KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACINIŲ SISTEMŲ INŽINERIJOS STUDIJŲ PROGRAMA

PAULIUS STASIULIONIS

AUTOMATIZUOTAS DUOMENŲ BAZĖS SCHEMOS SUKŪRIMAS FORMŲ ANALIZĖS PAGRINDU

Magistro darbas

Darbo vadovas
doc. R. Butkienė

KAUNAS, 2013

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACINIŲ SISTEMŲ INŽINERIJOS STUDIJŲ PROGRAMA

PAULIUS STASIULIONIS

AUTOMATIZUOTAS DUOMENŲ BAZĖS SCHEMOS SUKŪRIMAS FORMŲ ANALIZĖS PAGRINDU

Magistro darbas

Darbo vadovas:
doc. dr. R. Butkienė
2013-05-24

Recenzentas:
doc. dr. T. Blažauskas
2013-05-24

Atliko:
IFM-1/4 gr. studentas
Paulius Stasiulionis
2013-05-24

KAUNAS, 2013

AUTORIŲ GARANTINIS RAŠTAS
DĖL PATEIKIAMO KŪRINIO

20.. - ..... - ..... d.
Kaunas

Autoriai,_____
(vardas, pavardė)

_____,

patvirtina, kad Kauno technologijos universitetui pateiktas baigiamasis bakalauro (magistro) darbas (toliau vadinama – Kūrinys)_____
(kūrinio pavadinimas)

_____

pagal Lietuvos Respublikos autorių ir gretutinių teisių įstatymą yra originalus ir užtikrina, kad

1) jį sukūrė ir parašė Kūrinyje įvardyti autoriai;
2) Kūrinys nėra ir nebus įteiktas kitoms institucijoms (universitetams) (tiek lietuvių, tiek užsienio kalba);
3) Kūrinyje nėra teiginių, neatitinkančių tikrovės, ar medžiagos, kuri galėtų pažeisti kito fizinio ar juridinio asmens intelektinės nuosavybės teises, leidėjų bei finansuotojų reikalavimus ir sąlygas;
4) visi Kūrinyje naudojami šaltiniai yra cituojami (su nuoroda į pirminį šaltinį ir autorių);
5) neprieštarauja dėl Kūrinio platinimo visomis oficialiomis sklaidos priemonėmis.
6) atlygins Kauno technologijos universitetui ir tretiesiems asmenims žalą ir nuostolius, atsiradusius dėl pažeidimų, susijusių su aukščiau išvardintų Autorių garantijų nesilaikymu;
7) Autoriai už šiame rašte pateiktos informacijos teisingumą atsako Lietuvos Respublikos įstatymų nustatyta tvarka.

Autoriai

_____ (vardas, pavardė)	_____ (parašas)
_____ (vardas, pavardė)	_____ (parašas)
_____ (vardas, pavardė)	_____ (parašas)
_____ (vardas, pavardė)	_____ (parašas)

SANTRAUKA

Šių laikų informacinių sistemų užsakovai reikalauja pigaus ir greitai sukurto produkto – informacinės sistemos. Bet norint kokybiškai sukurti informacinę sistemą, ne visada gaunasi greitai ir pigiai. Šio darbo tikslas – automatizuoti informacinės sistemos kūrimo procesą, taip sumažinant laiko kaštus kuriant produktus. Daugelis analitikų – programuotojų kuriant informacinę sistemą naudojasi ta pačia kūrimo metodika: surenkami reikalavimai, projektuojama sistema, sistema realizuojama. Surenkant reikalavimus yra analizuojami įvairūs dalykinės srities aspektai, tarp jų ir dokumentinės formos. Šios formos, galima sakyti, yra pradinis būsimos ekraninės formos maketas.

Keletas šiame darbe nagrinėtų duomenų bazių valdymo sistemų turi vedlius, kurie generuoja ekranines formas iš turimos duomenų bazės schemos. Taigi, galima daryti prielaidą, kad yra algoritmas, kuris veiktų priešinga kryptimi. Šiame darbe, eksperimento būdu, bus ištirta galimybė, turint ekraninės formos maketą, iš jo sugeneruoti duomenų bazės schemą.

SUMMARY

Automated Development of Database Schema Based on Forms Analysis

Nowadays customers of information systems require cheap and quicly built product – information system. But in order to create information system, it will not be cheap and fast. The aim of this work is to automate creation process of the information system, thus reducing the time cust of product developining. Many analysts use the same development method to create the information system: collecting requirements, designing system, realising system. During requirements collection analyst analize various aspects of the subject area, including the forms of the documentary. These forms are in original form of the forthcoming screen froms.

Some of database magament systems, which have been analized in this work, have wizards that generate screen forms from existing database schema. Therefore, it can be assumed that there is algorithm which works in the opposite direction. In this paper will be analyzed, in experimental way, the possibility to generate database schema form from the screen layout.

Turinys

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

TERMINŲ IR SANTRUMPŲ ŽODYNAS

DB – Duomenų bazė.
IS – Informacinė sistema.
DBVS – Duomenų bazių valdymo sistema.
PA – Panaudojimo atvėjis.
Forma – ekraninė duomenų įvedimo forma. Iš anksto parengtas dokumento arba kitokių duomenų ruošinys su tuščiais laukais, kuriuos reikia užpildyti. Toliau darbe naudojama kaip froma.

1 ĮVADAS

Šiuo metu rinkoje yra didelis komercinių ir nemokamų (taip pat ir atviro kodo) duomenų bazių valdymo sistemų (DBVS) pasirinkimas, pavyzdžiui: *Oracle, Microsoft SQL Server, IBM DB2, Microsoft Access, Microsoft Visual Foxpro MySQL*. Visos jos turi savų privalumų ir trūkumų. Kai kurios duomenų bazių valdymo sistemos neturi įrankių formoms kurti. Tokiais atvejais pasirenkama trečių šalių programinė įranga: *Microsoft Visual Studio, HTML/PHP* programavimo kalbos. Tačiau, nepriklausomai nuo to, kokią duomenų bazių valdymo sistemą pasirenka analitikas – projektuotojas, duomenų bazės projektavimo eiga yra panaši: išgaunami informacinės sistemos (IS) reikalavimai, suprojektuojama duomenų bazė (sukuriamos lentelės, nustatomi jų tarpusavio ryšiai), rašomos užklausos, kuriamos (arba generuojamos) duomenų įvedimo/redagavimo formos bei kuriamos duomenų išrinkimo ataskaitos. Bet analitikai, tirdami dalykinės srities reikalavimus, analizuoja ir dokumentų formas. Taigi, alternatyvus metodas - analizuoti ekraninius formų pavyzdžius ir struktūrinius organizacijos dokumentus, gautus iš dalykinės srities eksperto[1], ir taip gauti būtinus reikalavimus kuriamai informacinei sistemai.

Tačiau šiuolaikiniame informacinių technologijų amžiuje yra siekiama greitai ir kokybiškai pateikti produktą vartotojui. Taigi, kad duomenų bazės projektavimo procesas taptų lengvesnis, analitikui reikia pakeisti IS projektavimo procesą taip, kad realizacijos metu iš pradžių būtų kuriamos formos, o tik po to iš sukurtų formų būtų generuojama duomenų bazė. Toks IS kūrimo procesas leistų iš dalies automatizuoti duomenų bazės projektavimą.

Šiame darbe pateikiama būsimo sprendimo automatizuoti duomenų bazės kūrimą iš formų analizė. Darbo pradžioje iškeliama problema, ar galima automatizuoti duomenų bazės projektavimo procesą, kuriant jas iš ekraninių formų. Iš to išplaukia pagrindiniai darbo tikslai ir uždaviniai:

- ištirti duomenų bazės kūrimo procesą;
- suprojektuoti automatizavimo algoritmo programą.

Tai pateikiama 0-0 skyriuje.

Pagrindinis darbo objektas yra duomenų bazių projektavimo procesas. Analizuojant tiriamąjį objektą pastebėta, kad norint įgyvendinti siekiamą sprendimą reikia ištirti ir formose naudojamus elementus. Tai pateikiama 7 skyriuje.

9 skyriuje analizuojami esami sprendimai: *Oracle, Microsoft SQL Server, Microsoft Access, Microsoft Visual Foxpro, MySQL, OpenOfice.org Base,* tačiau nebuvo rasti siekiami rezultatai, t.y. duomenų bazės generavimas iš ekraninės formos.

Taigi, priimtas sprendimas kurti duomenų bazių projektavimą automatizuojantį įrankį.

2 DUOMENŲ BAZIŲ PROCESO IR ĮRANKIŲ ANALIZĖ

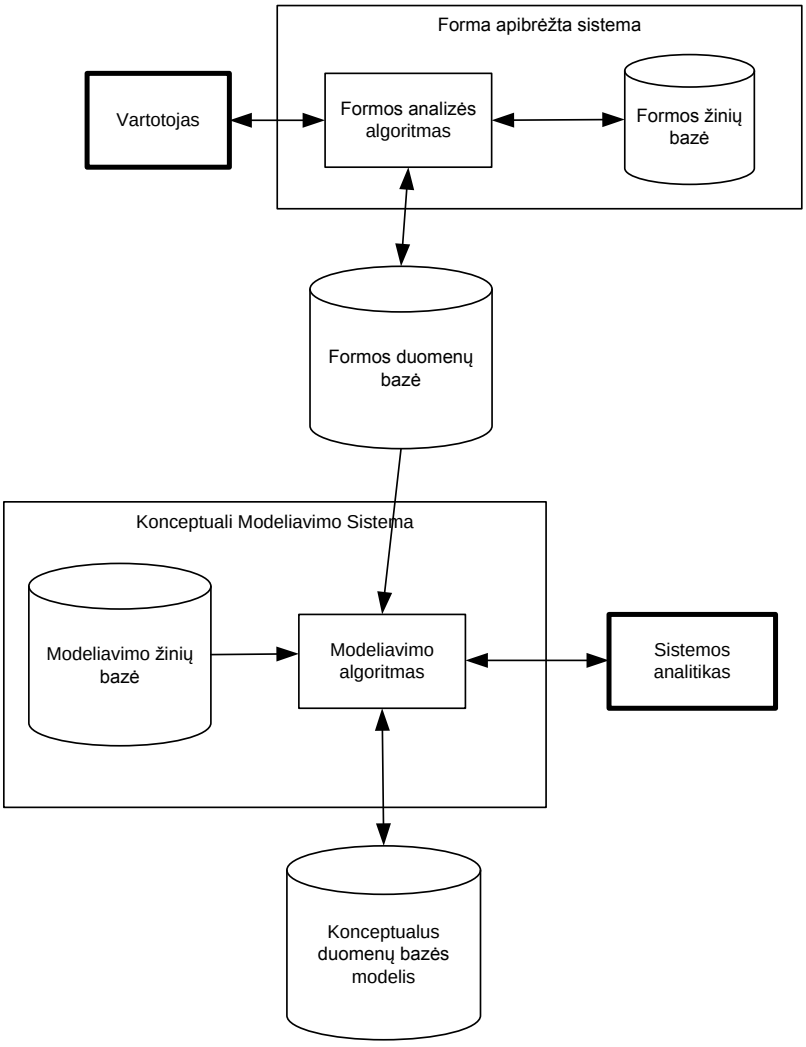
2.1 Analizės tikslas ir uždaviniai

Siekiant geriau suprasti problemas, kylančias kuriant duomenų bazes, ir įvertinti siūlomus šių problemų sprendimus, buvo suformuluoti šie analizės uždaviniai:

- Išnagrinėti duomenų bazės kūrimo procesą;
- Ištirti rinkoje esančias duomenų bazių valdymo sistemas;
- Ištirti esamus sprendimus;
- Išanalizuoti duomenų įvedimo formų struktūrą.

