvedamąja tikrinimo realizacija pateikta 21 paveiksle.



21 pav. Informacijos pilnumo tikrinimo realizacija

3.5 Duomenų bazės loginė schema

Duomenų bazių loginė schema pateikta 22 paveiksle.


Rezultatas
kodas
pavadinimas
paaiskinimas
alyga
VARCHAR(200)
VARCHAR(500)
Srautas_R
PK,FK2 id LONG
kodas VARCHAR(10)
pavadinimas VARCHAR(30)
paaiskinimas VARCHAR(200)
FK1,I1 ds_id_is INTEGER
Srauta
PK,I3 id
VARCHAR(10)
VARCHAR(30)
VARCHAR(200)
FK1,I1 ds_id_is INTEGER
FK2,I2 ds_id_i INTEGER
Sr_DS_elem
PK,I2 id
INTEGER
sr_id
ds_a_id_i
ds_a_id_i INTEGER
_R_elem
PK,FK2,FK3,I2 id
INTEGER
ds_a_id_is INTEGER
Rib_reiksme_DS
PK,I3 id
INTEGER
kodas VARCHAR(10)
vard_org VARCHAR(50)
n_vardas VARCHAR(30)
paaiskinimas VARCHAR(200)
FK1,I2,I1 ds_id
R_atributas
id
kodas VARCHAR(10)
vard_org VARCHAR(50)
egz_sk INTEGER
rodomumas BIT
butinumas BIT
formule VARCHAR(200)
FK1,I2,I a_id INTEGER
Veiksmo_rezultatas
PK,FK1,FK2 id LONG
Atributas
PK,I2 id INTEGER
kodas VARCHAR(10)
n_vardas VARCHAR(30)
unikalumas BIT
paaiskinimas VARCHAR(200)
PK,I
PK,I5
FK2,I4,I3
FK1,I2,I1

Kuriamo prototipo metaduomenų bazės aprašas: lentelės, jų atributai ir apribojimai, tarpusavio ryšiai pavaizduoti 2 lentelėje.

2 lentelė

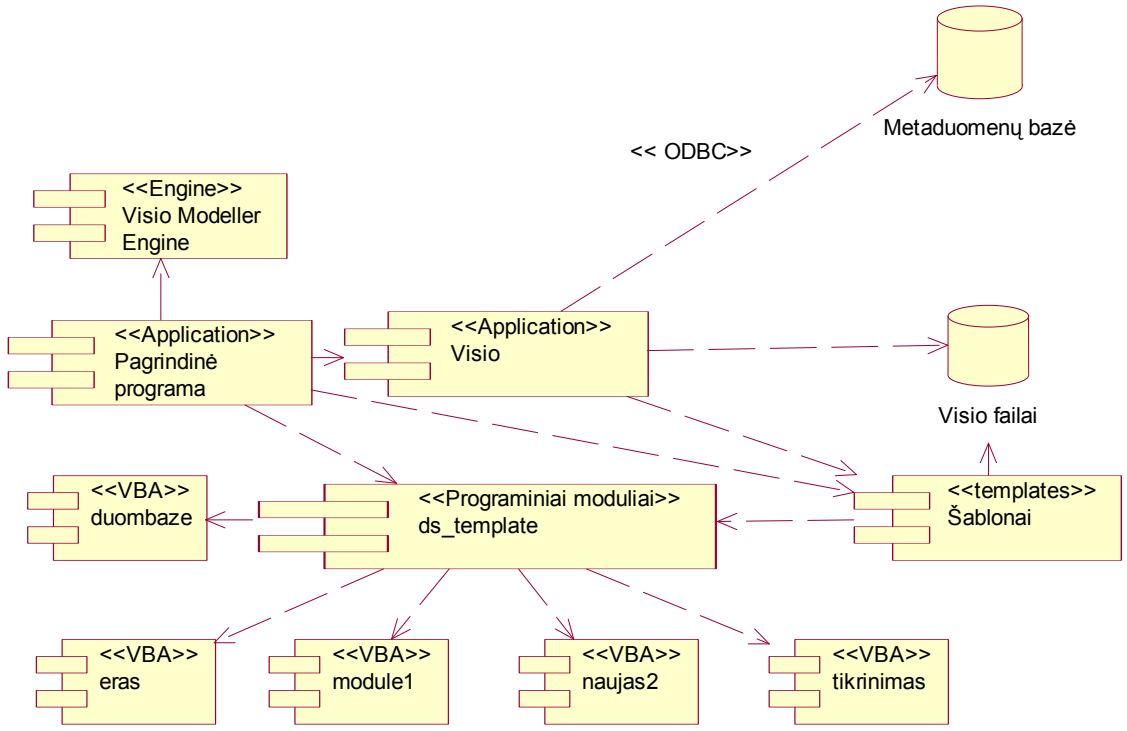
Metaduomenų bazės schemos aprašymas

Vardas	***Paaiškinimas***
Atributas	Lentelėje saugomi visi nagrinėjamo organizacijos veiklos konteksto atributai.
Esybe	Lentelėje saugomos visos nagrinėjamo organizacijos veiklos konteksto esybės.
Rysys	Lentelėje saugomi visi nagrinėjamo organizacijos veiklos konteksto ryšiai tarp esybių.
Rezultatas	Lentelėje saugomi informacija apie KIS funkcionalumo metu formuojamus rezultatus arba žodžiu perduodamus informacijos srautus.
R_atributas	Lentelėje saugomi informacija apie KIS funkcionalumo metu formuojamų rezultatų atributus.
Rib_reiksme_R	Lentelėje saugoma rezultate esančių atributų egzempliorių ribinių reikšmių informacija.
R_esybe	Lentelėje saugomi informacija apie KIS funkcionalumo metu formuojamų rezultatų išskirtas esybes.
Salygos_elem_R	Lentelėje saugoma informacija apie rezultato sudarymo sąlygos elementus.
Duom_salt	Lentelėje saugomi informacija apie organizacijos objektus (duomenų šaltinius) saugančius duomenis, reikalingus funkcijoms įvykdyti.
DS_atributas	Lentelėje saugomi informacija apie organizacijos objektų (duomenų šaltinius) saugančių duomenis, reikalingus funkcijoms įvykdyti, atributus.
Rib_reiksme_DS	Lentelėje saugoma duomenų šaltinyje esančių atributų egzempliorių ribinių reikšmių informacija.
DS_esybe	Lentelėje saugomi informacija apie esybes išskirtas iš organizacijos objektų saugančius duomenis, reikalingus funkcijoms įvykdyti.
Salygos_elem_DS	Lentelėje saugoma informacija apie duomenų šaltinio sudarymo sąlygos elementus.
Srautas_R	Lentelėje saugoma informacija apie informacijos srautą tarp duomenų šaltinio ir rezultato.
Srautas_DS	Lentelėje saugoma informacija apie informacijos srautą tarp duomenų šaltinio ir duomenų šaltinio.
Sr_R_elem	Lentelėje saugoma informacija apie informacijos srauto tarp duomenų šaltinio ir rezultato sandarą.
Sr_DS_elem	Lentelėje saugoma informacija apie informacijos srauto tarp duomenų šaltinio ir duomenų šaltinio sandarą.
Et_atributas	Lentelėje saugoma informacija apie duomenų šaltinių apdorojimo etapo metu įvedamus arba modifikuojamus atributus.
Etapas	Lentelėje saugoma informacija apie duomenų šaltinio apdorojimo etapus.
Perėjimas_Et	Lentelėje saugoma informacija apie perėjimus tarp duomenų šaltinio apdorojimo etapų.
Veiksmas_DS	Lentelėje saugoma informacija apie visus galimus duomenų šaltinių apdorojimo veiksmus.

Vardas	*Paaiškinimas*
Aktorius	Lentelėje saugoma informacija apie visus aktorius esančius nagrinėjamos veiklos kontekste.
DS_B_atributas	Lentelėje saugoma informacija apie duomenų šaltinių atributus įgyjančius reikšmes nurodytoje būsenoje.
DS_Busena	Lentelėje saugoma informacija apie duomenų šaltinių būsenas.
Perejimas_DS_B	Lentelėje saugoma informacija apie perėjimus tarp duomenų šaltinių būsenų.
Lankas_DS_B	Lentelėje saugoma informacija apie lanką tarp duomenų šaltinių būsenų ryšių.
Lankas_DS_R	Lentelėje saugoma informacija apie lanką tarp duomenų šaltinių ryšių.
Rysys_tarp_DS_B	Lentelėje saugoma informacija apie ryšius tarp duomenų šaltinių būsenų.
Rysys_tarp_DS	Lentelėje saugoma informacija apie ryšius tarp duomenų šaltinių.

3.5.1 Vartotojo reikalavimų specifikavimo metodikos realizacijos modelis

Kuriamo įvedamos ir išvedamos informacijos specifikavimo įrankio fizinis modelio vaizdas: komponentai ir tarpusavio ryšiai, pavaizduotas 23 paveiksle.



23 pav. Komponentų diagrama

Programos:

Visio – MS Visio programinė įranga, reikalinga kuriamo prototipo darbui.

Pagrindinė programa – sukurta pagrindinė programa, kuri atlieka sąsają tarp failų tarnybinė stotyje ir MS Visio paketo.

Visio Modeller Engine – MS Visio biblioteka, leidžianti kurti esybių – ryšių modelius programiškai

Programiniai moduliai – Visual Basic for Application programavimo kalba realizuotos papildomos savybės.

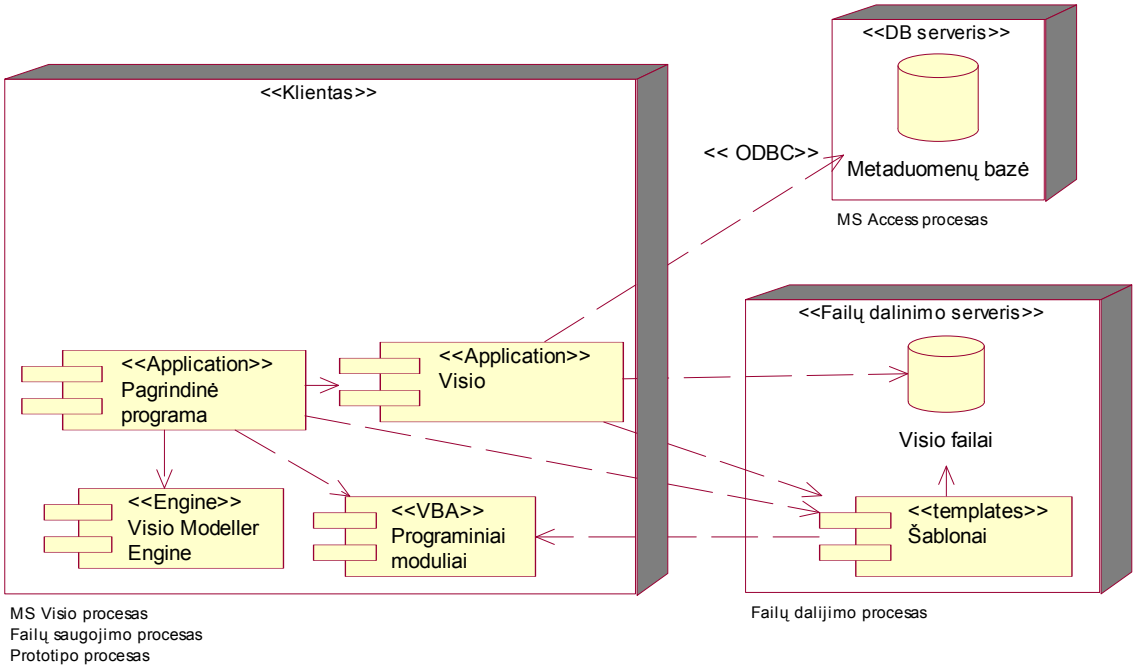
Failai:

Visio failai – pasirinkto CASE įrankio šablonai ir vartotojų sukurti dokumentai.

Metaduomenų bazė – įvedamos ir išvedamos informacijos specifikavimo prototipo duomenų saugykla.

Šablonai – MS Visio šablonai, turintys modulius, kurie papildo juos reikiamomis savybėmis.

Kuriamam prototipui reikalingi įrenginiai kartu su atliekamais procesais ir jų tarpusavio komunikavimo ryšiais pavaizduoti 24 paveiksle.



24 pav. Paskirstymo diagrama

Procesoriai:

Duomenų bazių serveris – tarnybinė stotis, kuriame veikia MS Access programa.

- *MS Access procesas* – duomenų bazių procesas

Klientas – taikomųjų programų kompiuteris, kuriuo dirba vartotojas. Jame įdiegta MS Visio programinė įranga ir sukurtas įvedamos ir išvedamos informacijos specifikavimo prototipas.

- *MS Visio procesas* – MS Visio programinės įrangos procesas.
- *Failų saugojimo procesas* – failų saugojimo į tarnybinę stotį procesas.
- *Prototipo procesas* – sukurto prototipo procesas

Failų dalinimo serveris – tarnybinė stotis, kurioje sudėti visi tinklo vartotojų dokumentai. Tarnybinėje stotyje sudedami prototipo Visio šablonai ir kuriami Visio dokumentai.

- *Failų dalinimo procesas* – failų dalinimo procesas.

4 Programinė vartotojo reikalavimų specifikavimo realizacija

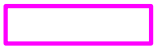
Projektinėje dalyje aprašytam modeliui realizuotas prototipas. Realizuojant įvedamos ir išvedamos informacijos specifikavimo prototipą, pasirinktas Microsoft Visio paketas, dėl jo lanksčios galimybių praplėtimo savybės. Microsoft Visio paketą galima praplėsti Visual Basic for Application programavimo kalbos pagalba. Ši programavimo kalba naudojama praplėsti visiems Microsoft kompanijos produktams.

Pasirinkimą taip pat nulėmė ir tai, kad šis paketas įsigyjamas perkant Microsoft Visual Studio .NET programavimo paketą. Visos paketo programos yra glaudžiai suintegruotos, todėl sukūrus praplėtimą Microsoft Visio paketui, jo praktinį panaudojimą galima pasiūlyti išbandyti analitikams, kurių komanda dirba su Microsoft Visual Studio .NET paketu. Tokiu būdu galima pagreitinti jo paplitimą.

Kuriant eksperimentinę sistemą, pasirinkti komponentų grafiniai žymėjimai pateikti 3 lentelėje.

3 lentelė

Komponentų grafiniai žymėjimai

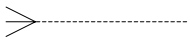
Eil. Nr.	***Komponento pavadinimas***	***Grafinis simbolis***
1.	Atributas	
2.	Atributų grupė	
3.	Atributų reikšmes ribojantis elementas	
4.	Elementas, identifikuojantis įvedamos/išvedamos informacijos šaltinio tipą	
5.	Elementas, identifikuojantis įvedamos/išvedamos informacijos šaltinio potipį	
6.	Teksto elementas	
7.	Neklasifikuotas elementas	Neklasifikuotas elementas
8.	Sąlyga	Nėra, užrašoma tekstu

Pastaba: pastorintas rėmelis naudojamas struktūros elementams, neturintiems atitikmens duomenų šaltinio šablone.

Sąryšių grafiniai žymėjimai pateikti 4 lentelėje.

4 lentelė

Sąryšių grafiniai žymėjimai

Eil. Nr.	***Pavadinimas***	***Grafinis žymėjimas***
1.	Nekartotinio atributo komponavimo į grupę ryšys	
2.	Kartotinio atributo komponavimo į grupę ryšys	
3.	Nekartotinės grupės komponavimo į grupė ryšys	
4.	Kartotinės grupės komponavimo į grupę ryšys	
5.	Ribojimas	

Sąryšis nr. 1 skirtas nekartotinio atributo komponavimui į grupę. Sąryšio pradžia – laikoma rodyklės pradžia be rodyklės, o pabaiga – kryptinė rodyklė. Sąryšio pradžia gali būti jungiama tik prie atributo, o pabaiga prie atributų grupės.