2.2 Esami sprendimai duomenų bazei kūrti dokumentų formų analizės pagrindu

Esybių ryšių modelio, kurio pagrindu kuriama DB, generavimo iš užpildytų duomenimis dokumentų formų algoritmo prototipas jau buvo aprašomas literatūroje. Benkt Wangler ir Bjorn Talldal yra rašę apie prototipinį algoritmą [1], kurio pagalba galima gauti esybių – ryšių modelį iš turimų dokumentinių ir ekraninių formų. Algoritmas turėtų būti įgyvendinamas tiek formomis paremtose posistemėse, kaip formos analizavimo algoritmas, tiek konceptualių modelių posistemėse, kaip esybių ryšių modeliavimo algoritmas. Taigi, turint formos eskizą, aprašomo algoritmo pagalba, turėtume gauti esybių ryšių modelį. Esybių ryšių modelį nesunkiai galima paversti į duomenų bazės schemą.



2.1 paveikslas. Forma ir modeliais paremtos sistemos architektūra [1]

Benkt Wangler ir Bjorn Talldal aprašė, kad pasitelkus formų analizės algoritmą, gaunama žinių bazė apie formą. Pritaikius modeliavimo taisykles, kaip ir koks formos elementas turi atsivaizduoti, galima gauti konceptualų duomenų bazės modelį (2.1 pav.). Svarbi šio metodo sąlyga forma turi būti užpildyta duomenimis t.y. žinių baze apie formos elementus turi būti pilna.

Kiekvienas formos elementas gali būti identifikuojamas kaip vienos reikšmės (tekstiniai laukai) arba daugiareikšmis (pasirinkimo laukai, pažymimieji laukai), šių tipų pagalba yra nustatomi ryšiai tarp formų, vėliau, atitinkamai pagal tai identifikuojamas ir ryšys tarp duomenų bazės lentelių. Vienos reikšmės elementai - tai tokie, kurie yra unikalūs ir nesikartoja per kelias lenteles, nesudarantys jokių ryšių tarp lentelių. Daugiareikšmiai elementai – tai tokie elementai, kuriuos išsaugoti neužtenka vienos lentelės. Saugant formos elementų duomenis, įrašai atsiranda dvejuose ar daugiau lentelių. Tarp jų galima identifikuoti ryšį.

Toks algoritmas yra paremtas formos elementų tipais ir duomenų, suvestų į formą, analize. Nežinant kas bus suvesta į formą, algoritmas veiktų netiksliai.

Pagal turimas žinias apie formą, t.y. formos elementų tipai, laukelių ilgis ir kiti metaduomenys, galima būtų generuoti duomenų bazę. Bet kartais formose naudojami elementai turi aprašymą, nenusakantį kokie duomenys bus vedami. Pavyzdžiui: telefono laukas formoje gali būti saugomas tiek skaičių (integer) tipo duomenimis, tiek teksto tipo duomenimis (string). Kas tiksliau bus saugoma esybėje ar duomenų bazės stulpelyje, galima būtų pasakyti tik iš turimų duomenų pavyzdžio arba vartotojas turėtų nusakyti koks tipas, ilgis ar koks kitas apribojimas turėtų būti naudojamas duomenų bazėje.

Rita Butkienė savo daktaro disertacijoje[2] papildė Benkt Wangler modelį papildomais elementais ir algoritmą suskirstė i du atskirus sluoksnius (posistemius). Vienas sluoksnis yra skirtas vartotojui. Kitas skirtas kompiuterizuojamos informacinės sistemos projektuotojui. Vartotojo sluoksnyje modelis yra tarsi dokumento kopija. Tačiau kiekvienas jos elementas yra tiesiogiai susijęs su tam skirtais kompiuterizuojamos informacinės sistemos projektuotojo sluoksnyje esančiais elementais. Vartotojo sluoksnis leidžia lengviau susišnekėti vartotojui ir projektuotojui, nes vartotojas mato tai, ka jis tikisi gauti iš kuriamos kompiuterizuotos informacines sistemos. IS projektuotojui belieka tik teisingai perteikti savo sudaryto modelio semantiką vartotojui. Kiekvieno sluoksnio elementas gali būti identifikuojamas kokiam sluoksnyje jis yra.

Savo darbe R. Butkienė bandė įrodyti, kad nebūtina formą užpildyti duomenimis, norint gauti konceptų modelį ir kad yra tam tikri dėsningumai formose, kurie yra identifikuojami kaip esybių – ryšių modelio elementai.

2.3 Tyrimo objektas, sritis ir problema

Kuriant tipines informacinės sistemas kūrimo procesas susideda iš tokių etapų: reikalavimų surinkimo, dalykinės srities analizės ir reikalavimų specifikavimo, sistemos projektavimo, kuris apima duomenų bazės ir ją naudojančių duomenų įvedimo formų (toliau, formų) ir ataskaitų modulių projektavimo, sistemos realizavimo, testavimo, bei diegimo.

Vieni šių etapų yra labiau, kiti mažiau automatizuoti. Duomenų bazei suprojektuoti ir realizuoti yra sukurta nemažai įrankių. Projektiniai sprendimai priimami įvertinus įvairią analizės ir reikalavimų specifikavimo etape surinktą informaciją: sistemos užsakovų ir naudotojų žodiniai dokumentuoti ir ne paaiškinimai, dokumentai aprašantys dalykinę sritį, dokumentų formų pavyzdžiai, esamos ar prototipinės sistemos. Dalis šios informacijos yra nestruktūrizuota ir prie jos analitikams ir projektuotojams dar reikia gerokai padirbėti, kad suvoktų užsakovų poreikius ir norus. Kita dalis – struktūrizuota, t.y. dokumentų formos, analogiškų sistemų ekraninių formų, formuojamų ataskaitų pavyzdžiai, kuriamos sistemos duomenų įvedimo formų eskizai, prototipai. Projektuoti sistemą, tame tarpe ir duomenų bazę, remiantis struktūrizuota informacija kur kas paprasčiau. Praktika ir kitų tyrėjų darbai rodo, kad analizuojant dokumentų formų pavyzdžius galima rasti dėsningumų, lemiančių duomenų bazės struktūrą. Tačiau patikimų ir patogių įrankių, leidžiančių automatiškai ar automatizuotai sukurti duomenų bazę išanalizavus dokumento struktūrą, nėra.

Informacinės sistemos kuriamos jau ne vieną dešimtmetį. Naudojimasis jomis daugumai žmonių tapo įprastu dalyku. Šiuolaikinis sistemų naudotojas yra pakankamai įgudęs reikšti savo reikalavimus struktūrizuotai, pavyzdžiui, pateikiant duomenų įvedimo formų eskizus. Iš kitos pusės sistemų kūrėjai, siekdami geresnio būsimų sistemos naudotojų poreikių patenkinimo, yra linkę pateikti naudotojams duomenų įvedimo formų prototipus. Tačiau įvedimo formų prototipams sukurti reikia skirti laiko, o esami įrankiai nepakankamai automatizuoja perėjimą nuo duomenų įvedimo formų eskizų ar prototipų iki duomenų bazės projekto. Įgudę projektuotojai kuria duomenų bazę remdamiesi savo ekspertinėmis žiniomis išnagrinėję dokumentų formų ar duomenų įvedimo formų pavyzdžius. Tačiau šios žinios nėra pakankamai formalizuotos, todėl nėra galimybės sukurti tinkamų, šias žinias taikančių įrankių.

Šio darbo **tyrimo objektu** yra duomenų bazių ir formų kūrimo procesas.

Šiame darbe **nagrinėjama problema** – nepakankamai automatizuotas duomenų bazės kūrimas duomenų įvedimo formų pagrindu. Autoriaus nuomone, formalizavus duomomenų bazių kūrėjų sukauptas žinias apie duomenų bazių projektavimą remiantis duomenų įvedimo formų prototipais ir įdiegus jas specialiose programose būtų galima efektyviau (greičiau) kurti duomenų bazes ir tuo pačiu informacines sistemas.

Taigi šio darbo **sritis -** duomenų bazių projektavimo automatizavimas.

2.4 Analizės metodai

Analizei pasirinkti šie metodai:

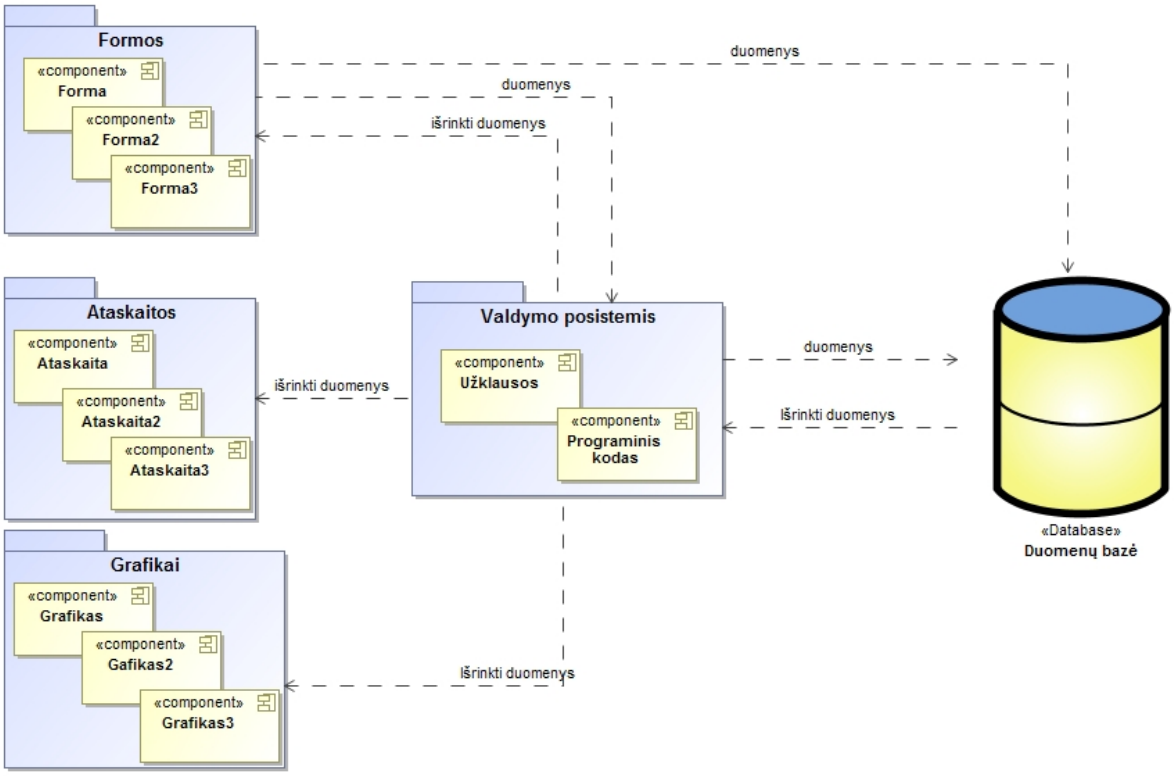
- Informacinės sistemos kūrimo procesui išanalizuoti atlikta šio proceso konceptuali analizė, apimanti duomenų įvedimo formos struktūros analizę, vartotojų analizė**.** Vartotojams išanalizuoti, renkuosi juos peržiūrėti keliais aspektais:
  - Kokie yra vartotojų tipai;
  - Kokia vartotojų kvalifikacija;
  - Kokia yra rinka.
- Tyrimo sričiai išanalizuoti atlikta DBVS lyginamoji analizė**.** Analizės metu pagal įvairius kriterijus lygintos įvairios DBVS ir jų turimi duomenų bazių kūrimo įrankiai (vedliai), bei jų funkcijos.