Sąryšis nr. 2 skirtas kartotinio atributo komponavimui i grupę. Sąryšio pradžia gali būti prijungta tik prie atributo, o pabaiga – prie atributų grupės.

Sąryšis nr. 3 skirtas nekartotinės atributų grupės komponavimui į atributų grupę. Abu ryšio galai gali būti sujungti tik su atributų grupėmis.

Sąryšis nr. 4 skirtas kartotinės atributų grupės komponavimui į atributų grupę. Abu ryšio galai gali būti sujungti tik su atributų grupėmis.

Sąryšis nr. 5 skirtas nurodyti atributų apribojimui. Sąryšio pradžia gali būti sujungta tik su atributą ar jų grupę ribojančiu elementu o pabaiga – su ribojamuoju atributu arba jų grupe.

4.1 Vartotojo reikalavimų specifikavimo įrankio struktūra

Kuriamo modelio komponentų diagrama pateikta 24 paveiksle, kuriame eksperimentinė realizacija dirba Microsoft Visio aplinkoje. Kadangi pati aplinka turi daug komponentų, grafinių žymėjimų, tai kuriant prototipą dalis šių savybių panaudota pakartotinai. Pakartotinis panaudojimas turi daug pranašumų, kurių pagrindiniai:

- laiko sutaupymas, nes nereikia gaišti tiek daug laiko kurti naujam produktui, kiek išmokti panaudoti esamus. Taip pat šiandienos programavimo kalbų autoriai keičia programavimo kalbų savybes į objektines, taip per kelias versijas ją padarant visiškai objektine programavimo kalba, nes pagrindinė objektinio programavimo savybė: pakartotinis panaudojimas, paveldėjimas turi didelius pranašumus prieš funkcijas ir procedūras struktūriniame projektavime,

- komponento testavimas, nereikia panaudojamo komponento testuoti. Sukurti komponentai yra ištestuoti jų autorių ir vartotojų, kurie naudoja programas su tais komponentais.

Trūkstami komponentai sukurti Visual Basic for Application programavimo kalba.

Kuriamos sistemos vartotojai yra užsakovas ir analitikas. Pagrindinis šio paketo tikslas sumažinti atstumą tarp šių žmonių grupių, nedaug padidinant abiejų pusių darbo sąnaudas.

Duomenų saugykla pasirinkta iš dviejų dalių:

- Microsoft Visio failai. Kuriamas prototipas dalį duomenų saugo grafiniame pavidale. Grafinės duomenų struktūros greičiau ir lengviau išsaugomos Microsoft Visio failuose, nei duomenų bazėje. Pagrindinis trūkumas šio pasirinkto varianto, kad Microsoft neatskleidžia saugojimo struktūros, todėl norint ateityje panaudoti grafinio modelio elementus iš išorės gali kilti nesklandumų.
- Microsoft Access duomenų bazė. Kuriama eksperimentinė realizacija metaduomenų bazėje saugos tik duomenų struktūrą, o ne pačius duomenų įrašus ar jų grafinius modelius. Duomenų struktūra užima sąlyginai nedaug diskinės vietos, lyginant su duomenimis, todėl nėra būtinybės apsunkinti paketą didele duomenų baze. Kuriant prototipą darbui tinkle taptų neišvengiama rinktis kitą duomenų bazių serverį, nes Microsoft Access nėra tinkamas tinkliniam darbui.

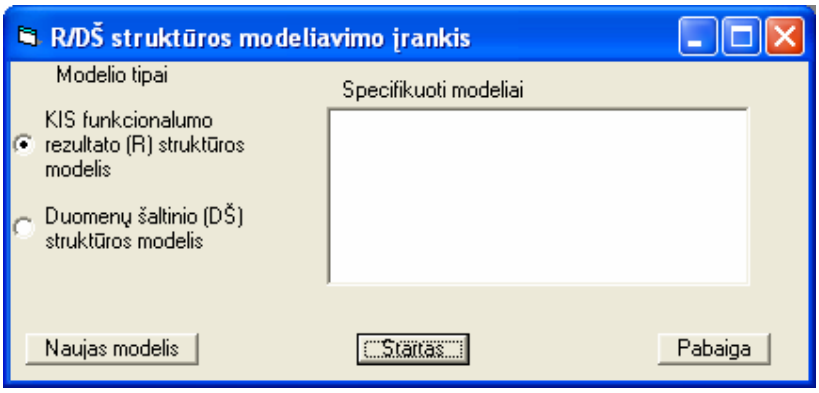
Vartotojas pateikia reikalavimus išvedamai ir įvedamai informacijai. Specifikuojant abiejų tipų informaciją, darbo proceso skirtumai atsiranda tik paskutiniuose proceso etapuose. Todėl realizuojant prototipą, pasirinkta naudoti du skirtingus šablonus. Pradedant specifikuoti vartotojo įvedamą/išvedamą informaciją Microsoft Visio paleidžiamas su konkrečiu šablonu, kuriame realizuotos papildomos savybės ir reikiami grafiniai žymėjimai. Saugant grafinę duomenų struktūrą, Microsoft Visio failuose saugoma kartu visa šablono informacija.

4.2 Realizuotos funkcijos

Realizuojant funkcijas nebuvo orientuotasi į konkrečią veiklos sritį, todėl programa pritaikyta visoms analitiko tiriamoms sritims. Prototipas skirtas analitikams, kurie specifikuoja įvedamą ir išvedamą vartotojo pateiktą informaciją. Metodika realizuota Microsoft Visio paketu.

4.2.1 Vartotojo reikalavimų specifikavimo įvedamos ir išvedamos informacijos nustatymas

Paleidus programą atsidaro pradinis langas, pateiktas 25 paveiksle.

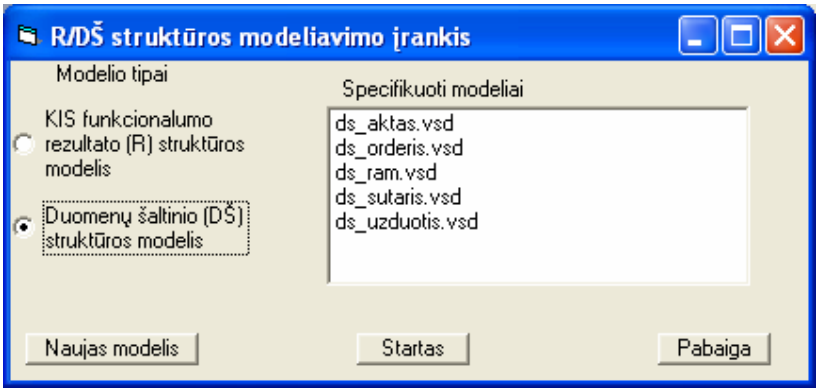


25 pav. Pagrindinis langas

Atsidariusiame lange analitikas būtinai turi nustatyti kokią informaciją nori specifikuoti: įvedamą ar išvedamą.

Informacijos tipo nustatymas reikalingas, nes įvedamai ir išvedamai informacijai specifikuoti yra sukurti skirtingi Microsoft Visio šablonai. Sukuriant naują failą, naudojamas Microsoft Visio šablonas, kuris savyje saugo su specifikavimu susijusią visą informaciją ir priemones, leidžiančias atlikti struktūros specifikavimo procesą Microsoft Visio aplinkoje.

Pasirinkus „KIS funkcionalumo rezultato (R) struktūros modelis“ punktą, bus atrinkti išvedamos informacijos grafiniai modeliai, esantys Microsoft Visio metaduomenų bazėje. Jų sąrašas bus pateiktas „Specifikuoti modeliai“ lauke (26 pav.).



26 pav. Pagrindinis langas su specifikuotų modelių sąrašu

Pasirinkus „Duomenų šaltinio (DŠ) struktūros modelis punktą, bus atrinkti įvedamos informacijos grafiniai modeliai, kurie yra išsaugoti Microsoft Visio metaduomenų bazėje. Jų sąrašas pateikiamas „Specifikuoti modeliai“ lauke.

Mygtukas „Naujas modelis“ sukuria pasirinkto tipo: įvedamos ar išvedamos informacijos, tuščią modelį ir paruošia Microsoft Visio aplinką informacijos specifikavimui.

Mygtukas „Startas“ atidaro iš specifikuotų modelių sąrašo pasirinktą modelį. Pasirinktas modelis atidaromas Microsoft Visio aplinkoje, duomenis nuskaitant iš Microsoft Visio metaduomenų bazės.

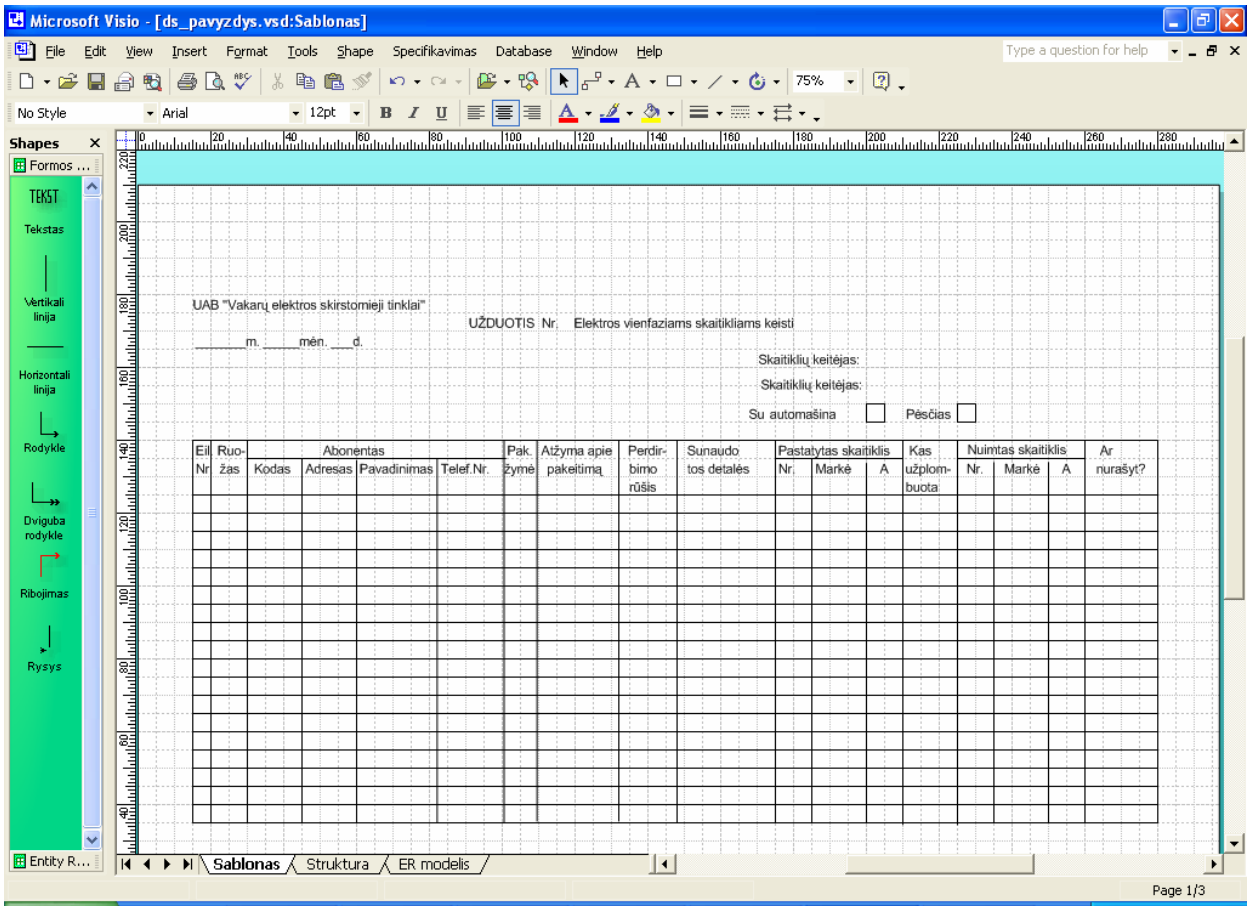
Mygtukas „Pabaiga“ pasirenkamas, kai norima pabaigti darbą. Paspaudus mygtuką uždaromas pagrindinis langas.

Pasirinkus mygtuką „Naujas modelis“ arba „Startas“, paleidžiama Microsoft Visio aplinka.

4.2.2 Vartotojo reikalavimų specifikavimo sluoksnių realizavimas

Vartotojo įvedamos ir išvedamos informacijos specifikavimo sluoksniai realizuoti Microsoft Visio puslapių pagalba. Kiekvienas puslapis atitinka vieną metodikos sluoksnį. Atidarytas Microsoft Visio langas su visais trimis sluoksniais pateiktas 27 paveiksle.

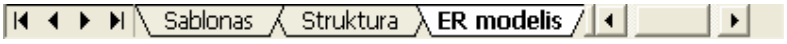
Vartotojas dažnai pateikia formas, kaip įvedama informaciją. Vienas pavyzdžių pateiktas 27 paveiksle. Tai vartotojo pateikta forma, kuria jis žino ir geba paaiškinti kiekvieno lauko kilmę.



27 pav. Microsoft Visio aplinka

Tai standartinė Microsoft Visio darbo aplinka su papildomomis funkcijomis, kurios pateikiamos meniu punkte „Specifikavimas“.

Trijų sluoksnių architektūros realizavimas pateiktas 28 paveiksle.



28 pav. Sluoksnių sąrašas

Puslapis „Šablonas“ atitinka pirmą projektinėje dalyje aprašytą sluoksnį. Jame specifikacijos saugykloje užregistruojamas formos šablonas įprastine vartotojui forma. Pateikiant vartotojui nestruktūrizuotus duomenis, analitikas privalo išsiaiškinti duomenų srauto struktūrą ir ją suprantamai pateikti vartotojui. Pavyzdžiui, jei specifikuojama išvedama informacija, kuri yra ataskaita, tai šiame lape įvedamas ataskaitos pavadinimas, nurodomi ataskaitos rekvizitai, pasirinktame lapo formate jie išdėstomi taip, kaip juos vartotojas pageidaus matyti sukurtoje informacinėje sistemoje, parenkamas raidžių dydis, stilius.

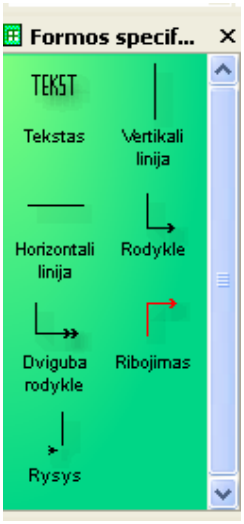
Puslapis „Struktūra“ atitinka antrą projektinėje dalyje aprašytą sluoksnį. Jame operuojama visais duomenų struktūros objektais, dažniausiai suformuotais iš vartotojo pateiktų formų ir ataskaitų.

Duomenų struktūros specifikacija susideda iš duomenų struktūros pavadinimo ir objektų aibės. Analitikas, remdamasis savo patirtimi ir bendraudamas su vartotoju, klasifikuoja elementus, jungia juos į grupes, veda sąlygas, nurodo ribojančius ir ribojamus elementus ir esant poreikiui specifikuoja naujus struktūros elementus, neturinčius atitikmens šablone. Yra šeši duomenų struktūros objektų ar jų grupių kvalifikacijos tipai: tekstas, atributas, atributų grupė, apribojimai, duomenų šaltinio tipas ir duomenų šaltinio potipis. Analitikas, bendraudamas su vartotoju remiasi šablonu, kurį vartotojas puikiai žino, o specifikavimą ir klasifikavimą atlieka struktūros šablone. Vartotojas bendravimo metu stebi, kaip analitikas interpretuoja struktūros elementus. Tokiu būdu vyksta abipusis mokymasis: analitikas gilinasi į dalykinę sritį, o vartotojas į formalų informacijos aprašymą.

Puslapis „ER modelis“ atitinka trečią projektinėje dalyje aprašytą sluoksnį – koncepcinio modelio fragmentą. Koncepcinis modelis yra kuriamos sistemos duomenų bazių modelio fragmentas. Koncepcinis modelis generuojamas išrenkant atributų grupes, juos sudarančius atributus, ir tarpusavio ryšius. Analitikas gali patikrinti sugeneruoto esybių-ryšių modelio, vartotojo pateiktos duomenų struktūros, teisingumą ir pilnumą

4.2.3 Grafinių žymėjimų realizacija

Realizuoti grafiniai žymėjimai, kurie aprašyti 3 ir 4 lentelėse, pateikti 29 paveiksle. Grafiniai žymėjimai patalpinti atskirame Microsoft Visio šablone.

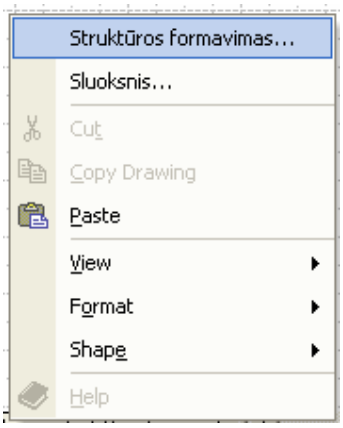


29 pav. Prototipo grafiniai žymėjimai

4.2.4 Vartotojo reikalavimų specifikavimo sluoksnių funkcijų realizacija

Nuosekliai atliekant vartotojo įvedamos ir išvedamos informacijos specifikavimą, pirmiausia suformuojamas dokumento formos atitikmuo Microsoft Visio struktūros lange (27 paveikslas).

Struktūros formavimo ir filtravimo funkcijos patalpintos šablono puslapio iškrentančiame meniu (30 paveiksle).

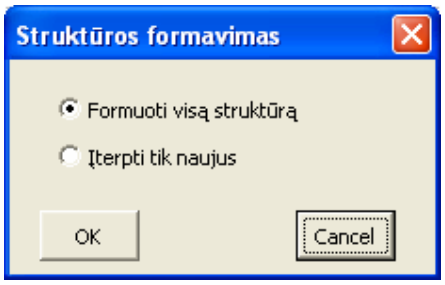


30 pav. Šablono iškrentantis meniu

Meniu punktas „Struktūros formavimas“ formuoja duomenų struktūros specifikaciją ir patalpina struktūros puslapyje.

Meniu punktas „Sluoksnis“ leidžia filtruoti objektus, kad vartotojui būtų lengviau dideliame dokumente.

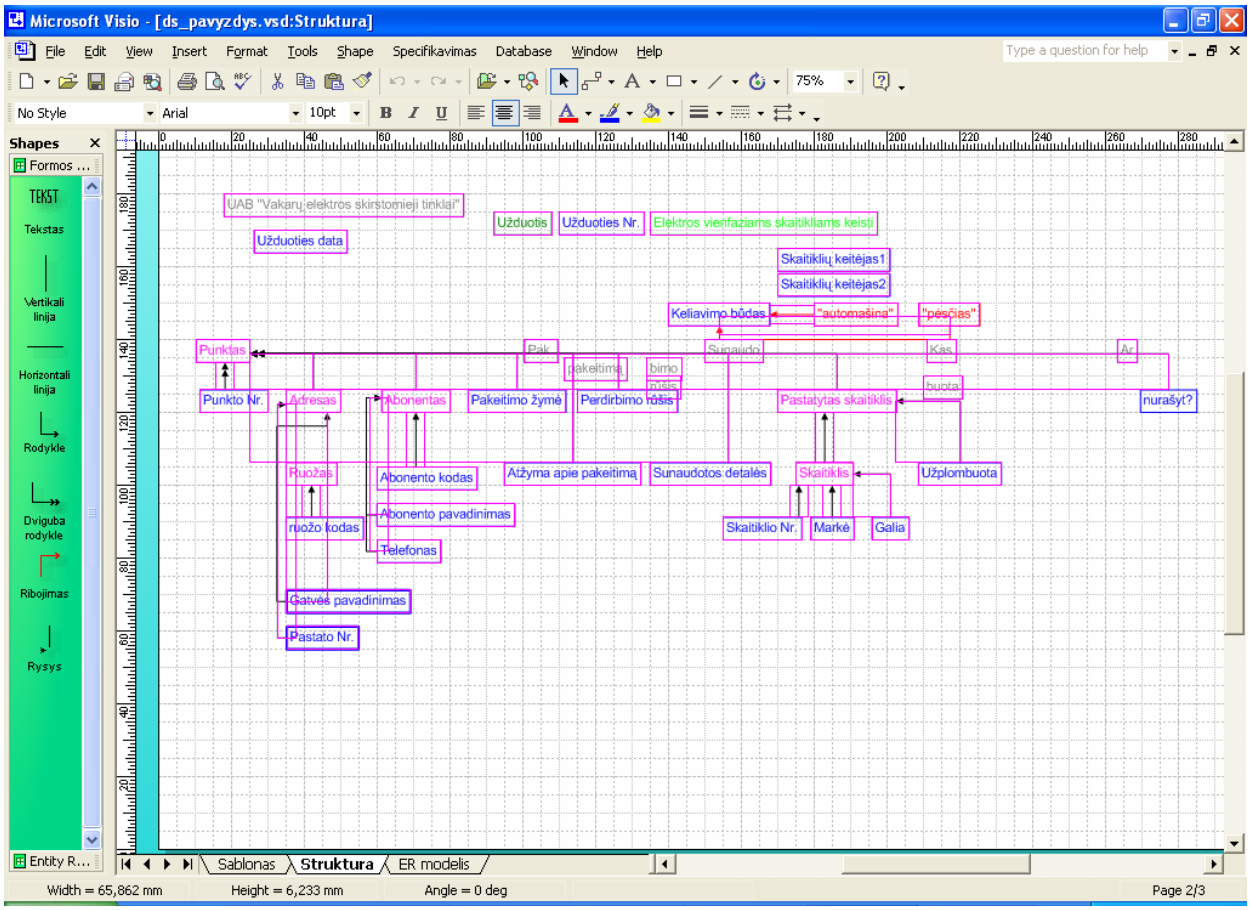
Struktūros formavimo langas pateiktas 31 paveiksle.



31 pav. Struktūros formavimo langas

Formuojant duomenų struktūros specifikaciją, galima formuoti struktūrą, ignoruojant anksčiau suformuotą struktūrą, arba yra galimybė įkelti tik naujus elementus.

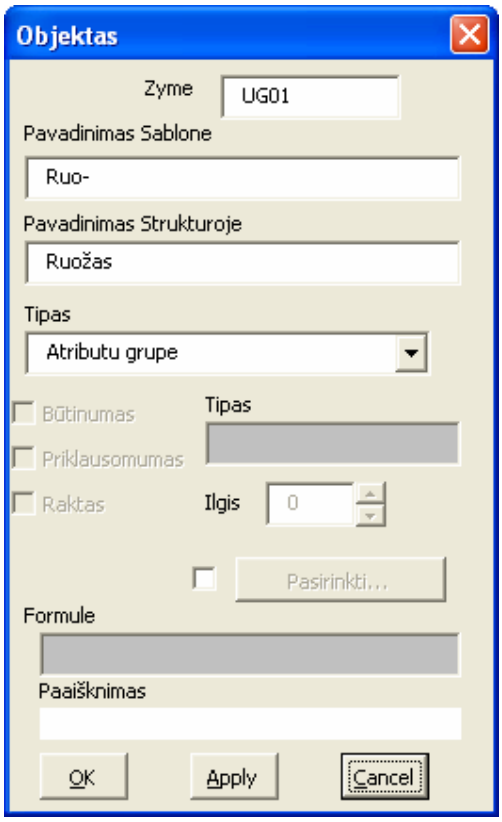
Formuojant duomenų struktūros specifikaciją, visi šablono elementai perkeliami į struktūros langą. Sutvarkyta duomenų struktūra pateikta 32 paveiksle. Jei perkeltų elementų neužtenka, šiame sluoksnyje galima įvesti papildomus struktūros elementus.



32 pav. Analizuojamo pavyzdžio sutvarkyta duomenų struktūra

Kaip matome iš sutvarkytos struktūros, kiekvienas elemento tipas turi žymėjimą, aprašytą 3 lentelėje.

Struktūros elemento parametrų nustatymo langas pateiktas 33 paveiksle.

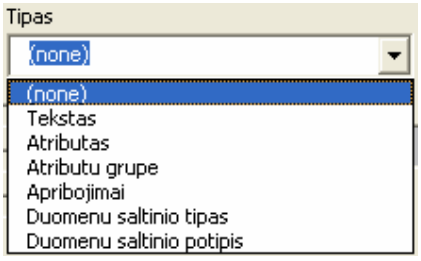


33 pav. Objektų parametrų nustatymas

Nustatant objekto parametrus būtini laukai:

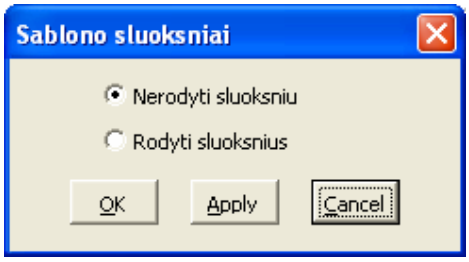
- pavadinimas Šablone,
- pavadinimas Struktūroje.

Objekto tipas pasirenkamas iš sąrašo, pateikto 34 paveiksle.



34 pav. Objektų parametrų nustatymas

Šablono filtravimo langas pateiktas 35 paveiksle.



35 pav. Šablono sluoksnių filtravimo langas

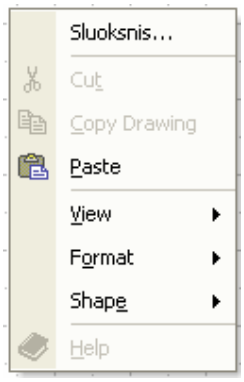
Mygtukas „OK“ atlieka pasirinktą sluoksnių vaizdavimą ir uždaro dialogo langą.

Mygtukas „Apply“ atlieka pasirinktą sluoksnių vaizdavimą, tačiau dialogo lango neuždaro. Paliekama vartotojui teisė dar pakeisti savo apsisprendimą.

Mygtukas „Cancel“ atšaukia dialogo vaizdavimą ir atliktus jame veiksmus.

Pateikta duomenų struktūra 32 paveiksle artima vartotojui. Vartotojas dalyvauja elementų parametrų nustatymo procese, pasakydamas kiekvieno elemento savybes.

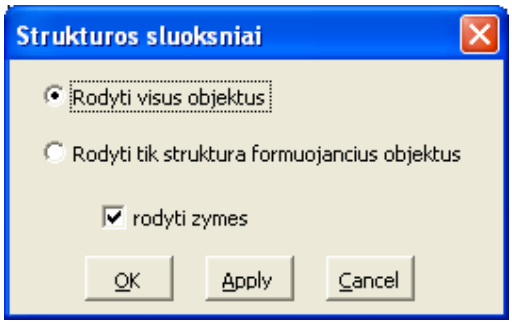
Duomenų struktūros funkcijos patalpintos struktūros puslapio iškrentančiame meniu (36 paveikslas).



36 pav. Struktūros iškrentantis meniu

Meniu punktas „Sluoksnis“ yra sukurtas papildomai. Jis leidžia filtruoti struktūros elementų vaizdavimą.

Vaizdavimo filtro dialogo langas pateiktas 37 paveiksle.

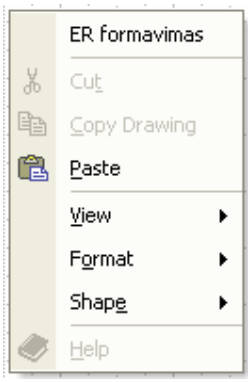


37 pav. Struktūros sluoksnių filtravimo langas

Pasirinkus „Rodyti visus objektus“, struktūros lange bus rodomi visi objektai, kurie perkelti iš šablono lango.

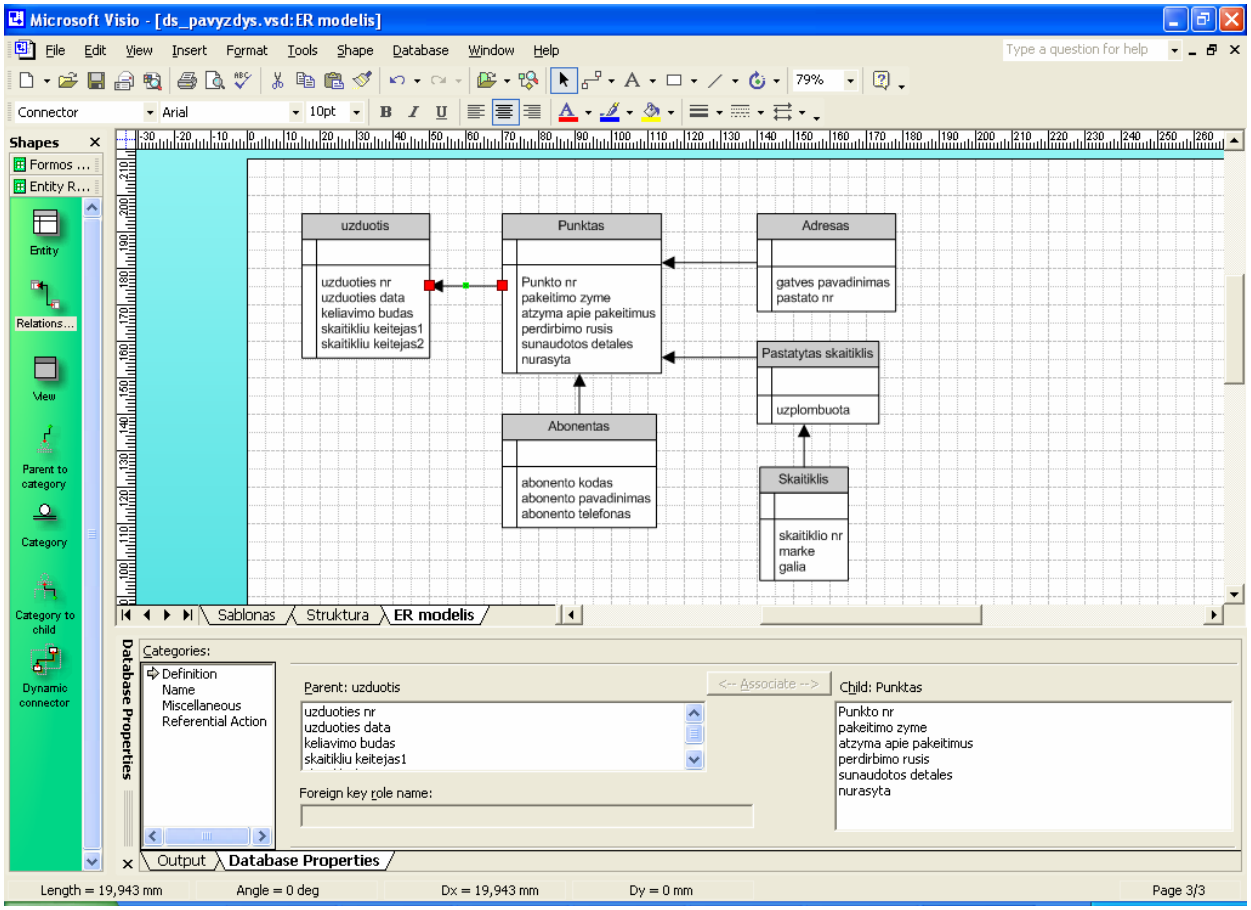
Pasirinkus „Rodyti tik struktūrą formuojančius elementus“ bus rodomi tik struktūrą formuojantys elementai. Jos neformuojantys elementai nebus ištrinti iš struktūros lango, tačiau bus panaikintas jų matomumo savybė.