2.5 Tyrimo objekto analizė

Tipinis informacinės sistemos kūrimo proceso susideda iš tokių etapų:

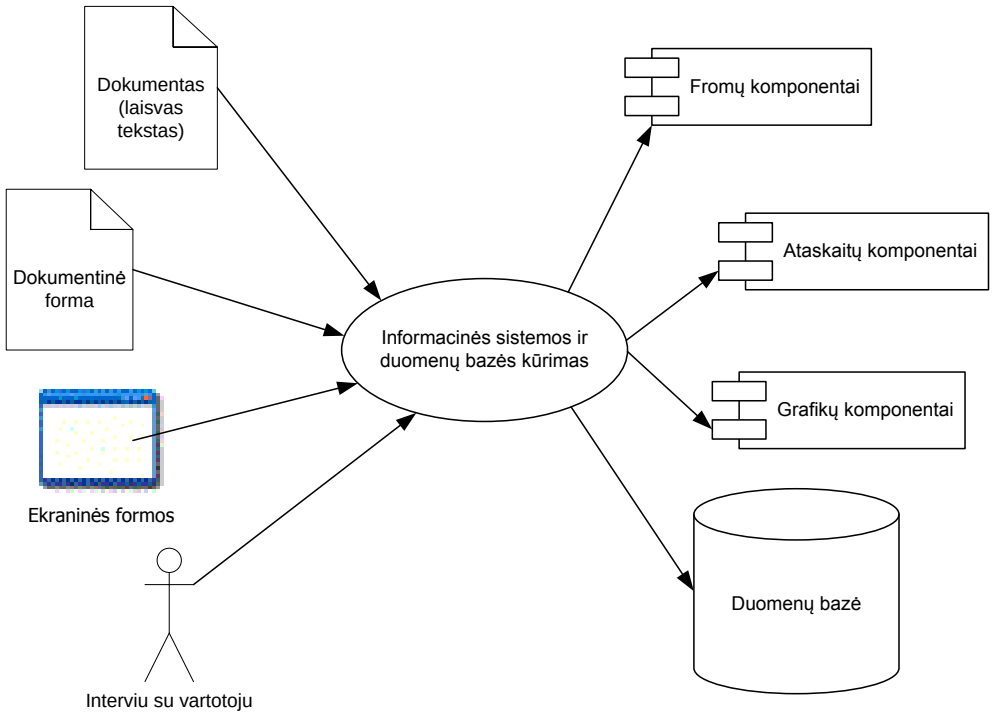
1) reikalavimų sistemai surinkimas, analizė ir specifikavimas;
2) sistemos projektavimas;
3) sistemos konstravimas (programavimas);
4) sistemos testavimas;
5) sistemos diegimas.

Prieš analizuojant šį procesą detaliau, reikėtų apžvelgti tipinės informacinės sistemos architektūrą, t.y. ją sudarančius komponentus. Tipinai komponentai yra šie: duomenų įvedimo formos, ataskaitos, grafikai ir duomenų bazė. Tipinė IS architektūra su ryšiais tarp komponentų pateikta 2.3 paveiksle. Duomenų bazė yra vienas iš esminių IS komponentų. Ją sudaro lentelės, stulpeliai, priminiai raktai, išoriniai raktai, kurie užtikrina ryšius tarp lentelių, indeksai, trigeriai (jei juos palaiko DBVS). Formos ir ataskaitos programuojamos pasirinktine programavimo kalba *C++*, *Java, PHP, Perl, Visual FoxPro, Visual basic* ir kt. Valdymo posistemyje aprašytas skaičiavimų programinis kodas ir užklausos. Užklausų aprašymui dažniausiai renkamasi *SQL* struktūrinė užklausų kalba. Duomenų bazė – informacinės sistemos sudedamoji dalis. Šie 2.2 paveiksle pavaizduoti komponentai gali išsidėstyti skirtingose sistemose arba integruoti į vieną, kaip *Microsoft Access* DBVS atveju**.** Ne išimtis ir nagrinėjamos DBVS.



2.2 paveikslas. Tipinė IS architektūra.

Duomenų bazė projektuojama surinktų reikalavimų ir jų analizės pagrindu. Reikalavimams suformuluoti informacija iš sistemos užsakovo ir naudotojo pusės gali būti pateikta įvairiomis formomis: žodine sakytine pokalbio metu, dokumentų, laisva forma aprašant dalykinę sritį, dokumentų formų, ataskaitų pavyzdžiais bei eskizais, sistemos analogų demonstravimu, sistemos ar tik jos dalies, pavyzdžiui, tik duomenų įvedimo formų prototipais (2.3 pav.). Išanalizavus visą gautą informaciją suformuluojami ir specifikuojami reikalavimai, priimami projektiniai sprendimai.

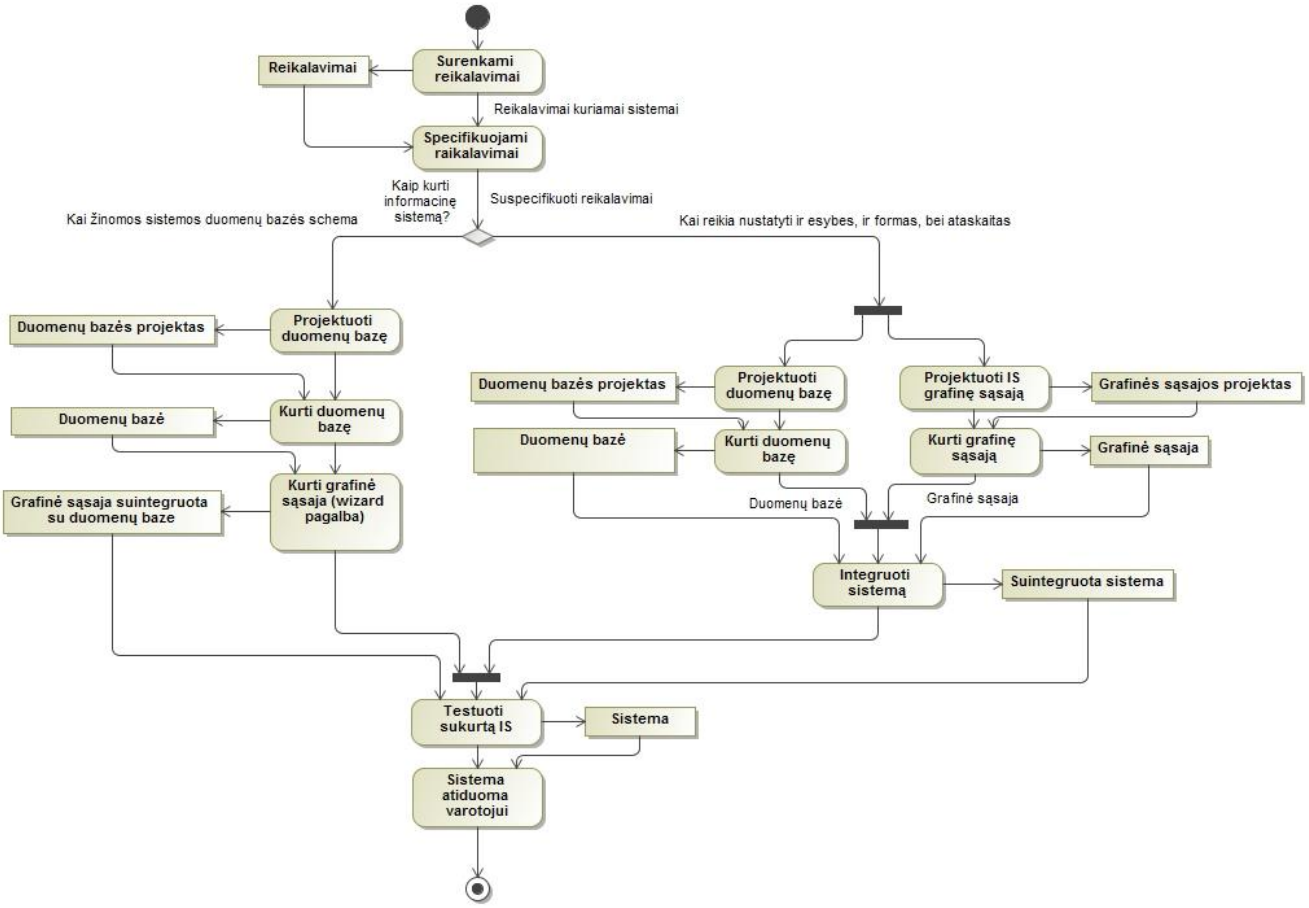


2.3 paveikslas. IS kūrimas.

Kiekvienas stulpelis turi savo tipą, kuris priklauso nuo įvedamų duomenų.

Informacinės sistemos komponentų kūrimas iš dalies priklauso nuo pasirinktų kūrimo technologijų. Kaip 2.4 paveiksle matome, yra keli variantai jiems kurti:

- Jei pasirenkama technologija, turinti fomų ir ataskaitų kūrimo vedlius, tai pirmiau sukuriama duomenų bazė , o formas ir ataskaitas galima kurti vedlio pagalba.
- Jei pasirenkama technologija be vedlio, tai duomenų bazė gali būti kuriama lygiagrečiai su formomis, ataskaitomis ir kitais komponentais. Po to, šie komponentai suintegruojami: duomenų bazės lentelės ir jų atributai yra sukabinami su formose esančiais elementais.



2.4 paveikslas. Duomenų bazių kūrimo procesas.

2.6 Vartotojų analizė

2.6.1 Vartotojų aibė, tipai ir savybės

Šio darbo vartotojų aibė yra programuotojai – analitikai arba vartotojai turintys minimalias žinias apie informacines sistemas. Žinoma, kai kurie DBVS turi vedlius, kurie iš sukurtos duomenų bazės generuoja formas. Taigi, net vartotojui, nedaug išmanančiam technologijas, yra gana lengva gauti duomenų bazės schemą.

Vartotojui keliami reikalavimai:

- turėti minimalias žinias IT technologijų srityje;
- DBVS pagrindų išmanymas.

2.6.2 Vartotojų tikslai ir problemos

Kaip ir buvo minėta šiame darbe, daugelis būsimo sprendimo vartotojų (analitikų) naudojasi tuo pačiu duomenų bazių kūrimo metodu. Vartotojų pagrindinis tikslas yra greitai sukurti produktą, kurį būtų galima iškart atiduoti kliento naudojimui.

Problemos su kuriomis šiuo metu susiduria vartotojai pateikiamos lentelėje (2.1 lentelė).

2.1 *lentelė. Problemų lentelė.*

Problema	**Kaip viskas vyksta dabar**
Automatizuotas formos kūrimas iš pateiktos duomenų bazės kai kuriose DBVS (MS Access, Open Office Database) yra įgyvendintas, bet kitus etapus tenka projektuoti neautomatizuotu būdu, pavyzdžiui DB kūrimu.	Informacinė sistema projektuojama ir programuojama rankomis, ne visada pasirenkama automatiniai įrankiai kurie atliktų vienokius ar kitokius darbus.
Duomenų bazės kūrimo etapas yra mažai automatizuotas.	Šis etapas turi kūrimo įrankių, bet neturi plačiai naudojamų automatizuotų sprendimų, kurie keliais mygtuko paspaudimais sugeneruoja duomenų bazės schemą.
Negalima pakeisti sistemos komponentų kūrimo eigos, jei norima išnaudoti esamų įrankių funkcionalumą, automatizuojantį komponentų kūrimą.	Nors ir yra žinoma busimų formų struktūra ir būtų galima projektavimą pradėti nuo jų, esami įrankiai verčia pirmiausiai projektuoti duomenų bazę, o po to kurti duomenų įvedimo formas.

2.7 Duomenų bazių valdymo sistemų analizė

Duomenų bazių valdymo sistemos pasirinkimas gali apspręsti informacinės sistemos komponentų kūrimo eigą. Kadangi dalis DBVS turi vedlius, kurie automatizuoja formų ir kitų IS komponentų sukūrimą, šiame skyriuje bus išanalizuotos šių DBVS savybės. Analizės metu bus siekiama nustatyti:

- Kokiu lygiu DBVS automatizuoja DB ir kitų komponentų kūrimą;
- Ar galima praplėsti DBVS funkcionalumą;
- Kokia DB projektavimo eiga?

Analizei pasirinkti šios DBVS:

- Oracle designer.
- Microsoft Office Access.
- Microsoft FoxPro.
- OpenBase SQL.
- MySQL.
- OpenOffice.org Base.

Kai kurios DBVS, tokios kaip *Microsoft Access*, *Oracle designer*, *OpenBase SQL, Microsoft FoxPro* turi formų kūrimo vedlius (*wizard*). Tai vienas iš būdų greitai sukurti informacinę sistemą paprastai ir nesudėtingai.

Toliau darbe aprašomos analizuotos duomenų bazių valdymo sistemos.

2.7.1 Oracle deginer

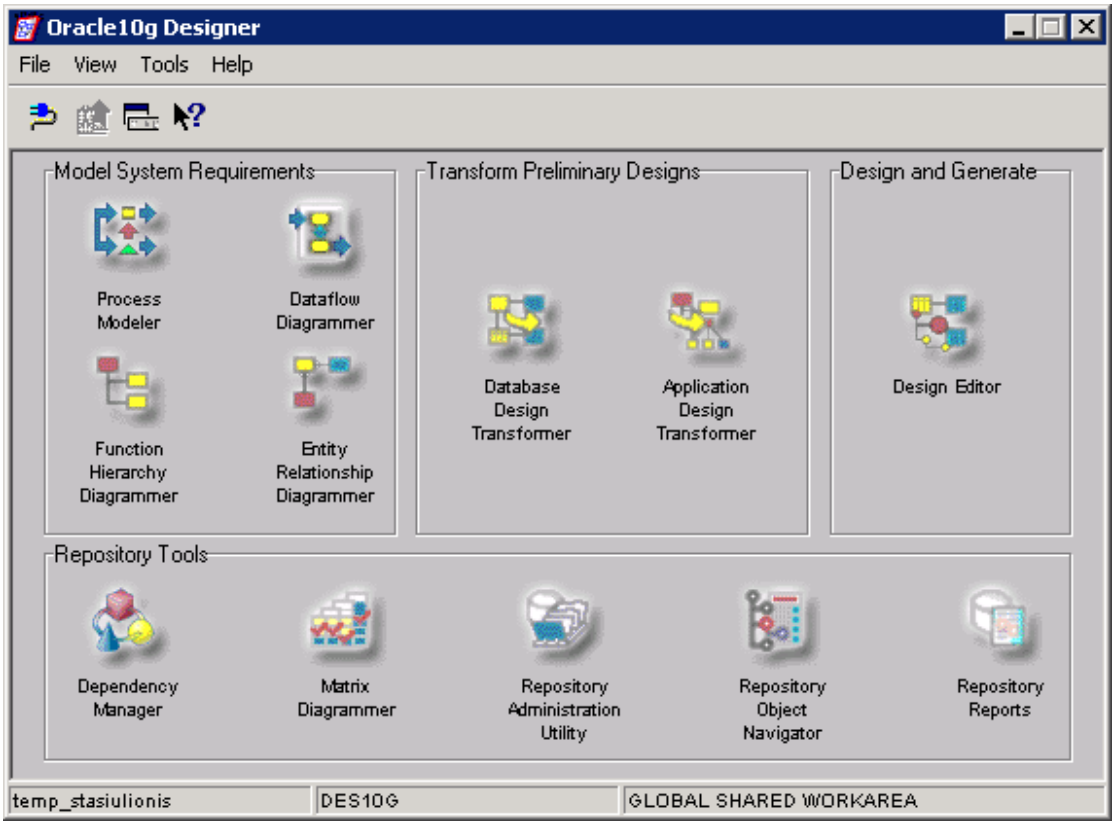
Tai *Oracle* kompanijos CASE įrankis , skirtas kurti informacines sistemas. *Oracle Designer* gali būti naudojamas kiekviename informacinės sistemos kūrimo gyvavimo ciklo etape – nuo dalykinės srities procesų modeliavimo iki pilnai funkcionuojančios informacinės sistemos realizavimo [5].

Oracle DBVS – Orcale kompanijos reliacinė duomenų bazių valdymo sistema. Pasaulyje naudojama labai plačiai duomenims saugoti, apdoroti ir analizuoti bankinėse, finansinėse, mokslinėse ir kitose labai didelėse informacinėse sistemose.

Designer įrankiai yra sugrupuoti į sritis (2.5 pav.):

- *Modeling System Requirements*
  Įrankiai naudojami šioje srityje aprašo:
  - ✓ biznio procesų modelius;
  - ✓ esybių ryšių diagramas;
  - ✓ funkcijų ir srautų diagramas;

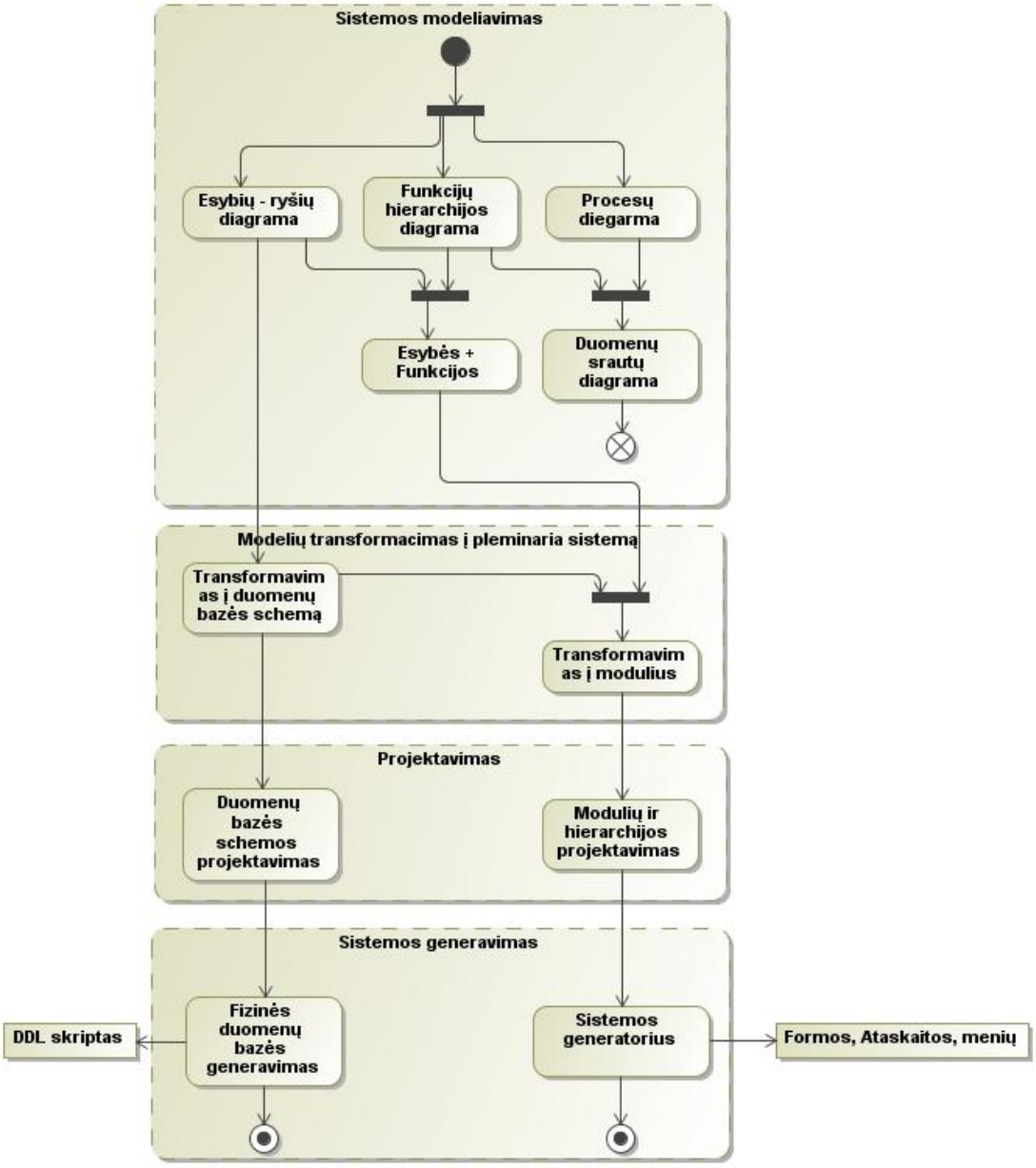
  Tai įrankių rinkinys skirtas organizacijos dalykinės srities, tikslams, procesams ir funkcijoms modeliuoti. Aprašomi sistemos detalūs reikalavimai, bei organizacijos duomenų srautai.
- *Transforming Preliminary Designs*
  Naudojama transformuoti į preminalią sistemą iš ankščiau sukurtų modelių.
- *Designing and Generating*
  Design Editor esantis šioje srityje skirtas toliau projektuoti sistemą, kuri atitiktų keliamus organizacijos reikalavimus. Plėtoti sistemą inžinerinių ir dizaino aspektais. Kurti serverio pusės komponentus ir kliento pusės programas.
- *Respository tools*
  Įrankiai naudojami šioje srityje yra skirti: kurti ir redaguoti informaciją apie sistemą, paržiūrėti ryšius tarp komponentų, generuoti ataskaitas iš turimos apie sistemą informacijos, administruoti sistemos informaciją [15].