Suformavus struktūrą ir ją patikrinus vartotojui, generuojama pateiktų reikalavimų koncepcinės schemos fragmentas. Generavimo funkcija realizuota „ER modelio“ lapo iškrentančiame meniu (38 pav.).



38 pav. ER eskizo iškrentantis meniu

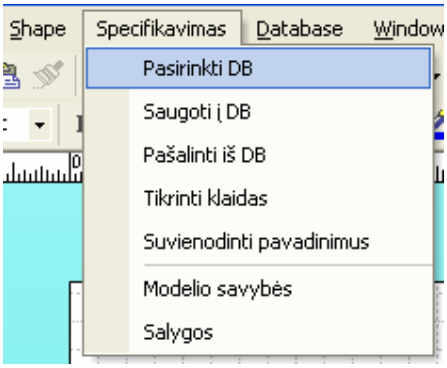
Pasirinkus meniu punktą „ER formavimas“ atliekami esybių – ryšių modelio generavimo veiksmai. Modelis generuojamas išrenkant atributų grupes, juos sudarančius atributus, ir tarpusavio ryšius. Sukurtos esybės ir jų tarpusavio ryšiai patalpinami ER modelio lape (39 paveikslas). Suformuot1 koncepcinę schemą analitikas turi patikrinti pagal esybių – ryšių tikrinimo taisykles, užtikrindamas jos teisingumą, įrašų integralumą.



39 pav. Sugeneruotas esybių – ryšių modelis

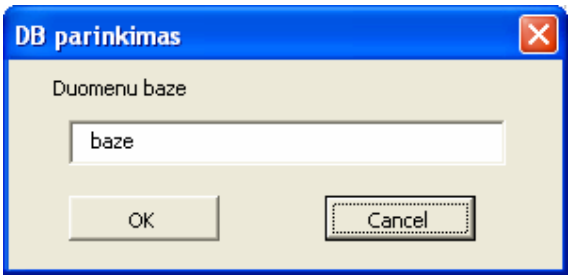
4.2.5 Vartotojo reikalavimų specifikavimo ryšio su metaduomenų baze realizavimas

Sugeneruotą koncepcinės schemos fragmentą galimą išsaugoti Microsoft Visio metaduomenų bazėje. Įvedamos ir išvedamos informacijos specifikavimo prototipe realizuotas modelio elementų išsaugojimas taip pat Microsoft Access metaduomenų bazėje. Realizuotos funkcijos patalpintos meniu punkte „Specifikavimas“ (40 paveikslas).



40 pav. Meniu punktas „Specifikavimas“

Meniu punktas „Pasirinkti DB“ leidžia nurodyti norimą metaduomenų bazės failą. Kuriant kelias koncepcines schemas, jas galima saugoti atskirose metaduomenų bazėse. Metaduomenų bazės pasirinkimo dialogo langas pateiktas 41 paveiksle.



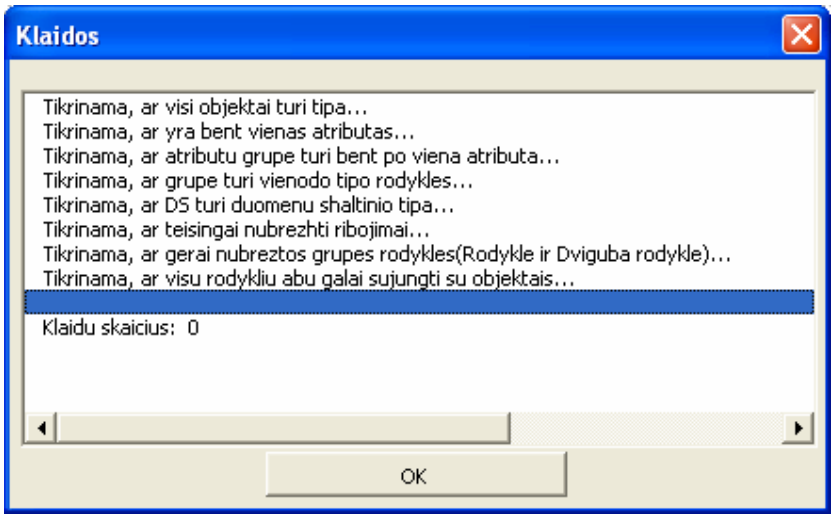
41 pav. Duomenų bazės pasirinkimo langas

Įvedimo lange „Duomenų bazė“ įvedamas metaduomenų bazės failo vardas.

Meniu punktas „Saugoti į DB“ išsaugo kuriamo koncepcinio modelio fragmento duomenis metaduomenų bazėje. Išsaugant duomenis į metaduomenų bazę, atliekamas duomenų tikrinimas (40 paveikslas).

Meniu punktas „Pašalinti iš DB“ pašalina visus aktyvios įvedamos/išvedamos informacijos elementus iš metaduomenų bazės.

Meniu punktu „Tikrinti klaidas“ realizuotas modelio duomenų tikrinimas. Tikrinimas atliekamas sintaksiniu būdu. Atlikto duomenų tikrinimo langas pateiktas 42 paveiksle. Tikrinimas automatiškai atliekamas saugant duomenis į metaduomenų bazę.



42 pav. Klaidų tikrinimo langas

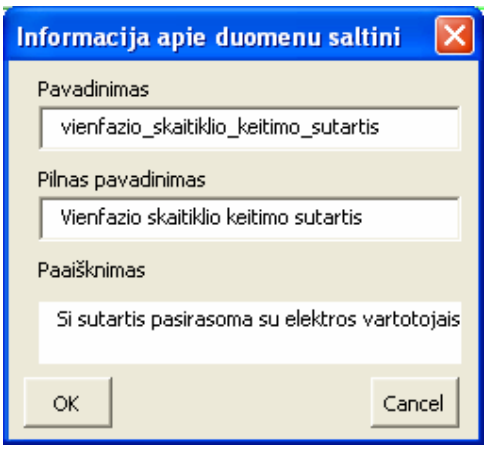
Tikrinimo sąlygos tikrina:

- visų objektų suklasifikavimą, t.y. kiekvienas objektas turi turėti nustatytą tipą. Jei objektas nėra priskirtas kuriam nors tipui, tai nurodomas neklasifikuoto objekto pavadinimas (pavadinimas imamas iš struktūros sluoksnio);
- atributų egzistavimą, t.y. įvedamos ir išvedamos informacijos duomenų šaltinis turi turėti bent vieną atributą;
- atributų egzistavimą atributų grupėje, t.y. kiekviena atributų grupė privalo turėti bent vieną atributą. Jei grupė neturi atributų, tai sekančioje eilutėje pateikiamos atributų grupės, neturinčios atributų,
- atributų grupių elementų ryšio vienodumas, t.y. kiekvieną atributų grupę gali sudaryti tik vienodo tipo atributai: pasikartojantys arba nepasikartojantys. Jei ši sąlyga netenkinama, tai nurodomas grupės pavadinimas;
- specifikuojamos informacijos tipas, t.y. kiekvienas išvedamos arba įvedamos informacijos šaltinis turi turėti tipą. Jei informacijos tipas nenurodytas, tai taip ir išvedama klaidų tikrinimo lange. Jei nurodyti du ir daugiau informacijos tipų, tai taip pat laikoma, kad nenurodytas specifikuojamos informacijos tipas;
- atributų reikšmes ribojančių ryšių teisingumas, t.y. atributus gali riboti tik kiti atributai arba ribojantys elementai. Tarp šių atributų gali būti nubrėžti tik ribojimo ryšiai. Jei ši sąlyga netenkinama, tai pranešimo lange nurodoma nuo kurio ir iki kurio objekto ryšys yra neteisingas;
- grupės narių teisingumas, t.y. grupę gali sudaryti tik atributai ir grupės. Jei nurodyta, kad grupę sudaro ir kiti elementai, tai pranešimo lange nurodoma, kad tokie elementai negali būti grupės nariais;

- Ryšių prijungimo teisingumas, t.y. abu visų ryšių galai turi būti sujungti su objektais. Jei bent vienas ryšio galas nesujungtas, tai pranešimo lange nurodomas tik objektų, nejungiančių ryšių skaičius;
- Objektų žymių unikalumo tikrinimas, t.y. viso projekto metu atributams ir grupėms turi būti priskirti unikalūs pavadinimai (Prototipe vadinama žymėmis). Radus pasikartojančią žymės reikšmę, nurodomi objektų pavadinimai pranešimų lange.

Meniu punktas „Suvienodinti pavadinimus“ atlieka duomenų korekcijas metaduomenų bazėje. Išsaugojus koreguotus įvedamos/išvedamos informacijos duomenis, senuose informacijos modeliuose, esančiuose metaduomenų bazėje, savybės lieka nepakitus. Ši funkcija atlieka vardų suvienodinimą.

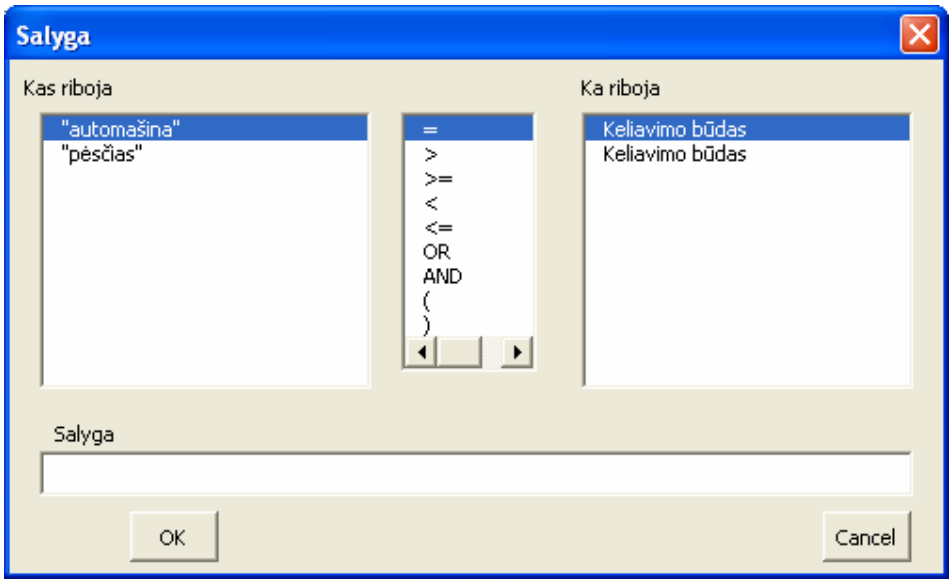
Meniu punktas „Modelio savybės“ iškviečia papildomos modelio informacijos įvedimo langą, kuris pateiktas 43 paveiksle.



43 pav. Informacijos aprašymo langas

Iškviestame lange įvedamas įvedamos/išvedamos informacijos sutrumpintas ir pilnas pavadinimai ir pilnas aprašymas.

Sąlygos skirtos nagrinėjamo dokumento apribojimams įvesti. Apribojimų įvedimo langas pateiktas 44 paveiksle.



44 pav. Informacijos aprašymo langas

Stulpelis „Kas riboja“ automatiškai užpildomas elementais, kurie struktūroje ką nors riboja. Stulpelis „Ką riboja“ užpildomas elementais, kurie struktūroje yra ribojami. Pati sąlyga užrašoma laisva forma eilutėje „sąlyga“. Ši funkcija padeda sukonstruoti sąlyga, tačiau teisingumo netikrina.

5 Vartotojo reikalavimų specifikavimo realizacijos įvertinimas

5.1 Vartotojo įvedamos ir išvedamos informacijos specifikavimo metodikos įvertinimas

Vartotojo reikalavimus specifikuojančios metodikos apžvelgtos pirmoje šio darbo dalyje. Siūlomas vartotojo įvedamos ir išvedamos informacijos specifikavimo konceptualizavimas palygintas su žinomomis specifikavimo metodikomis 5 lentelėje. Kriterijai vertinami penkiabalėje sistemoje, t.y. aukščiausias įvertinimas 5, o žemiausias – 1. Kriterijaus įvertinimas 5 apibrėžia, kad metodas yra geresnis už kitus, tačiau tai nereiškia, kad jis yra geriausias.

5 lentelė

Siūlomos metodikos palyginimas su žinomomis metodikomis

Metodas ***Kriterijus***	***EPRE***	***FORE***	***Rummler - Brache***	***R. Barker***	***G. Booch***	***Siūlomas metodas***
Reikalavimų specifikavimas	3	4	5	5	5	5
Informacijos pagrindimas	2	3	3	3	3	4
Vartotojo dalyvavimas	2	3	3	3	3	4
Modeliavimas	3	4	5	5	5	5
Saugyklos modelis	3	4	4	5	4	4
Viso	13	18	20	21	20	22

Kaip matyti iš 5 lentelės, siūlomas metodas yra sąlyginai pranašesnis už žinomus, o veiklos procesų atvirkštinė inžinerija pakankamai daug atsilieka nuo siūlomos metodikos, nors ir paremta ta pačia J. Choobineh pasiūlyta formų analize.

Kiekvieno kriterijaus detalesnis paaiškinimas pateiktas sekančiuose žingsniuose:

- vertinant reikalavimų specifikavimą, atsižvelgta į metodikos teisingumą, praktinį pritaikomumą, patogumą analitiko ir vartotojo atžvilgiu,
- vertinant informacijos pagrindimą, svarbiausias dėmesys kreiptas įvedamos ir išvedamos informacijos susiejimui. Dažnai vartotojas pateikia vieną informaciją, kaip įvedamą, o kuriamos informacinės sistemos rezultatuose pageidauja matyti rezultatus, kurių neįmanoma išskaičiuoti iš pateiktos informacijos. Tokia situacija iššaukia įvedamos informacijos specifikavimo korekcijas, kas gražina analitiką vienu žingsniu atgal. Šitokia situacija gali

nesusiklostyti, jei analitikas yra kompetentingas projektavimo ir tiriamoje srityse ir sugeba numatyti tokias situacijas iš anksto,

- vertinant vartotojo dalyvavimą, atsižvelgta į vartotojo indėlį į vartotojo reikalavimų specifikavimą, jo teisingumo tikrinimą,
- tikrinant modeliavimą, vertintas modelio patogumas analitikui ir vartotojui, elementarių operacijų, tokių kaip reikalavimo įterpimas, šalinimas, koregavimas,
- vertinant saugyklos modelį, atsižvelgta į saugyklos praplėtimo, pritaikymo specializuotiems poreikiams, priėjimo galimybė trečios šalies įrankiais.

Įvertinus visus kriterijus, pateikiami pagrindiniai siūlomo metodo pranašumai:

- įvedamos informacijos pagrindimas išvedamąja, taip išvengiant pakartotinio įvedamos informacijos specifikavimo,
- gilesnis vartotojo įtraukimas į specifikavimo procesą, įtraukiant jį į informacijos tikrinimo etapą,
- nesudėtingas saugyklos priėjimas trečios šalies įrankiais, paliekant atvirą metodo praplėtimą naujomis savybėmis.

5.2 Vartotojo reikalavimų specifikavimo duomenų saugyklos įvertinimas

Kuriant eksperimentinę realizaciją, viena pagrindinių sistemos sudėtinių detalių yra metaduomenų saugykla. Saugykla turi tokią svarbą, nes yra ta CASE architektūros dedamoji, kuri integruoja skirtingų projektavimo fazių duomenis į vieną visumą ir susieja tarp visų fazių. Todėl vertinant metaduomenų saugyklą, buvo atsižvelgta į keletą pagrindinių kriterijų, kuriuos turėtų tenkinti realizacijos metaduomenų bazė. Kriterijai įvertinti penkiabalėje sistemoje, kaip ir vertinant vartotojo įvedamos ir išvedamos informacijos specifikavimo metodą (6 lentelė).