2.5 paveikslas. Oracle designer pagrindinis langas

Sistemos kūrimo su *Oracle Designer* pagrindiniai etapai:

- Modeliavimas ir dalykinės srities analizė;
- Dalykinės srities konceptualių modelių kūrimas;
- Sistemos projektavimas ir realizacija.



2.6 paveikslas. Duomenų bazės kūrimo procesas, projektuojant Oracle Designer įrankiu

Paveiksle 2.6 pavaizduotas kūrimo procesas su *Oracle designer* įrankiu. Sistema pradžiai sumodeliuojama procesijų, funkcijų hierarchijos, duomenų srautų diagramomis. Turimi modeliai transformuojami į duomenų bazės schemą ir būsimos taikomosios programos modulius. Galutiniame etape yra baigiama projektuoti sistemą ir ji generuojama į kodą arba duomenų bazės DDL komandų rinkinį.

Šis *Oracle* produktas neturi formų ar kitų komponentų kūrimo vedlio, bet prototipinę formą automatiškai sugeneruoja iš veiklos modelio. Be to, šis produktas yra uždaras ir nėra galimybės jo praplėsti naujų funkcionalumu. Todėl jis nėra tinkamas problemos sprendimui.

2.7.2 Microsoft SQL Server

Microsoft SQL server reliacinė duomenų bazių valdymo sistema. *Microsoft SQL server* yra kliento – serverio duomenų bazė. Kliento – serverio architektūros duomenų bazių valdymo sistemos su dideliais duomenų kiekiais dirba daug geriau negu tokios DBVS kaip *Microsoft Access* [[14]].

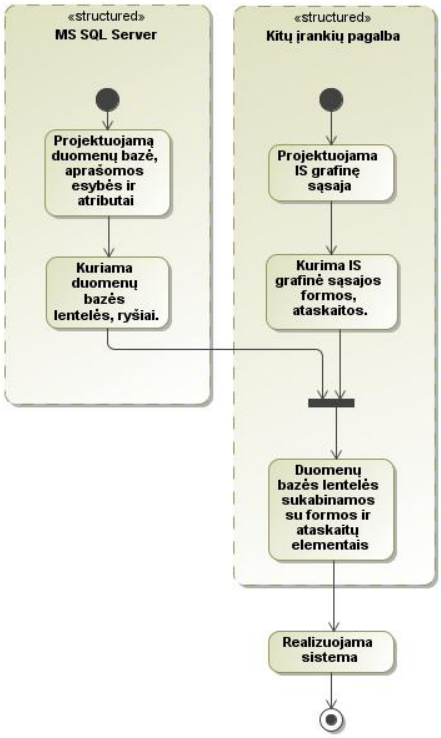
Microsoft SQL server palaiko struktūrizuotų užklausų kalbą *SQL* (angl. *Structured Query Language*). Kaip ir daugelis kitų duomenų bazių valdymo sistemų, šį sistema palaiko *SQL* standartus ir prideda savo išplėtimus.

„*Microsoft*“ kompanijai būdinga struktūrizuota užklausų kalba yra vadinama *Transact-SQL (T-SQL)*. Kuri leidžia pridėti, modifikuoti ar pašalinti duomenis.

Microsoft SQL server duomenų bazės variklyje palaiko *.NET* bendrinę kalbą (angl. *.NET Common Language Runtime (CLR)*). Tai leidžia duomenų bazėje sukurti objektus, funkcijas, procedūras, trigerius, apibrėžti vartotojų tipus ir vartotojų funkcijas, programavimo kalbas palaikančias *CLR*. *Visual Studio* paketas palaiko *CLR* integravimo galimybę su šia duomenų bazių valdymo sistema.

Duomenų bazės formos kuriamos su kitais įrankiais tokiais kaip *Visual Studio*. Duomenų bazių valdymo sistema skirta tik duomenų bazės architektūrai ir duomenims laikyti.

Microsoft SQL server palaiko XML (angl. extensible markup language). Tai Microsoft SQL Server suteikia galimybę pateikti užklausas ir duomenis XML kalba. Tai leidžia organizacijoms sumažinti investicijas ir sutaupyti optimizavimo ir duomenų saugojimo srityse [13].



2.7 paveikslas. Duomenų bazės kūrimo procesas, kuriant Microsoft SQL server DBVS

Duomenų bazės kūrimo procesas (2.7 pav.) šia DBVS yra paprastesnis lyginant su tokiais įrankiais kaip Microsoft Access, OpenOffice.org ir pan., nes kuriama tik duomenų bazė ir visa jos logika. Formos, ataskaitos ir užklausos duomenims gauti realizuojamos kitais įrankiais ir programavimo kalbomis.

Ši duomenų bazių valdymo sistema yra netinkama, nes visai neturi vartotojo sąsajos formoms kurti. Informacinėse sistemose dažniausiai naudojama tik kaip duomenų bazė.

2.7.3 Microsoft Office Access.

Microsoft korporacijos sukurta reliacinė DBVS, kuri turi intuityvią grafinę vartotojo sąsają (2.8 pav.). Microsoft Office Access yra sudedamoji Microsoft Office paketo dalis.

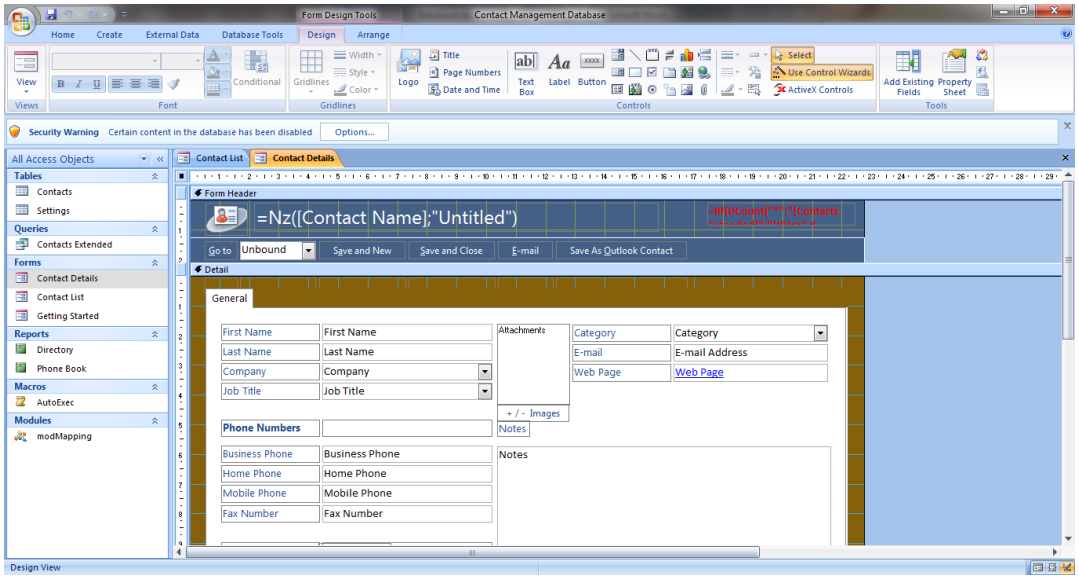
Duomenų bazė formas, ataskaitas, makro komandų rinkinius laiko viename faile.

Pagrindinai komponentai:

- Lentelės (*Tables*) – duomenų bazių lentelės.
- Formos (*Forms*) – duomenų atvaizdavimui, įvedimui, redagavimui, šalinimui skirtas komponentas.
- Užklausos (*Queries*) – duomenų išrinkimui iš DB užklausų rašymui.

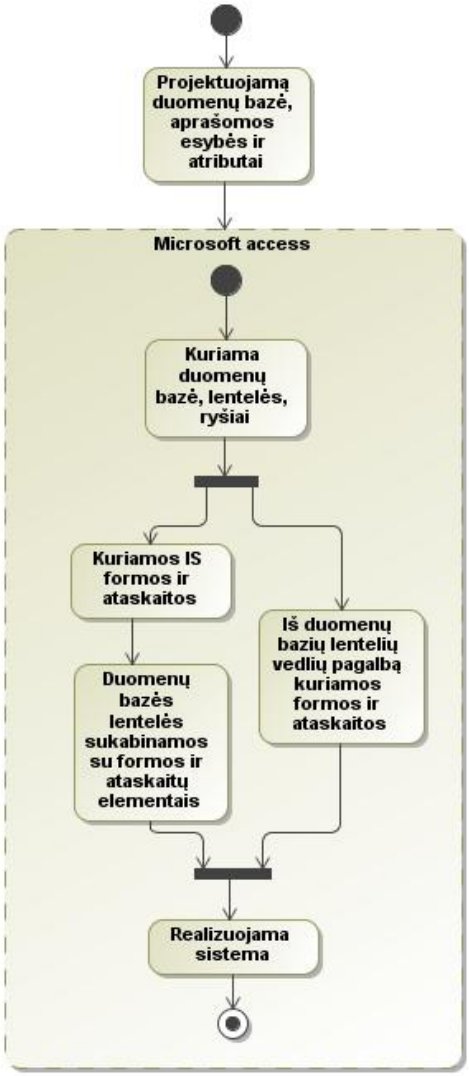
– Ataskaitos (*Reports*) – duomenų atvaizdavimui.
– Makro komandos (*Macros*) – makro komandų rašymui.
– Moduliai (*Modules*) – funkcijų ir procedūrų rašymui.

Formos kuriamos yra su *Form design* komponentu. Norint sukurti bet kokią formą su vedlio (*Wizard*) pagalba reikalinga turėti jau paruoštas duomenų bazės lenteles, esybes.



2.8 paveikslas. Microsoft Access 2007 langas

Naudojant šį įrankį, duomenų bazės kūrimo procesas (2.9 pav.) yra toks, kaip ir ankščiau minėtas: pirmiausiai, kuriamos duomenų bazės lentelės ir ryšiai. Toliau, vartotojas gali rinktis ar kurti su vedlio (*angl. wizards*) pagalba, ar kurti formas ir jose esančius elementus, programinio kodo (*SQL* užklausų, makro komandų) pagalba ir sukabinti su duomenų bazės lentelėse esančiais stulpeliais. Taip gaunama duomenų bazė, kurią jau galima naudoti.



2.9 paveikslas. Duomenų bazės kūrimo procesas naudojant Microsoft Access DBVS

Įrankis nėra tinkamas problemos sprendimui, nes tai nėra atviro kodo programinė įranga ir negalima saugomoje duomenų bazėje atskirti kur yra formos dalis, o kur yra duomenų bazės dalis.