6 lentelė

Saugyklos įvertinimas

Kriterijai	***Įvertinimas***	***Pastabos***
Grafinės informacijos saugojimas	4	Grafiniai modelių žymėjimai saugomi Microsoft Visio failuose, todėl kiekvienas modelis turi atskirą failą. Būtų patogiau turėti visus žymėjimus vienoje metaduomenų bazėje, kartu nustatant ryšius ir kardinalumus tarp jų, o ne jų dublikatus SQL standarto duomenų bazėje.
Išskirtų objektų informacijos	5	Visa matoma informacija saugoma Microsoft Access metaduomenų saugykloje. Eksperimento metu specifikuojant

Kriterijai	*Įvertinimas*	*Pastabos*
saugojimas		vartotojo reikalavimus informacijos praradimo nepastebėta. Galima teigti, jog bet kurie vartotojo pateikti reikalavimai įvedamai ar išvedamai informacijai teisingai išsaugomi siūlomoje metaduomenų saugykloje ir galės būti atkurti neprarandant informacijos.
Atskirų specifikavimo fragmentų susietumas	4	Siūlomame metode yra numatytas specifikuojamos informacijos susiejimas tarpusavyje, panaikinant perteklinę informaciją. Objektų vienodumo identifikavimui naudojama semantinė žodžių analizė, todėl skirtingiems vartotojams tą patį objektą vadinant skirtingai, iškiltų susiejimo problema. Analitikas turi užtikrinti, kad ši situacija nesusiklostyt ir visi kuriamos sistemos dalyviai vartotų tą patį terminų žodyną.
Viso:	13	

Įvertinus siūlomą saugyklą pagal užbrėžtus reikalavimus, matoma, jog ji tenkina 87% reikalavimų. Metaduomenų saugykla yra atvira tobulinimui, todėl priimta, kad toks rezultatas yra tenkintinas.

5.3 Vartotojo įvedamos ir išvedamos informacijos specifikavimo prototipo įvertinimas

Rinkoje yra daug CASE paketų, kurie apima visas projektavimo stadijas: vartotojo reikalavimų surinkimas, patikrinimas ir kuriamos informacinės sistemos programinio kodo, duomenų bazės sugeneravimą. Šiame etape padarytos klaidos kainuoja brangiausiai, nes tai vienas iš pirmųjų kūrimo etapų.

Yra sukurta gausybė CASE paketų, kurių viena iš atliekamų funkcijų yra vartotojo reikalavimų specifikavimas. Dauguma yra ne pirmosios versijos. Tai parodo kad šis klausimas išlieka aktualus. Egzistuojančių ir kuriamo įvedamos ir išvedamos informacijos specifikavimo prototipo palyginimas pateiktas 7 lentelėje. Kriterijų vertinimui, kaip ir kituose įvertinimuose, pasirinkta penkiabalė vertinimo sistema.

7 lentelė

CASE paketų ir prototipo palyginimas

CASE paketas Kriterijus	*Provision WorkBench*	*Rational Suite*	*Oracle Suite*	*Sukurtas prototipas*
Reikalavimų specifikavimas	5	5	5	5

Reikalavimų saugojimas	4	4	5	4
Reikalavimų pagrindimas	3	3	3	5
Generavimas	4	4	5	5
Viso:	16	16	18	19

Kaip matyti iš 7 lentelės, sukurtas prototipas pranašesnis už jau esamus CASE paketus vartotojo įvedamos ir išvedamos informacijos specifikavimo aspektu. Kiekvieno kriterijaus detalesnis vertinimas paaiškintas sekančiuose žingsniuose:

- vertinant įvedamos ir išvedamos informacijos specifikavimą, atsižvelgta į naudojamą metodiką, pritaikomumą,
- vertinant informacijos saugojimą, atkreiptas dėmesys į metaduomenų saugyklos praplėtimą,
- vertinant informacijos pagrindimą, atsižvelgta į išvedamos informacijos pagrindimą įvedamąja, nes neįmanoma paskaičiuoti, tai, ko vartotojas neįvedama,
- vertinant generavimą, atsižvelgta į suderinamumą su skirtingomis tvarkyklėmis.

Išvados

1. Atlikta vartotojo įvedamos ir išvedamos informacijos specifikavimo metodikų analizė. Juose atkreiptas dėmesys į sąlyginai mažą vartotojo dalyvavimą reikalavimų surinkime ir tikrinime, todėl sudaryti kokybės kriterijai, kurie turi būti siekiami, norint įtraukti daugiau vartotoją (kurie yra siekiami šiame darbe).
2. Įvairūs autoriai siūlo metodus gilesniam vartotojo įtraukimui, tačiau dauguma jų turi realizacijos prototipą, o esami CASE paketai, specifikuojant įvedamą ir išvedamą informaciją, vartotojo neįtraukia į informacijos tikrinimą, o pasikliauja analitiko žiniomis tiriamoje veiklos sferoje, todėl šiame darbe pateikiamas šios problemas vienas iš sprendimų.
3. J. Choobineh ir kitų pasiūlytu metodikos pagrindu atlikta projektinė analizė ir pasiūlytas gilesnis vartotojo įtraukimas į reikalavimų surinkimo procesą: reikalavimų struktūrizavimą ir tikrinimą.
4. Įvedamos ir išvedamos informacijos specifikavimo prototipo realizavimas Microsoft Visio aplinkoje pagreitino jo kūrimą, nes pakartotinai panaudoti komponentai: grafinė vartotojo sąsaja, fizinio duomenų bazių lygio formavimas iš koncepcinio modelio.
5. Pasiūlyto metodo pagrindu sukurta metaduomenų bazė, kurioje saugoma vartotojo įvedama ir išvedama informacija. Eksperimento metu specifikuojant vartotojo reikalavimus informacijos praradimo nepastebėta, todėl galima teigti, kad informacija išsaugoma teisingai.
6. Atlikti skirtingų dalykinių sričių eksperimentai parodė, kad vartotojas įtraukiamas į įvedamos ir išvedamos informacijos tikrinimo procesą.
7. Siūlomas vartotojų įvedamos ir išvedamos informacijos specifikavimo metodas palygintas su pagrindiniais žinomais metodais. Nustatyta, kad išvedamos informacijos pagrindimas įvedamąja ir vartotojo įtraukimas į sustruktūrizuotų reikalavimų tikrinimą lemia siūlomo metodo pranašumą prieš analizės dalyje aptartus metodus.
8. Atliktos analizės ir eksperimento pagrindu nustatyta, kad siūlomas vartotojo įvedamos ir išvedamos informacijos specifikavimo konceptualizavimas gali būti naudojamas realių IS kūrime.

Literatūros šaltiniai

1. **R. Barker.** CASE*METHOD: Entity Relationship Modelling. *Addison – Wesley Publ. Co., New York*, 1990.
2. **G. Booch.** Object-Oriented Analysis and Design with Applications (2nd Edition). *Addison-Wesley Publ. Co., New York*, 1993
3. **R. Butkienė, R. Butleris.** Extending functionality of CASE tools to support requirements engineering. *Computer Science Reports: Emerging Database Research in East Europe: proceedings of the pre-conference workshop of VLDB 2003.* Brandenburg, 2003, no. 14, p. 12-16.
4. **R. Butkienė, R. Butleris.** The approach for user requirements specification. *Advances in Databases and Information Systems: 5th East-European Conference ADBIS'2001: research communications.* Vilnius, 2001, Vol. 1, pp. 225-240.
5. **R. Butkienė, V. Savickas.** Reikalavimų KIS įvedamai ir išvedamai informacijai specifikavimas. *Informacinės Technologijos*, 2002, 401-407 psl.
6. **R. Butleris, V. Trijonis.** Įvedamos ir išvedamos informacijos specifikavimo konceptualizavimas. *Informacinės Technologijos*, 2004, 400-408 psl.
7. **J. Chobineh, M. Mannino.** An Expert Database Design System Based on Analysis of Forms. *IEEE transactions on software engineering*, Vol.14, 1998.
8. **J. Choobineh, M. Mannino, V. Tseng.** A form-based approach for database analysis and desigh. *Communications of the ACM*, 1992, Vol.35, No 2, pp.108-120.
9. **K. -H. Kim, Y. -G. Kim.** Process reverse engineering for BPR: A form-based approach. *Information & Management*, 1998, Vol 33, 187-200 psl.
10. **H. Lee, Ch. Yoo.** A Form Driven Object-Oriented Reverse Engineering Methodology. *Information System*, 2000 Vol. 25, No 3, pp. 235-259 .
11. **P. Loucopoulos , V. Karakostas.** System Requirements Engineering. *McGraw-Hill*, 1995.
12. **M. Mannino, V. P. Tseng.** Inferring database requirements from examples in forms. *Seventh International Conference on Entity – Relationship approach*, Rome, 1988, pp. 1-25.
13. **J. R. Rumbaugh.** Object – Oriented Modelling and Design. *Prentice Hall*, 1990.
14. **M. H. Walker, N. Eaton.** Microsoft Office Visio 2003 Inside Out. *Microsoft Press*, 2003, pp. 912
15. **B. Wangler.** Contributions of Function Requirements Modelling. Doctoral Thesis. *Stockholm University*, 1993,189–230.

16. **P. Loucopoulos, V. Karakostas.** System Requirements Engineering. *McGraw-Hill*, 1995.
17. Proforma Products – Know Your Business. [žiūrėta 2005-01-10]. Prieiga per internetą: http://www.proformacorp/Products/provision.asp
18. UML diagramming, OO software modeling, Source code engineering Tool MagicDraw UML from No Magic. [žiūrėta 2005-01-10]. Prieiga per internetą: http://www.magicdraw.com/
19. Rational Suite – Product Overview. [žiūrėta 2005-01-10]. Prieiga per internetą: http://www-306.ibm.com/software/awdtools/suite
20. Oracle Development Tools. [žiūrėta 2005-01-10]. Prieiga per internetą: http://www.oracle.com/tools/index.html
21. Visio 2003 Product Information: The Microsoft Office business and technical diagramming program. [žiūrėta 2005-01-10]. Prieiga per internetą: http://office.microsoft.com/visio/

Priedai

1 priedas.. 58
2 priedas.. 66

1 priedas

ĮVEDAMOS IR IŠVEDAMOS INFORMACIJOS SPECIFIKAVIMO KONCEPTUALIZAVIMAS

Vidas Trijonis, Rimantas Butleris

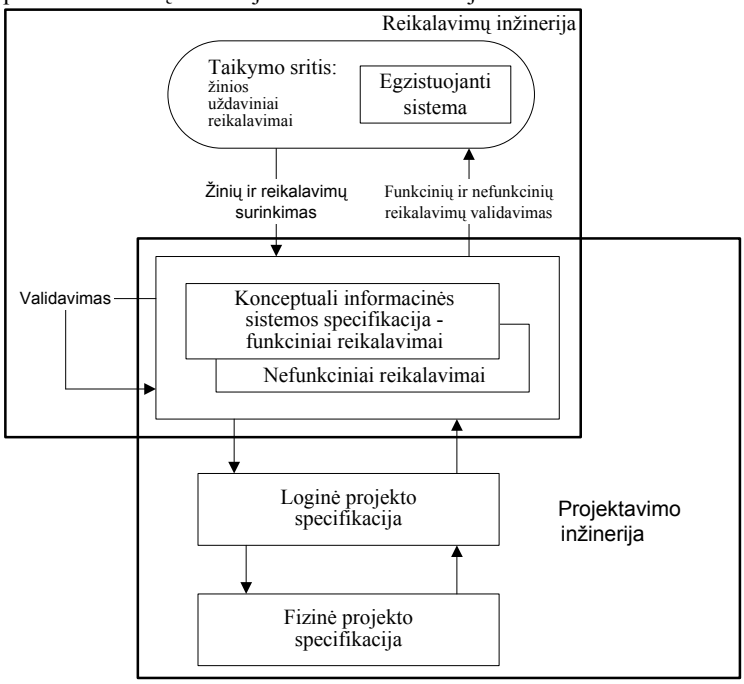
Kauno Technologijos Universitetas

Studentų 50-308, 51368 Kaunas

Pateikiamas funkcinių reikalavimų specifikavimo metodas veiklai modeliuoti. Metodas apibrėžia vartotojo reikalavimus kuriamos sistemos įvedamos ir išvedamos informacijos struktūrai. Įvedamos ir išvedamos informacijos sudėtis tradiciškai pateikiama tam tikrų formų pagrindu. Kuriama sistema suteikia galimybę iš pateiktų vartotojo reikalavimų sudaryti duomenų koncepcinę schemą Metodas sudarytas iš trijų sluoksnių: 1) vartotojo įvedamos informacijos, 2) duomenų struktūros analizės 3) koncepcinės schemos. Sudarant antrą sluoksnį atliekama duomenų analizė: atskiriami atributai, esybės, ryšiai tarp jų.

1 Įvadas

Informacijos sistemos projektavimas pradedamas procesu, apimančiu vartotojo reikalavimų surinkimą ir jų specifikavimą. Informacijos sistemos gyvavimo ciklo etapai pavaizduoti 1 paveiksle [10]. Informacijos sistemoje galima išskirti du gyvavimo ciklo etapus: reikalavimų inžinerija ir sistemos inžinerija.



1 pav. Informacinės sistemos gyvavimo ciklo schema.

Reikalavimų inžinerijos etape vartotojas pateikia reikalavimus. Jis kelia reikalavimus įvedamai ir išvedamai informacijai. Įvedamos informacijos struktūra priklauso nuo to, kokią informacijos struktūrą vartotojas pageidauja gauti sistemos išeigoje. Vartotojas pateikia reikalavimus natūralia kalba, dokumentų formomis, naudojamų informacijos sistemų funkcinių komponentų savybėmis.

Šiame straipsnyje pateikta trumpa žinomų metodologijų informaciniams srautams specifikuoti apžvalga jų naudojamų duomenų šaltinių bei formuojamų modelių aspektais, o taip pat pristatytas modelis, leidžiantis konceptualiai aprašyti vartotojo reikalavimuose bei kitoje dalykinės srities analizės medžiagoje pateiktą įvedamos ir išvedamos informacijos struktūrą bei jai teikiamus apribojimus. Pasiūlytas modelis adekvatus tipiniam sistemos analizės ir jos rezultatų specifikavimo procesui pradiniuose IS gyvavimo ciklo etapuose ir tai sumažina supratimo atotrūkį tarp vartotojo ir analitiko vykdomos veiklos. Tradiciniuose metoduose duomenų struktūros specifikaciją atitinkantys esybių-ryšių (naudojamų SSADM metode [1]) ar klasių (naudojamuose objektiškai orientuotame (OO) projektavime [2, 12]) modeliai yra formuojami dalykinę sritį sudarančių esybių ar objektų aibių analizės pagrindu, kai tuo tarpu šiuo atveju, duomenų struktūroms modeliuoti panaudojama jau nusistovėjusių arba numatomų panaudoti informacinių srautų specifikacija.