2.7.4 Microsoft FoxPro

Microsoft kompanijos sukurta objektiškai orientuota reliacinė DBVS (2.10 pav.), kuri leidžia sukurti stabilius įmonių duomenų bazių sprendimus. Šis DBVS gali laikyti duomenis ir kituose serveriuose - Microsoft SQL Server[12],

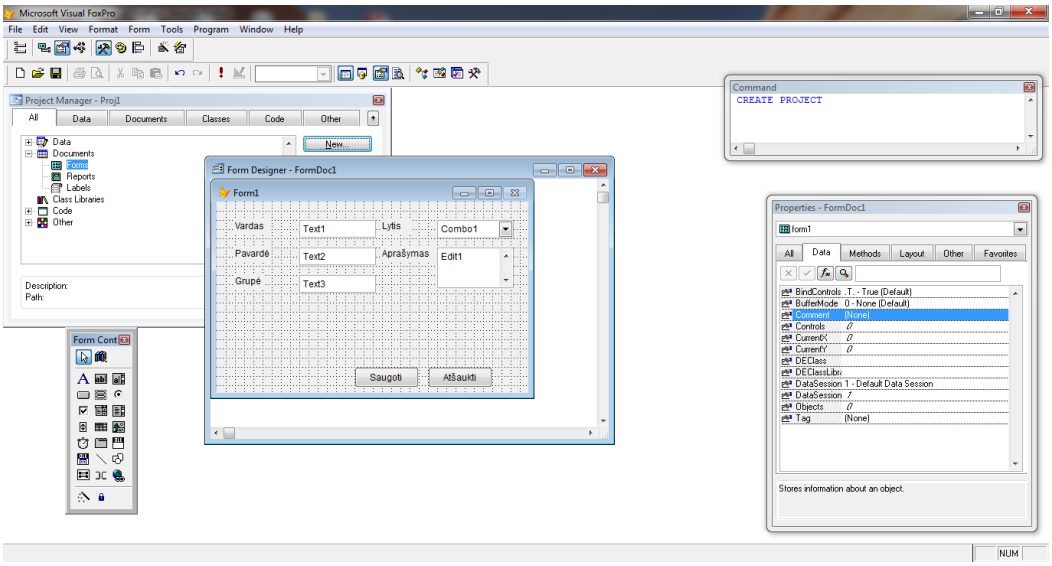
Šis DBVS yra skirstomas į tokius komponentus:

- Duomenų bazė.
  - Duomenų bazės (*databases*) – šiame komponente laikoma lokali duomenų bazė;
  - Duomenų išrinkimo užklausų (*queries*) - duomenų išrinkimui iš duomenų bazės skirtas komponentas.
- Dokumentai.
  - Formos (*forms*) – duomenų įvedimui, redagavimui, šalinimui ir vaizdavimui skirtas komponentas;
  - Ataskaitos (*reports*) – duomenų atvaizdavimui ataskaitoje skirtas elementas;
  - Etiketės (*labels*) - duomenų atvaizdavimui ataskaitoje skirtas elementas;
- Klasės.
- Kiti.

- Meniu (*menus*) – formų aktyvavimui skirtas komponentas.
- Tekstiniai failai (*text files*) – duomenų ir pastoviųjų kintamųjų saugojimui skirti tekstiniai failai.
- Kiti failai (*other files*) .

Formos, ataskaitos, duomenų bazė saugoma atskiruose failuose. Todėl bendram projekte galima atskirti kur yra formos, kur duomenų bazė ar kiti elementai. Formos failą galima nuskaityti ir iš failo struktūros įmanoma suprasti, kokie elementai užprogramuoti vienoje ar kitoje formoje.

Formas galima kurti tiek su vedlio (*wizard*) pagalba, tiek patiems, su *form designer* dėliojant reikalingus elementus ir susiejant juos su jau sukurtomis duomenų bazėmis.



2.10 paveikslas. Microsoft Visual FoxPro SP2 langas.

Duomenų bazės kūrimo procesas yra panašus kaip ir *Microsoft Access*, *Open Office.org Base* (2.9 pav.). Sukuriamos duomenų bazės lentelės ir iš jų generuojamos arba kuriamos formos, bei ataskaitos.

DBVS yra iš dalies tinkama kuriamam sprendimui. Nors *Microsoft Visual FoxPro* nėra laisvai platinama, bet formų failų struktūrą galima nuskaityti ir suprasti, kokie elementai užprogramuoti toje formoje. Viena sąlyga, formos failas turi būti sukurtas kaip atskira klasė.

2.7.5 MySQL

Tai greita, lanksti, daugiavartotojiška programinė įranga, paremta struktūrine užklausų kalba. Šis DBVS serveris yra pamėgtas daugelio pasaulio vartotojų. *MySQL* yra atviro kodo programinė įranga ir yra laisvai platinama[10].

Daugelis vartotojų renkasi šią DBVS internetinėms svetainėms kurti, nes ji yra suderinta su daugelių platformų.

Priėjimui prie *MySQL* duomenų gali būti panaudota daug kalbų ir bibliotekų, pavyzdžiui: *PHP, C, C++*, C#, Java, Perl, Python. Kiekvienai šių kalbų sukurtos specialios bibliotekos (API). Taip pat *MySQL* duomenų bazėms yra sukurta *ODBC* sąsaja *MyODBC*, leidžianti duomenis pasiekti betkuria kalba, neturinčia specialios bibliotekos, tačiau palaikančia *ODBC* komunikavimo mechanizmą.

Kadangi duomenų bazės valdymo sistema savyje neturi įrankių kurti formoms, kūrimo procesas yra toks pat kaip ir *Microsoft SQL server* (2.7 pav.).

MySQL DBVS yra tinkamas kuriamam sprendimui, kadangi jis yra prieinamas su daugeliu programavimo kalbų. O DBVS paremta *SQL* (struktūrine užklausų kalba).

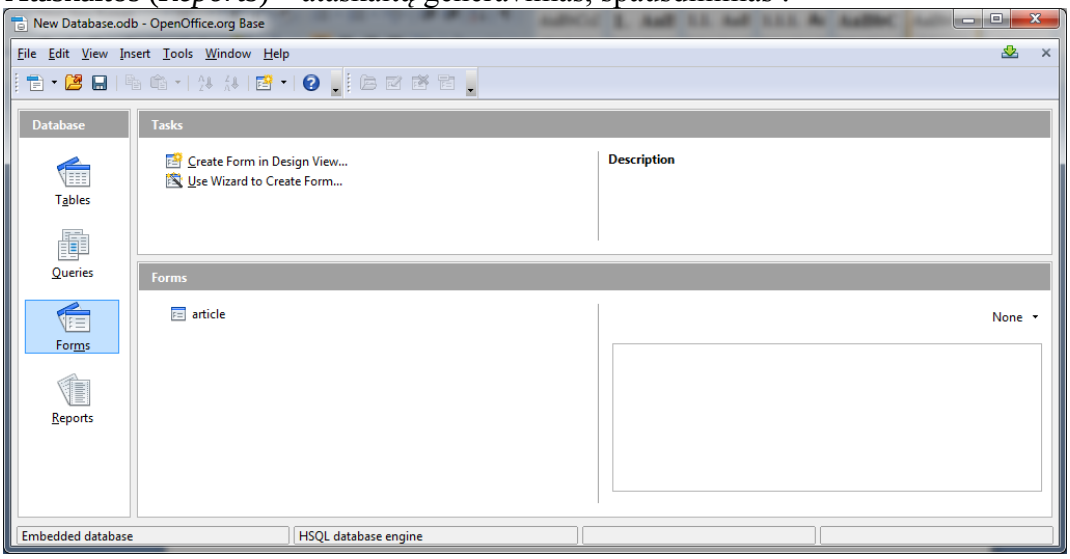
2.7.6 OpenOffice.org Base

Tai - atviro kodo paketo *OpenOffice.org* duomenų bazių modulis (2.11 pav.), pasirodęs pakete nuo 2.0 versijos. *OpenOffice.org Base* DBVS naudoja *HSQLDB* DBVS variklį, kuris yra parašytas *Java* kalba [9].

Taip pat, be integruotos *HSQLDB* DBVS yra galimybė jungtis prie išorinių DBVS, tokių kaip *MySQL*, *PostgreSQL*, *Oracle* ir kt., naudojant vidines, *ODBC* ar *JDBC* tvarkykles.

Pagrindiniai OpenOffice.org Base (2.11 pav.) komponentai:

- Lentelės (*Tables*) - DB lentelių kūrimas, duomenų įvedimas;
- Užklausos (*Queries*) – duomenų išrinkimo užklausų rašymas;
- Formos (*Forms*) – formų kūrimas, duomenų įvedimas;
- Ataskaitos (*Reports*) – ataskaitų generavimas, spausdinimas .



2.11 paveikslas. Open Office.org 3.2 Base langas

Kadangi *OpenOffice.org* yra atviro kodo įrankių rinkinys, analoginis komerciniam *Microsoft Office* produktui, tai duomenų bazės kūrimo procesas (žiūrėti 2.9 pav.) yra kaip ir jau nagrinėtame *Microsoft Access* DBVS.

DBVS yra tinkama kuriamam sprendimui, nes tai atviro kodo programinė įranga. Dėl to yra lengva nuskaityti failus, kuriuose yra formos ir kiti elementai, aprašyti bendros paskirties duomenų struktūrų aprašomąja kalba XML (*Extensible Markup Language).*

2.7.7 DBVS analizės apibendrinimas

Nagrinėtos DBVS buvo surašytos į lentelę (2.2 lentelė), kurioje jos lyginamos keturiomis savybėmis: gebėjimas skaityti SQL kodą, kaip duomenys saugomi, grafinė vartotojo sąsaja formoms kurti, naudojamas metodas, kuriuo projektuojama duomenų bazė. Šios savybės parodo, kad visose duomenų bazių valdymo sistemose projektavimo eiga yra panaši. Ir kiek patogu paprastam vartotojui susiprojektuoti duomenų bazę.

2.2 *lentelė. Lyginamoji lentelė.*

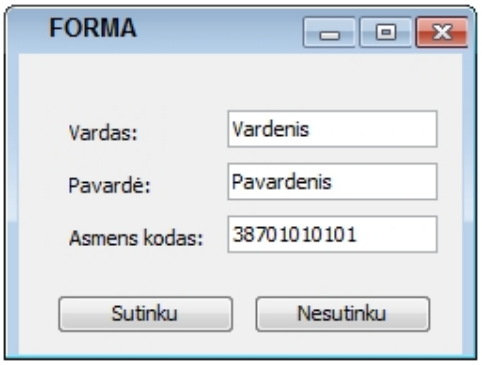
Savybė / DBVS	Gali skaityti SQL kodą	Turi formų kūrimo vedlius	Galima praplėsi funkcionalumą	Grafinė vartotojo sąsaja formoms kurti	Duomenų bazės projektavimo eiga
Oracle Desginer 10g	Taip	Ne	Ne	Taip	• Veiklos procesų diagramos; • Duomenų srautų diagramos; • ER diagrama; • Funkcijų hierarchijos diagrama; • Formos, ataskaitos ir duomenų bazės.
Microsoft SQL server	Taip	Ne	Ne	Ne	• Lentelės; • Ryšiai tarp lentelių; Toliau naudojami trečiųjų asmenų sukurti įrankiai arba rašoma programa.
Microsoft Office Access 2007	Taip	Taip	Ne	Taip	• Lentelės; • Ryšiai tarp lentelių; • Užklausos; • Formos; • Ataskaitos.
Microsoft FoxPro 9.0 SP2	Iš dalies	Taip	Iš dalies (trečiųjų šalių programine įranga)	Taip	• Lentelės; • Ryšiai tarp lentelių; • Užklausos; • Formos; • Ataskaitos.
MySQL	Taip	Ne	Ne	Ne	• Lentelės; • Ryšiai tarp lentelių; Toliau naudojami trečiųjų asmenų sukurti įrankiai arba rašoma programa.
OpenOffice.org 3.2	Taip	Taip	Iš dalies (trečiųjų šalių programine įranga)	Taip	• Lentelės; • Ryšiai tarp lentelių; • Užklausos; • Formos; • Ataskaitos.

Visų nagrinėtų sprendimų kūrimo procesas yra gana panašus. Esminis skirtumas, kad vienų DBVS formos kuriamos pačioje duomenų bazių valdymo sistemoje, o kitų formas kuria trečiųjų asmenų sukurti įrankiai ar programinė įranga.

2.8 Formos struktūros analizė

Šiame skyriuje išnagrinėsime duomenų įvedimo formos struktūrą, ją sudarančius elementus. Tai leis suprasti, kokio tipo informacija vadovaujamasi projekatuojant duomenų bazę ir kaip tai gali būti panaudota projektavimui automatizuoti.

Formos – tai tam tikras parengtas arba kitokių dokumentų ruošinys su tuščiais laukais[21]. Informacinėse sistemose formos naudojamos duomenims įvesti arba išvesti(2.12 pav.).



2.12 Formos pavyzdys

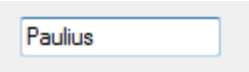
Formą sudaro laukai. Kiekvienas laukas yra skirtingo tipo, priklausomai nuo to kokie duomenys bus vedami į ta lauką.

2.8.1 Pagrindiniai formų elementai

Laukai šiame darbe bus vadinami formos elementais. Tarp programuotojų yra ilgainiui nusistovėję formose naudojami elementai: teksto langelis (angl. *text box*), žymimasis langelis (angl. *check box*), išskleidžiamas sąrašas (angl. *combo box*), sąrašo laukas (angl. *list box*), pasirinkimo laukas (angl. *radio button*), mygtukas (angl. *button*), duomenų tinklas (angl. *grid*). Šiuos elementus galima rasti visose programavimo įrankiuose, kurie turi formų tvarkymo priemones. Minėti elementai padeda valdyti, įvesti ir redaguoti duomenų bazių duomenis. Kiti formose naudojami elementai dažniausiai būna specifiniai, pasirinkto programavimo įrankio ar duomenų bazės valdymo sistemos, naudojami elementai.

2.8.2 Teksto langelis

Stačiakampio ribojama vieta kompiuterio ekrane, skirta teksto laukui(2.13 pav.).



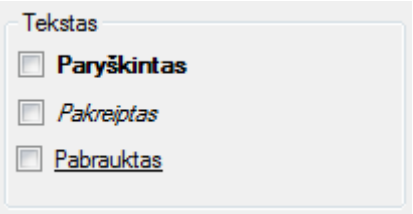
2.13 paveikslas. Teksto langelis (angl.TextBox)

Į teksto langelį gali būti rašoma: komanda, parinktis, parametro reikšmė, nuostata, vardas, slaptažodis, adresas ir kitoks programai pateikiamas tekstas. Renkamą tekstą galima taisyti iki bus paspaustas mygtukas su nurodymu programai, kad ji priimtų surinktą tekstą [21].

Teksto laukelio tikslas yra leisti vartotojui įvesti tekstinę informaciją naudojamą programoje. Vartotojo sąsajos gairės rekomenduoja vienos eilutės teksto laukelį, kai tik naudojama viena eilutė, bet galima ir kelių eilučių teksto laukas, jeigu yra daugiau nei viena eilutė. Tipiškas teksto laukelis yra bet kokio dydžio stačiakampis.

2.8.3 Žymimasi langelis

Žymimas langelis (2.14 pav.), tai formos valdikis - mažas kvadratinis langelis, turintis dvi būsenas: *pažymėtas* arba *nepažymėtas*.



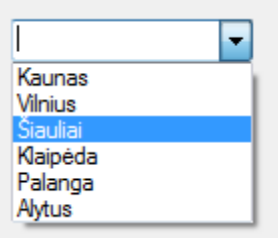
2.14 paveikslas. žymimasis langelis (angl. check box)

Pažymėto langelio viduje būna varnelė, kryžiukas arba kitoks ženklas.

Žymimasis langelis valdo kurią nors parinktį: nustato ar ji pasirinkta, ar nepasirinkta. Jei greta (vienoje grupėje) yra keli žymimieji langeliai, tai kiekvieno jų būsena nepriklauso nuo gretimų langelių ir gali būti pažymėta kiek norima langelių. Tuo žymimieji langeliai skiriasi nuo akučių [21].

2.8.4 Išskleidžiamas sąrašas

Išskleidžiamasis sąrašas (2.15 pav.), tai išplečiamojo sąrašo arba sąrašo laukas ir vienos eilutės teksto laukelio derinys, leidžiantis vartotojui reikšmę įvesti tiesiogiai į valdiklį, arba pasirinkti iš esamų pasirinkimų sąrašo.[21]



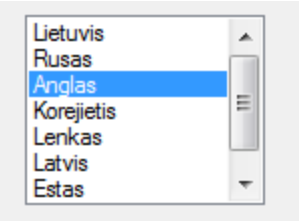
2.15 paveikslas. Išsiskleidžiantis sąrašas (angl. combo box)

Šis grafinis elementas formoje, dažniausia valdo dvi duomenų bazės lenteles, kurios sujungtos „vienas su daug“ ryšiu.

2.8.5 Sąrašo laukas

Valdiklis, kuriame galima matyti sąrašą, jį peržiūrėti ir iš jo pasirinkti norimą reikšmę (2.16 pav.).

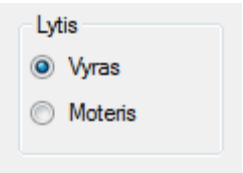
Kol sąrašas neišskleistas, langelyje matomas tik vienas pasirinktas sąrašo elementas ir išskleidimo mygtukas (trikampėlis), kurį paspaudus sąrašas išskleidžiamas. Iš išskleisto sąrašo galima pasirinkti kitą elementą. Į langelį netelpančius elementus galima pamatyti pasinaudojus slinkjuostę, kuri tokiu atveju įdedama į sąrašo langelį [21].



2.16 paveikslas. sąrašo laukas (angl. list box)

2.8.6 Pasirinkimo laukas

Pasirinkimo laukas (2.17 pav.), tai mažas skritulėlis, turintis dvi būsenas: *pažymėtas* arba *nepažymėtas*, ir esantis tokių akučių grupėje, iš kurių tik viena gali būti pažymėta.[21]



2.17 paveikslas. pasirinkimo laukas

2.8.7 Mygtukas

Ekrane pavaizduotas mygtuko paveikslėlis (2.18 pav.), kurį paspaudus pelės klavišu vykdomas tam tikras veiksmas – komanda [21].

2.18 paveikslas. Mygtukas (angl. button)

Formose, kurios turi sąryšį su duomenų bazėmis, mygtukuose užprogramuojama SQL komandos, kurios įveda, redaguoja, šalina ar atvaizduoja duomenis iš duomenų bazės.

2.8.8 Duomenų tinklelis

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

2.19 paveikslas. Duomenų tinklelis (angl. Grid)

Tinklelis arba duomenų tinklelis yra grafinės vartotojo sąsajos elementas (valdiklis) (2.19 pav.), kuris rodo lenteles duomenims peržiūrėti. Tipišką tinklelį sudaro keletas arba visos toliau iš išvardintų charakteristikų:

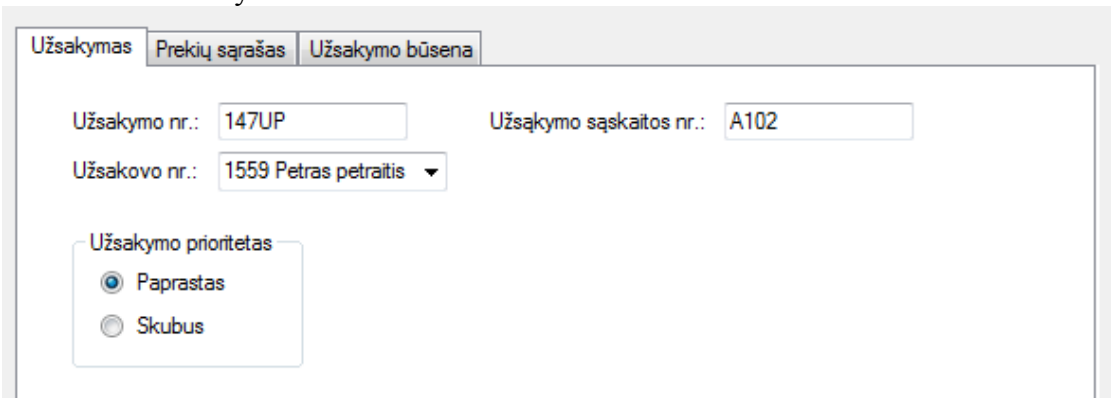
- Paspaudus stulpelio antraštę, jei norite pakeisti rūšiavimo tvarką tinklelyje;
- Paleidžiant stulpelio antraštes keisti jų dydį ir jų seką;
- Galimybė langelyje redaguoti jame esančius duomenis;
- Eilučių ir stulpelių skyriklius, ir kintamoji fono spalva.

Taigi, tinklelis plokštumos sritis, suskirstyta į langelius gulsčiomis ir stačiomis lygiagrečiomis linijomis, išdėstytomis vienodu atstumu (dažniausiai) viena nuo kitos [21].

2.8.9 Kortelė

Grafinės sąsajos elementas – ąseles turinti lango dalis (2.20 pav.) , kurioje rodomos tam tikros vienos rūšies nuostatos, informacija arba vienas dokumentas (dokumento dalis).

Kortelės turinį atspindi pavadinimas, užrašytas jos ąselėje. Vienu metu matoma tik viena lango kortelė. Kitų kortelių matomos tik ąselės su užrašais. Spustelėjus ąselę, atveriamas ją atitinkančios kortelės turinys.



2.20 paveikslas. Kortelės (angl. Tabs)

Į kai kurių programų vieną langą galima įkelti keletą dokumentų – kiekvieną savo kortelėje. Pavyzdžiui, naršyklės „Mozilla FireFox“ ir „Opera“ gali atverti keletą tinklalapių atskirose kortelėse, skaičiuoklės „Microsoft Excel“, „OpenOffice.org Calc“ kortelėse atveria skaičiuoklės lakštus.

Naudojant korteles galima atvaizduoti ir redaguoti duomenis iš skirtingų duomenų bazės lentelių.

2.8.10 Formos elementų analizės apibendrinimas

Formos elementų analizės metu buvo apžvelgti populiariausiai naudojami formos elementai. Kiekvienas elementas išsiskiria savo tipu ir saugomais duomenimis. Iš elemento tipo, galima nuspėti kokia duomenų bazės struktūra yra suprojektuota. Todėl šiame skyriuje išnagrinėsime keltą populiarių įrankių:

- Vidual Sudio;
- HTML/PHP.