2 Apdorojamos informacijos specifikavimo metodų analizė

Taikant žinomas metodologijas informacinei sistemai kurti, procesų eiga dažniausiai orientuota į sistemų analitiką. Daugumoje jų viena pirmųjų veiklų yra probleminės srities supratimas ir vartotojų reikalavimų supratimas, kuris įgyjamas atliekant reikalavimų inžineriją. Tačiau taikant jas, informacinei sistemai keliamų reikalavimų specifikacijos sudarymo eiga nėra natūrali. Pavyzdžiui, duomenų modelio elementai į specifikaciją įtraukiami anksčiau, nei išvedamos informacijos vieneto (pavyzdžiui, ataskaitos) elementai, kurie motyvuoja duomenų modelio elementų atsiradimą specifikacijoje. Taikant tokias metodologijas, sukurti modeliai nėra suprantami vartotojui, kadangi jie tiesiogiai neišvedami iš jo teikiamų reikalavimų sistemos išvedamai bei į ją įvedamai informacijai. Tuomet kyla klausimas: ar teisingai analitikas suspecifikavimo vartotojo keliamus reikalavimus, nes vartotojo dalyvavimas specifikacijos validavimo procese yra apsunkintas

1 lentelėje pateikiamos sukurtos metodologijos, jų duomenų šaltiniai ir formuojami modeliai[9]. Matome, kad dalis pristatomų metodologijų duomenų šaltiniais laiko dokumentų formas. Taikant tradicinius sistemų kūrimo metodus, vartotojo pateikti reikalavimai užrašomi natūralia kalba ir specifikuojami formalizuotai. O perėjimas nuo dalinai struktūrizuotos specifikacijos prie formalizuotos yra paprastesnis, nei perėjimas nuo natūralios kalbos pagrindu aprašytos informacinių srautų sudėties. Egzistuojantys automatizuoti dokumentų formų analizės metodai parodė [14, 6], kaip dalinai struktūrizuota informacija dokumentų formų pavyzdžių pavidale gali būti panaudota kuriamos sistemos koncepcinei schemai išgauti. Tačiau šie metodai nėra plačiai naudojami, nes paruošiamieji darbai atima daug laiko, o visiška garantija, jog gauta specifikacija yra teisinga, negaunama. Tradiciniai sistemų kūrimo metodai siūlo analizuoti dokumentų formas reikalavimams kompiuterizuotos IS statiniam duomenų modeliui išgauti, tačiau formalizuotų priemonių šiam tikslui jie neturi.

1 lentelė. Projektavimo inžinerija.

Duomenų šaltiniai	Modelio šaltiniai	Kuriamas modelis	Pagrindinės charakteristikos	Metodo pavadinimas	Autoriai
Formos	Duomenys	ER modelis	Formos modelio pateikimas Ekspertinių DB projektavimo sistemų kūrimas		Choobineh ir kiti
			Verslo dokumentų formų naudojimas Pokalbis su vartotoju		Kim ir kiti
			EER naudojimas terminų žodynui		Batini ir kiti
	Duomenys	Formų specifikacija, paremta FORMAL	Labai aukšto lygio paslaugų teikimas su didelėmis galimybėmis kompiuterizuoti įvairiausius duomenis apdorojančias veiklas		Shu ir kiti
	Biznio procesai	EPC modelis	BPR procesų modeliavimo metodo pateikimas: Palaiko tik informaciją apdorojančias veiklas	EPRE[8]	Kim ir kiti
Natūrali kalba	Duomenys	ER modelis	Pirmoji konceptualaus duomenų modeliavimo metodika		Chen
		EDFD	ERD ir DSD sujungimas		Bailin
	Objektai	Objektų, dinaminis, funkcinis modeliai	Paremta leksinės analizės metodu		Booch

Duomenų šaltiniai	Modelio šaltiniai	Kuriamas modelis	Pagrindinės charakteristikos	Metodo pavadinimas	Autoriai
Esamų sistemų šaltiniai	Duomenys ir procesai	Duomenų srautų modelis	Struktūrinio projektavimo metodologija	SSADM	Yourdon
		Duomenų, procesų modeliai, duomenų/procesų matrica	Programinės įrangos projektavimo metodologija: Įtraukia informacijos strateginį planavimą į struktūrinį metodą		Martin
	Objektai	NER ir UPM	Pateikia eksperimentinį objektų modelį	MOODD	Silva
		Objektų modelis	Formalus OO analizės ir projektavimo modelis	OMT UML	Rumbaugh ir kiti Quatrani

Kauno technologijos universiteto informacijos sistemos katedroje siūlomas funkcinių reikalavimų specifikavimo metodas siūlomas reikalavimų specifikavimo procesas imituoja natūralią reikalavimų analizės eigą [3, 4]. Šis procesas prasideda nuo kuriamos kompiuterizuotos informacijos sistemos (KIS) konteksto apibrėžimo ir reikalavimų įvedamos ir išvedamos informacijos specifikavimo [5].

Išvedamoji informacija yra pagrindinis dalykas, kurio tikisi KIS vartotojas. Todėl KIS išvedamos informacijos tipas bei struktūra yra pagrindinis vartotojo funkcinis reikalavimas, nuo kurio priklauso kiti kuriamai sistemai keliami funkciniai reikalavimai. Pateikdamas reikalavimus KIS išvedamai informacijai, vartotojas juos pateikia neformaliai, operuodamas tokiomis sąvokomis kaip ataskaita, suvestinė, sąrašas. Analitikas tokia forma pateiktą informaciją formalizuoja, paversdamas juos esybėmis, klasėmis, konceptais, asociacijomis. Vartotojo pateikti reikalavimai natūralia kalba nėra apibrėžti vienareikšmiškai, todėl analitikas juos gali suprasti neteisingai. Analitiko sudaryta koncepcinė schema nėra suprantama vartotojui, todėl vartotojas negali patikrinti jos pilnumo ir teisingumo. Tokia formalizavimo eiga turi daug privalumų, tačiau dėl jo tarp vartotojo ir analitiko atsiranda atotrūkis, apsunkinantis jų bendravimą.

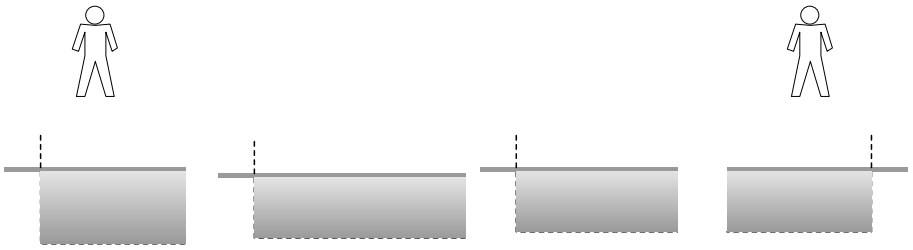
Įvedama informacija – tai organizacijos objektai, saugantys informaciją apie organizacijos veiklą: įvairūs dokumentai, žodiniai pranešimai, egzistuojančių kompiuterizuotų sistemų ekraninės formos, elektroniniu būdu perduodami duomenų paketai ir pan.

Išvedama informacija – tai įvairios ataskaitos, rodikliai, kiti duomenų srautai, kuriuos vartotojui turi išvesti kuriama sistema. Vartotojui juos apibrėžti nesunku, nes jis dažniausiai žino, kokios informacijos jam reikia kasdieninėje organizacijos veikloje. Išvedama informacija apibendrina, pateikia tam tikru aspektu įvedamą informaciją.

Įvedamos ir išvedamos informacijos struktūra panaši: formos, ataskaitos, ekraninės užsklandos, todėl specifikuojant vartotojo reikalavimus galima remtis tuo pačiu modeliu.

3 Įvedamos ir išvedamos informacijos specifikavimo konceptualus modelis

KIS funkcionalumo įvedamai ir išvedamai informacijai specifikuoti naudojamas trijų etapų metodas: vartotojo informacijos suvedimo, duomenų analizės, koncepcinės schemos fragmento sudarymo. Modelio taikymo metu apdorojamų duomenų būsenos bei gaunamų rezultatų sudėtis, kartu atspindintys ir modelio sudarymo procesą, pavaizduoti 2 paveiksle.



2 pav. Reikalavimų specifikavimo procesai.

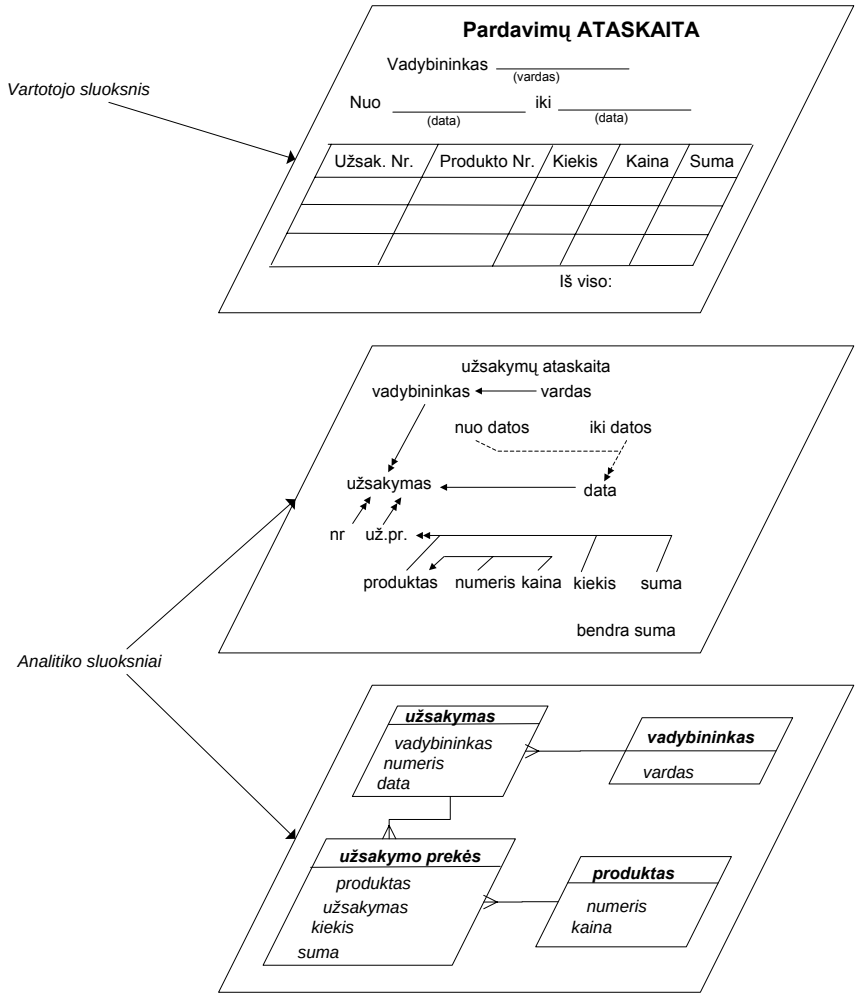
Pirmasis etapas skirtas vartotojo pateiktos informacijos suvedimui. Duomenys suvedami vartotojo pateikta forma tam, kad vartotojas suprastų ir galėtų pats patikrinti modelio teisingumą ir pilnumą. Antrajame etape analitikas perkelia visus duomenų struktūros objektus, dažniausiai formų laukus, į analitinį sluoksnį. Šiame sluoksnyje kiekvienam objektui

analitikas priskiria tipą. Trečias etapas – koncepcinės schemos generavimas iš analitiko suformuotos atributų, jų grupių, sudarytų ryšių tarp atributų ir sudarytų ryšių tarp atributų grupių aibės.

Pristatomo modelio architektūroje išsiskiriami trys sluoksniai, kurie pavaizduoti 3 paveiksle.

Trijų sluoksnių modelis turi trejopą tikslą:

- Vartotojo sluoksnis leidžia lengviau susišnekėti vartotojui ir analitikui, nes vartotojas mato tai, ką jis ir tikisi gauti iš kuriamos KIS (pavyzdžiui, kada kalbama apie KIS funkcionalumo rezultatus). Be to naudojamos dokumentų formos vartotojui yra gerai pažįstamos, todėl jam lengviau perteikti savo žinias analitikui. Projektuotojui belieka rasti būdą kaip teisingai perteikti savo sudaryto modelio semantiką vartotojui. Tuo būdu vyksta abipusis apsimokymas: inžinierius gilinasi į dalykinę sritį, o vartotojas į formalų informacijos aprašymą.
- Toks modelio suskirstymas į sluoksnius, leidžia specifikuoti kiekvieno struktūros elemento kilmę, t.y., kiekvienas struktūros elementas projektuotojo sluoksnyje gali būti motyvuotas atitinkamu elementu iš vartotojo sluoksnio.
- Sugeneruota koncepcinė schema yra duomenų bazės modelio fragmentas, kurių visuma sudaro kuriamos sistemos duomenų bazių modelį. Kuriamoje sistemoje yra galimybė duomenų bazės fragmentą perkelti į duomenų bazę, naudojant ODBC sąsają.



3 pav. Reikalavimų specifikacijos sluoksniai.

3.1 Informacijos struktūrizavimo etapas

Visa informacija apie pirminę duomenų formą yra surenkama šioje fazėje. Formos skirstomos į elektronines formas ir dokumentų formas. Elektroninės formos naudojamos liktinių sistemų reinžinerijoje.

Kuriamoje metodologijoje remiamasi Choobineh ir kitų pasiūlytu apibrėžimu, kad forma yra struktūrizuotų kintamųjų (t.y. formos laukų), kurie atitinkamai sutvarkyti duomenų įvedimui ir vaizdavimui, rinkinys [7].

Formos dažniausiai naudojamos standartizuoti ir suprastinti įvairios informacijos, tokios kaip užsakymo blankas, mokesčių forma, pirkimo sąskaita-faktūra ar tiekimo paraiškos forma, apdorojimą, todėl verta atkreipti dėmesį į jas, kaip įeinančios informacijos srautą [11].

Dalį duomenų vartotojas pateikia nestruktūrizuotus. Priklausomai nuo kompiuterizuojamos veiklos nusistovėjimo, įvardinamas skirtingas vartotojo pateikiamos struktūrizuotos ir nestruktūrizuotos informacijos santykis. Vartotojui pateikiant nestruktūrizuotą informaciją, ją reikia struktūrizuoti. Tokiu būdu lengviausia išskirti atributus ir atributų grupes sekančioje fazėje.

Šiame etape specifikacijos saugykloje užregistruojamas formos šablonas, kuris artimas vartotojui. Pateikiant vartotojui nestruktūrizuotus duomenis, analitikas privalo išsiaiškinti duomenų srauto struktūrą ir ją suprantamai pateikti vartotojui.

3.2 Duomenų struktūros analizė

Šiame etape operuojama visais duomenų struktūros objektais, dažniausiai suformuotais iš vartotojo pateiktų formų ir ataskaitų.

Duomenų struktūros specifikacija susideda iš duomenų struktūros pavadinimo ir objektų aibės. Pavadinimas reiškia antraštės lauką, kuris nusako duomenų struktūros semantinę reikšmę. Dažniausiai pateikiama duomenų struktūra formomis arba ataskaitomis.

Analitikas, remdamasis savo žiniomis bei vartotojo pagalba, perkeltiems objektams iš vartotojo sluoksnio priskiria tipą, nurodo apribojimus, objektų tarpusavio ryšius. Kiekvienas vartotojo sluoksnio elementas yra tiesiogiai susiejamas su atitinkamu duomenų struktūros objektu. Yra šeši duomenų struktūros objektų ar jų grupių tipai: tekstas, atributas, atributų grupė, apribojimai, duomenų šaltinio tipas ir duomenų šaltinio potipis.

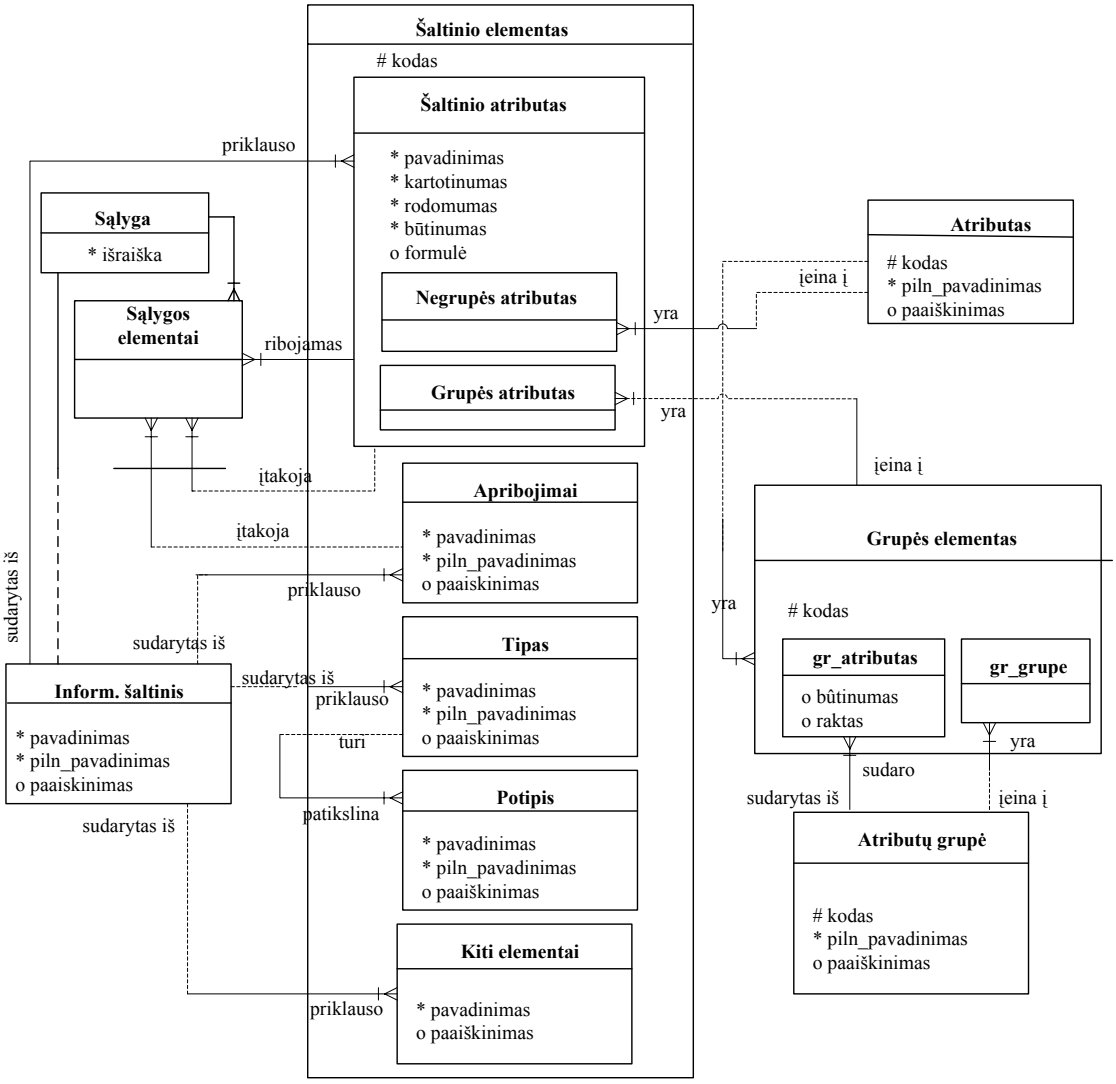
3.3 Koncepcinės schemos generavimas

Struktūros modelyje suformavus struktūrą, t.y. duomenų struktūros objektams priskyrus tipą, įvedus apribojimą, sujungus atributus ir jų grupes, galima generuoti koncepcinį modelį. Koncepcinis modelis yra kuriamos sistemos duomenų bazių modelio fragmentas. Koncepcinis modelis generuojamas išrenkant atributų grupes, juos sudarančius atributus, ir tarpusavio ryšius. Analitikas gali patikrinti sugeneruoto esybių-ryšių modelio, vartotojo pateiktos duomenų struktūros, teisingumą ir pilnumą. Teisingumo tikrinimui palengvinti esybes galima išdėstyti pagal Oracle siūlomą dėstymą [1].

Sukurta metodologija integruota į Microsoft Visio paketo duomenų bazių modelio šabloną. Tokiu būdu nereikia sukurti sąsajos tarp kuriamos programos ir duomenų bazių serverio, sumažėja klaidų tikimybė. Microsoft Visio duomenų bazių modelio šablonas turi integruotą ODBC sąsają su duomenų bazėmis[13].

4 Meta modelio koncepcinė schema

Meta-modelio koncepcinė schema, kuriamos KIS funkcionalumo rezultatų ir duomenų šaltinių struktūrai specifikuoti analitiko sluoksnyje esybių-ryšių modelio notacijoje pateikta 4 paveiksle.



4 pav. Informacijos struktūros meta modelio koncepcinė schema.

Informacijos šaltinis – tai esybė, atitinkanti tam tikrą duomenų šaltinį arba KIS funkcionalumo rezultatą.

Atributas – tai kiekybinė arba kokybinė tam tikro organizacijos objekto, apie kurį turi būti saugoma informacija, savybė. Taip pat gali būti savybė, kuri klasifikuoja arba identifikuoja šiuos organizacijos objektus. Atributas įtraukiamas į KIS specifikaciją, jei jis yra įtrauktas bent į vieną rezultatą ar duomenų šaltinį (t.y. aprašo jo struktūrą).

Atributų grupė – tai atributai ar jų grupės, aprašančios tą patį objektą. Skirtingiems objektams gali būti būdingos tos pačios savybės. Taigi, tas pats atributas ar jų grupė gali būti įtrauktas į kelias skirtingas atributų grupes. Atributų grupė gali būti specifikuota, jei specifikuojami ją sudarantys atributai ar atributų grupės.

Šaltinio elementas – tai informacijos šaltinio (duomenų šaltinio ar KIS funkcionalumo rezultato) struktūrą aprašantys elementai.

Šaltinio atributas – tai tam tikro rezultato ar duomenų šaltinio struktūros elementas, kurio reikšmės bus įvedamos, saugomos, apdorojamos, ar išvedamos kuriamojoje KIS. Vieni atributai aprašo tik rezultatą ar duomenų šaltinį (kurie taip pat yra organizacijos objektai). Šie atributai bus vadinami neįtrauktais į atributų grupę. Kiti atributai aprašo kitų (ne duomenų šaltinių ar rezultatų) objektų savybes, kurios yra pateikiamos tam tikruose duomenų šaltiniuose ar rezultatuose. Šie atributai bus vadinami įtrauktais į atributų grupes.

Tipas – tai šaltinio struktūros elementas, identifikuojantis šaltinio tipą. Šaltinio tipas identifikuoja tik vieną rezultatą. Kiekvienam šaltiniui gali būti specifikuotas tik vienas tipas. Šaltinio tipu gali būti jo pavadinimas, pavyzdžiui, „Užsakymų suvestinė“.

Potipis – tai šaltinio struktūros elementas, identifikuojantis šaltinio tipo potipį. Pavyzdžiui, tipo “Užsakymų suvestinė“ potipiu gali būti „metinė“ suvestinė. Šaltinio tipo potipis identifikuoja tik vieną šaltinį. Kiekvienam šaltiniui gali būti specifikuoti keli tipo potipiai.

Ribojantis elementas – tai šaltinio atributų reikšmes ribojantis elementas, kuris yra tam tikra atributo reikšmė. Pavyzdžiui, „skubus“ yra atributo „Užsakymo tipas“ reikšmė. Kiekvienam šaltiniui gali būti specifikuota keletas ribojančių elementų.

Kiti elementai (neklasifikuoti elementai) - tai šaltinio struktūros elementas, kuris nėra atributas, nei tipas, nei potipis, nei ribojantis elementas. Kiekvienam šaltiniui gali būti specifikuota keletas tokių elementų. Tokiais elementais gali būti kiekvienas šaltinio struktūros elementas, jei jis nesuklasifikuojamas į bent vieną iš ankščiau aprašytų struktūros elementų tipų.

Sąlyga – tai sąlyga, kurios reikia laikytis formuojant tam tikrų rezultatą ar duomenų šaltinį. Dažnai, formuojant tam tikrą rezultatą, uždedami apribojimai rezultatų atributo reikšmėms (pavyzdžiui, įvairūs laiko intervalai). Kas ir kaip įtakoja išvedamų rezultato atributų reikšmes, nurodoma sąlygos išraiškoje. Jei šaltiniui yra specifikuotas bent vienas ribojantis elementas, vadinasi šaltiniui turi būti specifikuota sąlyga. Pavyzdžiui, skubiam užsakymui suformuoti, turi būti patenkinta tokia sąlyga „Užsakymo tipas = ‚skubus' ". Koks tai šaltinio elementas ir kokį kitą elementą įtakoja, yra nurodyta specifikuojant sąlygos elementus.

Sąlygos elementas – tai elementas, parodantis koks šaltinio elementas (atributas ar ribojantis elementas) įtakoja kitą šaltinio elementą (atributą). Sąlygą turi sudaryti bent vienas elementas. Pavyzdžiui, pora „Užsakymo tipas" ir „skubus" yra sąlygos elementas.

5 Išvados ir tolimesni darbai

Atlikus esamų metodų analizę, pastebėta, kad pradiniame reikalavimų etape orientuojamasi į reikalavimų specifikavimą. Pastebėta, kad specifikuojant reikalavimus tradiciniais metodais išlieka ženklus supratimo atotrūkis tarp vartotojo ir analitiko. Didžiausias trūkumas, kad vartotojas pats negali dalyvauti reikalavimų specifikacijos pilnumo ir teisingumo tikrinime. Pateiktame modelyje sudaromo koncepcinio modelio pilnumo ir teisingumo tikrinime dalyvauja pats vartotojas.

Pateiktame modelyje procesai parodo sklandų perėjimą nuo vartotojo pateiktų reikalavimų iki koncepcinės schemos. Sudarytoje kuriamo modelio metaduomenų bazėje saugomi analitiko sluoksnių duomenys.

Ateityje planuojama sukurti CASE įrankio prototipą, kuris suteiktų galimybę įvedamos ir išvedamos informacijos koncepcinės schemos pagrindu, generuoti dalykinės srities duomenų bazių modelį.

Literatūros sąrašas

1. **R. Barker.** CASE*METHOD: Entity Relationship Modelling. *Addison – Wesley Publ. Co., New York*, 1990.
2. **G. Booch.** Object-Oriented Analysis and Design with Applications (2nd Edition). *Addison-Wesley Publ. Co., New York*, 1993
3. **R. Butkienė, R. Butleris.** Extending functionality of CASE tools to support requirements engineering. *Computer Science Reports: Emerging Database Research in East Europe: proceedings of the pre-conference workshop of VLDB 2003.* Brandenburg, 2003, no. 14, p. 12-16.
4. **R. Butkienė, R. Butleris.** The approach for user requirements specification. *Advances in Databases and Information Systems: 5th East-European Conference ADBIS'2001: research communications.* Vilnius, 2001, Vol. 1, pp. 225-240.
5. **R. Butkienė, V. Savickas.** Reikalavimų KIS įvedamai ir išvedamai informacijai specifikavimas. *Informacinės Technologijos*, 2002, 401-407 psl.
6. **J. Chobineh, M. Mannino.** An Expert Database Design System Based on Analysis of Forms. *IEEE transactions on software engineering*, Vol.14, 1998.
7. **J. Choobineh, M. Mannino, V. Tseng.** A form-based approach for database analysis and desigh. *Communications of the ACM*, 1992, Vol.35, No 2, pp.108-120.
8. **K. -H. Kim, Y. -G. Kim.** Process reverse engineering for BPR: A form-based approach. *Information & Management*, 1998, Vol 33, 187-200 psl.
9. **H. Lee, Ch. Yoo.** A Form Driven Object-Oriented Reverse Engineering Methodology. *Information System*, 2000 Vol. 25, No 3, pp. 235-259 .
10. **P. Loucopoulos , V. Karakostas.** System Requirements Engineering. *McGraw-Hill*, 1995.
11. **M. Mannino, V. P. Tseng.** Inferring database requirements from examples in forms. *Seventh International Conference on Entity – Relationship approach*, Rome, 1988, pp. 1-25.
12. **J. R. Rumbaugh.** Object – Oriented Modelling and Design. *Prentice Hall*, 1990.
13. **M. H. Walker, N. Eaton.** Microsoft Office Visio 2003 Inside Out. *Microsoft Press*, 2003, pp. 912
14. **B. Wangler.** Contributions of Function Requirements Modelling. Doctoral Thesis. *Stockholm University*, 1993,189–230.

The conceptualization of specification of incoming and out coming information specification

Functional requirements specification method in earliest stages of development is presented. This method defines user requirements for structure of incoming and out coming data. Most user incoming and out coming information comes in form or report base, which serves as information source. This method creates conceptual schema from user requirements. It's divided into three phases: 1) user incoming and out coming information, 2) data structure analysis, 3) conceptual schema generation.

2 priedas

Metaduomenų saugyklos aprašas

Vardas	*Duomenų tipas*	*Pirminis raktas*	*Būtinas*	*Unikalus*	*Paaiškinimas*
Atributas					Lentelėje saugomi visi nagrinėjamo organizacijos veiklos konteksto atributai.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus atributą apibudinantis kodas, kodą sudaro sistemos vartotojas (analitikas/projektuotojas).
n_vardas	varchar 50		+		Atributo vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
unikalumas	bool		+		Požymis rodantis ar atributo reikšmė turi būti unikali. TRUE – reikšmė turi būti unikali, o FALSE - kad reikšmės unikalumas nėra būtinas.
paaiskinimas	varchar 200				Detali informacija apie atributą.
tipas	varchar 30				Atributo duomenų tipas.
e_id	int				Esybės, kuriai priklauso atributas id.
Esybe					Lentelėje saugomos visos nagrinėjamo organizacijos veiklos konteksto esybės.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus esybę apibudinantis kodas, kodą sudaro sistemos vartotojas.
n_vardas	varchar 30		+		Esybės vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
paaiskinimas	varchar 200				Detalizuota informacija apie esybę.
Rysys					Lentelėje saugomi visi nagrinėjamo organizacijos veiklos konteksto ryšiai tarp esybių.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kard1	varchar 3		+		Ryšio kardinalumas viename ryšio gale. Kardinalumo aprašymo struktūra: <būtinumas>,<kardinalumas>. Būtinumas gali įgyti šias reikšmes: 0-nebūtinas, 1- būtinas. Kardinalumas gali įgyti šias reikšmes: 1 – vienas, x – daug.
kard2	varchar 3		+		Ryšio kardinalumas antrame ryšio gale. Kardinalumo aprašymo struktūra: <būtinumas>,<kardinalumas>. Būtinumas gali įgyti šias reikšmes: 0-nebūtinas, 1- būtinas. Kardinalumas gali įgyti šias reikšmes: 1 – vienas, x – daug.
e_id1	int		+		Esybių, kurias jungia ryšys id.
e_id2	int		+		
rys_ds_id	int				Motyvuojančio ryšio tarp duomenų šaltinių id.
Rezultatas					Lentelėje saugomi informacija apie KIS funkcionalumo metu formuojamus rezultatus arba žodžiu perduodamus informacijos srautus.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
pavadinimas	varchar 30		+		Rezultato pavadinimas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
paaiskinimas	varchar 200				Detalizuota informacija apie rezultatą.
salyga	varchar 500				Užrašoma žodinė informacija apie sąlygas, kurias

Vardas	*Duomenų tipas*	*Pirminis raktas*	*Būtinas*	*Unikalus*	*Paaiškinimas*
					reikia tenkinti formuojant tam tikrą rezultatą.
R_atributas					Lentelėje saugomi informacija apie KIS funkcionalumo metu formuojamų rezultatų atributus.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
vard_org	varchar 50		+		Rezultato atributo pavadinimas, kuris paimtas iš nagrinėjamo veiklos konteksto, toks koks pateikiamas originaliame rezultate.
egz_sk	int		+		Konkretaus atributo egzempliorių skaičius išvedamų į rezultatą.
rodomumas	bool		+		Ar atributas matomas tik specifikacijos modelyje ar ir originale. FALSE – originale ir modelyje, TRUE - tik modelyje.
butinumas	bool		+		Požymis rodantis ar atributo reikšmės būtinai turi būti pateikiama rezultate. TRUE – reikšmė turi būti pateikiama, o FALSE - kad reikšmė nebūtina.
formule	varchar 200				Jei atributo reikšmė yra skaičiuojama, išsaugoma formulė, pagal kuria gali būti skaičiuojam atributo reikšmė.
r_id	int		+		Rezultato id, kuriam priklauso atributas.
a_id	int		+		Sistemos atributo id, su kuriuo siejamas rezultato atributas, nes skirtinguose rezultatuose ar duomenų šaltiniuose tas pats sistemos atributas gali būti skirtingai įvardintas.
Rib_reiksme_R					Lentelėje saugoma rezultate esančių atributų egzempliorių ribinių reikšmių informacija.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
vard_org	varchar 50		+		Reikšmė, kuris paimtas iš nagrinėjamo veiklos konteksto.
n_vardas	varchar 30		+		Reikšmė, kuria KIS siūlo naudoti vartotojas, tai gali būti ir ta pati reikšmė paimta iš nagrinėjamo veiklos konteksto
paaiskinimas	varchar 200				Detalizuota informacija apie reikšmę.
r_id	int		+		Rezultato id, kuriame naudojama ši reikšmė.
R_esybe					Lentelėje saugomi informacija apie KIS funkcionalumo metu formuojamų rezultatų išskirtas esybes.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
vard_org	varchar 50		+		Esybės pavadinimas, kuris paimtas iš nagrinėjamo veiklos konteksto.
rodomumas	bool		+		Ar esybė matoma tik specifikacijos modelyje ar ir originale. FALSE – originale ir modelyje, TRUE - tik modelyje.
tipas	varchar 30		+		
r_id	int		+		Rezultato id, kuriam priklauso esybė.
e_id	int		+		Sistemos esybės id, su kuria siejama rezultato esybė, nes skirtinguose rezultatuose ar duomenų šaltiniuose ta pati sistemos esybė gali būti skirtingai įvardinta.
Salygos_elem_R					Lentelėje saugoma informacija apie rezultato sudarymo sąlygos elementus.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
r_id	int		+		Rezultato, kuriam formuojam sąlyga id.

Vardas	*Duomenų tipas*	*Pirminis raktas*	*Būtinas*	*Unikalus*	*Paaiškinimas*
r_a_id1	int		+		Rezultato atributo id, kuris įtakojamas sąlygos.
r_a_id2	int				Rezultato atributo id, kuris įtakoja rezultato atributą.
rb_id	int				Rezultato ribinės reikšmės id, kuri įtakoja rezultato atributą.
Duom_salt					Lentelėje saugomi informacija apie organizacijos objektus (duomenų šaltinius) saugančius duomenis, reikalingus funkcijoms įvykdyti.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
pavadinimas	varchar 30		+		Duomenų šaltinio pavadinimas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
paaiskinimas	varchar 200				Detalizuota informacija apie duomenų šaltinį.
salyga	varchar 500				Užrašoma žodinė informacija apie sąlygas ir logiką, kurias reikia tenkinti formuojant duomenų šaltinį.
h_lygis	int				Duomenų šaltinio hierarchijos lygis.
DS_atributas					Lentelėje saugomi informacija apie organizacijos objektų (duomenų šaltinius) saugančių duomenis, reikalingus funkcijoms įvykdyti, atributus.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
vard_org	varchar 50		+		Duomenų šaltinio atributo pavadinimas, kuris paimtas iš nagrinėjamo veiklos konteksto.
egz_sk	int		+		Konkretaus atributo egzempliorių skaičius įvedamų į konkretų duomenų šaltinį.
rodomumas	bool		+		Ar atributas matomas tik specifikacijos modelyje ar ir originale. FALSE – originale ir modelyje, TRUE - tik modelyje.
butinumas	bool		+		Požymis rodantis ar atributo reikšmės būtinai turi būti įvedama į duomenų šaltinį. TRUE – reikšmė turi būti pateikiama, o FALSE - kad reikšmė nebūtina.
formule	varchar 200				Jei atributo reikšmė yra skaičiuojama, išsaugoma formulė, pagal kuria gali būti skaičiuojam atributo reikšmė.
ds_id	int		+		Duomenų šaltinio id, kuriam priklauso atributas.
a_id	int		+		Sistemos atributo id, su kuriuo siejamas duomenų šaltinio atributas, nes skirtinguose rezultatuose ar duomenų šaltiniuose tas pats sistemos atributas gali būti skirtingai įvardintas.
Rib_reiksme_DS					Lentelėje saugoma duomenų šaltinyje esančių atributų egzempliorių ribinių reikšmių informacija.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
vard_org	varchar 50		+		Reikšmė, kuris paimtas iš nagrinėjamo veiklos konteksto.
n_vardas	varchar 30		+		Reikšmė, kuria KIS siūlo naudoti vartotojas, tai gali būti ir ta pati reikšmė paimta iš nagrinėjamo veiklos konteksto
paaiskinimas	varchar 200				Detalizuota informacija apie reikšmę.
ds_id	id		+		duomenų šaltinio id, kuriame naudojama ši reikšmė.

Vardas	*Duomenų tipas*	*Pirminis raktas*	*Būtinas*	*Unikalus*	*Paaiškinimas*
DS_esybe					Lentelėje saugomi informacija apie esybes išskirtas iš organizacijos objektų saugančius duomenis, reikalingus funkcijoms įvykdyti.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
vard_org	varchar 50		+		Esybės pavadinimas, kuris paimtas iš nagrinėjamo veiklos konteksto.
rodomumas	bool		+		Ar esybė matoma tik specifikacijos modelyje ar ir originale. FALSE – originale ir modelyje, TRUE - tik modelyje.
tipas	varchar 30		+		
ds_id	int		+		Duomenų šaltinio id, kuriam priklauso esybė.
e_id	int		+		Sistemos esybės id, su kuria siejama duomenų šaltinio esybė, nes skirtinguose rezultatuose ar duomenų šaltiniuose ta pati sistemos esybė gali būti skirtingai įvardinta.
Salygos_elem_DS					Lentelėje saugoma informacija apie duomenų šaltinio sudarymo sąlygos elementus.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
ds_id	int		+		Duomenų šaltinio, kuriam formuojam sąlyga id.
ds_a_id1	int		+		Duomenų šaltinio atributo id, kuris įtakojamas sąlygos.
ds_a_id2	int				Duomenų šaltinio atributo id, kuris įtakoja duomenų šaltinio atributą.
rb_id	int				Duomenų šaltinio ribinės reikšmės id, kuri įtakoja rezultato atributą.
Srautas_R					Lentelėje saugoma informacija apie informacijos srautą tarp duomenų šaltinio ir rezultato.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
pavadinimas	varchar 30		+		Vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
paaiskinimas	varchar 200				Detali informacija apie srautą.
ds_id_is	int		+		Duomenų šaltinio id, iš kurio išeina srautas.
r_id_i	int		+		Rezultato id į kurį įeina srautas.
Srautas_DS					Lentelėje saugoma informacija apie informacijos srautą tarp duomenų šaltinio ir duomenų šaltinio.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
pavadinimas	varchar 30		+		Vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
paaiskinimas	varchar 200				Detali informacija apie srautą.
ds_id_is	int		+		Duomenų šaltinio id, iš kurio išeina srautas.
ds_id_i	int		+		Duomenų šaltinio id į kurį įeina srautas.
Sr_R_elem					Lentelėje saugoma informacija apie informacijos srauto tarp duomenų šaltinio ir rezultato sandarą.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
sr_id	int		+		Srauto id.
ds_a_id_is	int		+		Duomenų šaltinio iš kurio išeina srautas, atributo id.
r_a_id_i	int		+		Rezultato atributo id, į kurį pereina informacija iš duomenų šaltinio atributo.
Sr_DS_elem					Lentelėje saugoma informacija apie informacijos srauto tarp duomenų šaltinio ir duomenų šaltinio sandarą.

Vardas	*Duomenų tipas*	*Pirminis raktas*	*Būtinas*	*Unikalus*	*Paaiškinimas*
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
sr_id	int		+		Srauto id.
ds_a_id_is	int		+		Duomenų šaltinio iš kurio išeina srautas, atributo id.
ds_a_id_i	int		+		Duomenų šaltinio atributo id, į kurį pereina informacija iš duomenų šaltinio atributo.
Et_atributas					Lentelėje saugoma informacija apie duomenų šaltinių apdorojimo etapo metu įvedamus arba modifikuojamus atributus.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
ds_a_id	int		+		Duomenų šaltinio atributo id, kuris to etapo metu apdorojamas.
et_id	int		+		Etapo id.
Etapas					Lentelėje saugoma informacija apie duomenų šaltinio apdorojimo etapus.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
pavadinimas	varchar 30		+		Vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
paaiskinimas	varchar 200				Detali informacija apie etapą.
ds_id	int		+		Duomenų šaltinio id, kuriam priklauso etapas.
Perėjimas_Et					Lentelėje saugoma informacija apie perėjimus tarp duomenų šaltinio apdorojimo etapų.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
v_id	int		+	+	Veiksmo id, kurį vykdant atliekamas perėjimas.
et_is_id	int				Etapo id iš kurio įvyksta perėjimas
et_i_id	int				Etapo id į kurį įvyksta perėjimas
Veiksmas_DS					Lentelėje saugoma informacija apie visus galimus duomenų šaltinių apdorojimo veiksmus.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
pavadinimas	varchar 30		+		Vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
paaiskinimas	varchar 200				Detali informacija apie veiksmą.
ak_id1	int		+		Aktoriaus id, kuris yra agentas t.y. atlieka veiksmą arba jį inicijuoją.
ak_id2	int		+		Aktoriaus id, kuris yra veiksmo rezultatų (srauto) gavėjas.
Aktorius					Lentelėje saugoma informacija apie visus aktorius esančius nagrinėjamos veiklos kontekste.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
pavadinimas	varchar 30		+		Vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
paaiskinimas	varchar 200				Detali informacija apie aktorių.
DS_B_atributas					Lentelėje saugoma informacija apie duomenų šaltinių atributus įgyjančius reikšmes nurodytoje būsenoje.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
et_a_id	int		+		Šaltinio apdorojimo etapų atributų id.
ds_b_id	int		+		Duomenų šaltinio būsenos id.
DS_Busena					Lentelėje saugoma informacija apie duomenų šaltinių būsenas.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro

Vardas	*Duomenų tipas*	*Pirminis raktas*	*Būtinas*	*Unikalus*	*Paaiškinimas*
					sistemos vartotojas.
pavadinimas	varchar 30		+		Vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
paaiskinimas	varchar 200				Detali informacija apie aktorių.
et_id	int		+		Duomenų šaltinio apdorojimo etapo id, kurio metu šaltinis patenka į šia būseną.
ds_b_id	int				Duomenų šaltinio būsenos id, kuri yra tėvinė nagrinėjamos duomenų šaltinio būsenos atžvilgiu.
Perejimas_DS_B					Lentelėje saugoma informacija apie perėjimus tarp duomenų šaltinių būsenų.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
ds_b_is_id	int				Duomenų šaltinio id, iš kurio vyksta perėjimas.
ds_b_i_id	int				Duomenų šaltinio id, į kurį vyksta perėjimas.
et_id	int		+		Duomenų šaltinio apdorojimo etapo id, kurio metu įvyksta perėjimas tarp būsenų.
Lankas_DS_B					Lentelėje saugoma informacija apie lanką tarp duomenų šaltinių būsenų ryšių.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
tipas	varchar30		+		Lanko tipas. Galimos dvi reikšmės: eskliuzyvinis – galioja bent vienas lake esantis ryšys būtinai, visi kiti negalioja; neeskliuzyvinis – galioja bent vienas būtinai ir daugiau lanke esančių ryšių.
l_ds_r_id	int				Motyvuojančio lanko esančio ant duomenų šaltinių ryšių id
Lankas_DS_R					Lentelėje saugoma informacija apie lanką tarp duomenų šaltinių ryšių.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kodas	varchar 10		+	+	Unikalus įrašą apibudinantis kodas, kodą sudaro sistemos vartotojas.
tipas	varchar 30		+		Lanko tipas. Galimos dvi reikšmės: eskliuzyvinis – galioja bent vienas lake esantis ryšys būtinai, visi kiti negalioja; neeskliuzyvinis – galioja bent vienas būtinai ir daugiau lanke esančių ryšių.
Rysys_tarp_DS_B					Lentelėje saugoma informacija apie ryšius tarp duomenų šaltinių būsenų.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kard1	varchar 3		+		Ryšio kardinalumas viename ryšio gale. Kardinalumo aprašymo struktūra: <būtinumas>,<kardinalumas>. Būtinumas gali įgyti šias reikšmes: 0-nebūtinas, 1- būtinas. Kardinalumas gali įgyti šias reikšmes: 1 – vienas, x – daug.
kard2	varchar 3		+		Ryšio kardinalumas antrame ryšio gale. Kardinalumo aprašymo struktūra: <būtinumas>,<kardinalumas>. Būtinumas gali įgyti šias reikšmes: 0-nebūtinas, 1- būtinas. Kardinalumas gali įgyti šias reikšmes: 1 – vienas, x – daug.
vardas1	varchar 30		+		Vieno ryšio galo vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
vardas2	varchar 30		+		Kito ryšio galo vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
rys_ds_id	int		+		Motyvuojančio ryšio tarp duomenų šaltinių id.
l_ds_b_id1	int				Lanko tarp duomenų šaltinių būsenų ryšių id, kuriam priklauso pirmas ryšio galas.

Vardas	*Duomenų tipas*	*Pirminis raktas*	*Būtinas*	*Unikalus*	*Paaiškinimas*
l_ds_b_id2	int				Lanko tarp duomenų šaltinių būsenų ryšių id, kuriam priklauso antras ryšio galas.
ds_b_id1	int		+		Duomenų šaltinio būsenos id, iš kurio eina ryšys.
ds_b_id2	int		+		Duomenų šaltinio būsenos id, į kurį eina ryšys.
Rysys_tarp_DS					Lentelėje saugoma informacija apie ryšius tarp duomenų šaltinių.
id	int	+	+	+	Unikalus įrašo id, kuri sugeneruoja DBVS.
kard1	varchar 3		+		Ryšio kardinalumas viename ryšio gale. Kardinalumo aprašymo struktūra: <būtinumas>,<kardinalumas>. Būtinumas gali įgyti šias reikšmes: 0-nebūtinas, 1- būtinas. Kardinalumas gali įgyti šias reikšmes: 1 – vienas, x – daug.
kard2	varchar 3		+		Ryšio kardinalumas antrame ryšio gale. Kardinalumo aprašymo struktūra: <būtinumas>,<kardinalumas>. Būtinumas gali įgyti šias reikšmes: 0-nebūtinas, 1- būtinas. Kardinalumas gali įgyti šias reikšmes: 1 – vienas, x – daug.
vardas1	varchar 30		+		Vieno ryšio galo vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
vardas2	varchar 30		+		Kito ryšio galo vardas, kuris gali būti paimtas iš nagrinėjamo veiklos konteksto arba sukurtas paties vartotojo.
l_ds_r_id1	int				Lanko tarp duomenų šaltinių ryšių id, kuriam priklauso pirmas ryšio galas.
l_ds_r_id2	int				Lanko tarp duomenų šaltinių 100 ryšių id, kuriam priklauso antras ryšio galas.
ds_id1	int		+		Duomenų šaltinio id, iš kurio eina ryšys.
ds_id2	int		+		Duomenų šaltinio id, į kurį eina ryšys.