2.9 Formų kūrimo įrankiai

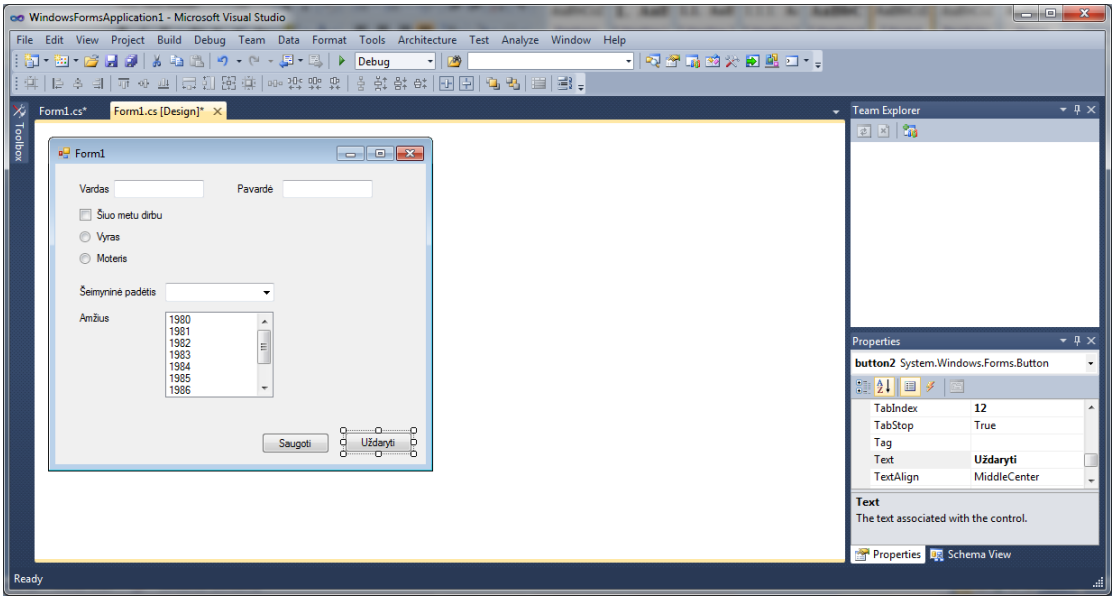
Išnagrinėjus įvairias duomenų bazių valdymo sistemas, pastebėta, kad keletas nagrinėtų duomenų bazių valdymo sistemų neturi specialių įrankių, pritaikytų formoms kurti. Tokioms duomenų bazėms valdyti formos sukuriamos kitais įrankiais, bei priemonėmis.

2.9.1 Microsoft Visual Studio

„*Microsoft Visual Studio*“(2.21 pav.) yra komponentų pagrindu paremti kūrimo įrankiai ir technologijos. Šiomis priemonėmis kuriamos galingos, aukštos kokybės sistemos. Be to, „*Visual Studio*“ optimizuoja komandos, kuri projektuoja, kuria ir diegia įmonėse sprendimus [17].

„Visual Studio“ buvo "*Microsoft*" pagrindinis vystomas produktas nuo *Visual Basic* dienų. „*Visual Studio 2010*“ su giliomis tradicijomis, pataisyta vartotojo aplinka ir keletu naujų funkcijų, kad ir toliau produktas būtų laikomas kaip pagrindinė kodo kūrimo platforma [16].

Visual Studio įrankis gali būti naudojamas kurti konsolines ir grafines vartotojo sąsajas turinčias programas: *Windows Forms* programų, interneto svetainių, interneto programų ir interneto servisus, tiek rašyti kodą visose platformose: *Microsoft Windows*, *Windows Mobile*, *Windows CE .NET Framework*, *.NET Framework* ir *Microsoft Silverlight* [17].



2.21 paveikslas. „Microsoft Visual Studio 2010“ vartotojo sąsaja

Formų ar konsoliniai programavimo produktai, kuriami šiuo įrankiu, yra sujungiami su duomenų bazių valdymo sistema, *Visual Studio* yra suderinta su *Microsoft SQL server*, bet galima

kuriamą produktą, parsisiuntus papildomas bibliotekas, susieti ir su kitomis populiariomis DBVS: *MySQL, PostgreSQL* ir pan.

2.9.2 HTML/PHP

PHP (angl. *Hypertext Preprocessor*) yra plačiai naudojama atviro kodo bendrosios paskirties skriptų kalba, kuri yra ypač tinkama interneto svetainių kūrimui ir gali būti integruota su *HTML*.

PHP programavimo kalbos struktūra yra panaši į *C, Java, Perl* kalbų struktūrą. Teikdama įrankius, reikalingus manipuliuoti informacija, *PHP* padeda kurti dinaminį *HTML* turinį. *PHP* lengvai pritaikomas ir efektyvus įrankis, todėl tampa vienu svarbiausių gerinant paslaugų kokybę.[[19]]

HTML (angl. *Hyper text Markup Language*) – tai hiperteksto žymėjimo kalba, naudojama pateikti turinį internete. Kalbą standartizuoja *W3* konsorciumas. *HTML* yra pagrindinis blokas tinklalapiams atvaizduoti internete [20].

Taigi, interneto tinklalapių programavimo kalba *PHP* yra populiari šiais laikais. Formos kuriamos *HTML* hiperteksto žymėjimo kalba, *PHP* programavimo kalbos sąveikos pagalba yra sukabinamos su tokiomis populiariomis duomenų bazėmis: *MySQL, Microsoft SQL server, PostgreSQL* ir kitomis DBVS.

2.9.3 Formų kūrimo įrankių analizės apibendrinimas

Kadangi apžvelgti formų kūrimo įrankiais sukurtos formos gali valdyti įvairias duomenų bazių valdymo sistemas, tai papildomų įrankių, vedlių, šie įrankiai neturi. Dėl to, lygiagrečiai kuriamoms formoms reikia projektuoti ir duomenų bazę. Suprojektavus duomenų bazę, formos elementai programinio kodo pagalba yra susiejami su duomenų bazės lentelėmis, bei jų stulpeliais.

2.10 Tyrimo tikslas ir uždaviniai

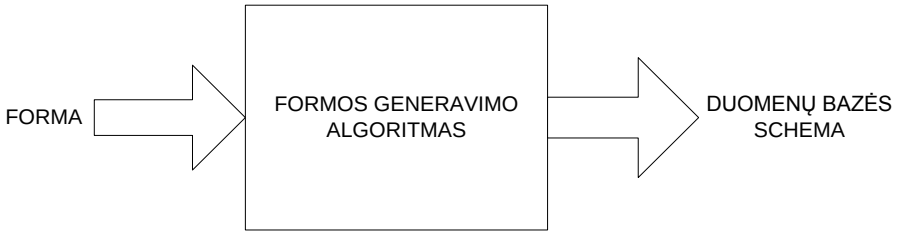
Išanalizavus duomenų bazių valdymo sistemas, formų kūrimo įrankius ir kitus sprendimus literatūros šaltiniuose, buvo nustatyta, kad nagrinėti įrankiai pakankamai neautomatizuoja duomenų bazių kūrimo iš formų. Dėl to buvo nuspręsta siekti tokio tikslo: sukurti prototipinę sistemą, kuri sudarytų galimybę automatiškai kurti duomenų bazę iš duomenų įvedimo formų ir tokiu būdu leistų sutrumpinti DB kūrimo laiką.

Norint įgyvendinti ši tikslą buvo iškelti šie uždaviniai:

- Specifikuoti algoritmą, duomenų bazėms iš formų kurti.
- Suprojektuoti prototipinę programą.
- Suprogramuoti prototipinę programą duomenų bazių generavimui iš formų.
- Eksperimentiškai ištirti sukurtą programą.

2.11 Siekiamas sprendimas

Siekiamas sprendimas – universalus algoritmas (2.22 pav.), kuris pateikus turimą duomenų įvedimo formą, iš jos būtų sugeneruota duomenų bazės schema. Algoritmas tūrėtų būti ištestuotas eksperimentine sistema. Šiuo sprendimu būtų pagreitintas duomenų bazės kūrimo procesas. Pagreitinti procesą galima: išanalizavus dalykinės srities dokumentų šablonus, iškart kurti jų analogą grafinėje sąsajoje – formose. Iš formoje sukurtų elementų (teksto langelis, išsiskleidžiantis sąrašas, pasirinkimo langelis, lentelių ir kitų elementų.),generuoti reliacines duomenų bazes, populiariose DBVS.



2.22 paveikslas. Siekiamo sprendimo diagrama

Norint pasiekti siekiamą sprendimą, reikia ištirti formose esančius elementus ir jų atvaizdavimo taisykles įvairiuose DBVS.

2.12 Analizės išvados

1. Išanalizavus įvairias DBVS nustatyta, kad visi šie įrankiai automatizuoja duomenų bazės ir pačios IS kūrimo procesą, kuris yra paremtas vienu IS projektavimo metodu, kurį naudoja daugelis analitikų: pagal specifikuotus reikalavimus sukuriama duomenų bazė, o po to programos moduliai (duomenų įvedimo formos, ataskaitos ir pan.).
2. Dažniausiai analitikai kurdami kompiuterizuotas IS taiko dalykinės srities dokumentų, popierinių ir ekraninių formų analizę, taigi, galima daryti išvadą, kad ekraninė forma ar jos maketas sukuriami pirmiau, negu suprojektuojama duomenų bazė. Iš to išplaukia poreikis turėti sprendimą, kuris automatizuotų duomenų bazės kūrimą ekraninių formų analizės pagrindu.
3. Išanalizavus formų kūrimo įrankius, nustatyta, kad formose naudojami elementai yra pasikartojantys, ir galima daryti prielaidą, kad įmanoma sukurti formų analizės algoritmą, kurį būtų galima taikyti įvairiosiose DBVS.
4. Keturios iš analizuotų duomenų bazės valdymo sistemų turi automatizuotus formų generavimo vedlius (angl. wizards), bet nei vienoje nėra galimas atvirkštinis generavimas: iš formos kurti duomenų bazę, todėl yra poreikis sukurti programą.

3 SISTEMOS REIKALAVIMAI

3.1 Reikalavimų specifikacija

Sprendimo taikymo sritis. Sprendimas bus taikomas kuriant duomenų bazes. Kaip minėta analizės dalyje (2 skyrius), šiuo metu pirmiau kuriama duomenų bazė ir vedlių pagalba kuriamas formų rinkinys informacinei sistemai. Naujoje programinėje įrangoje bus bandomas įgyvendinti atvirkštinis vedlio veikimas, turint grafinę vartotojo sąsają, pagal elementų taisykles, gauti duomenų bazę. Šis sprendimas bus pritaikomas programinės įrangos ir informacinių sistemų kūrimo procese, taip pat turėtų palengvinti programuotojo ir projektuotojo darbą, nes daugelis programuotojų ir projektuotojų, kuriant IS grafinę vartotojo sąsają remiasi dokumentų analizės pateiktomis popierinėmis formomis.

3.1.1 Funkciniai reikalavimai

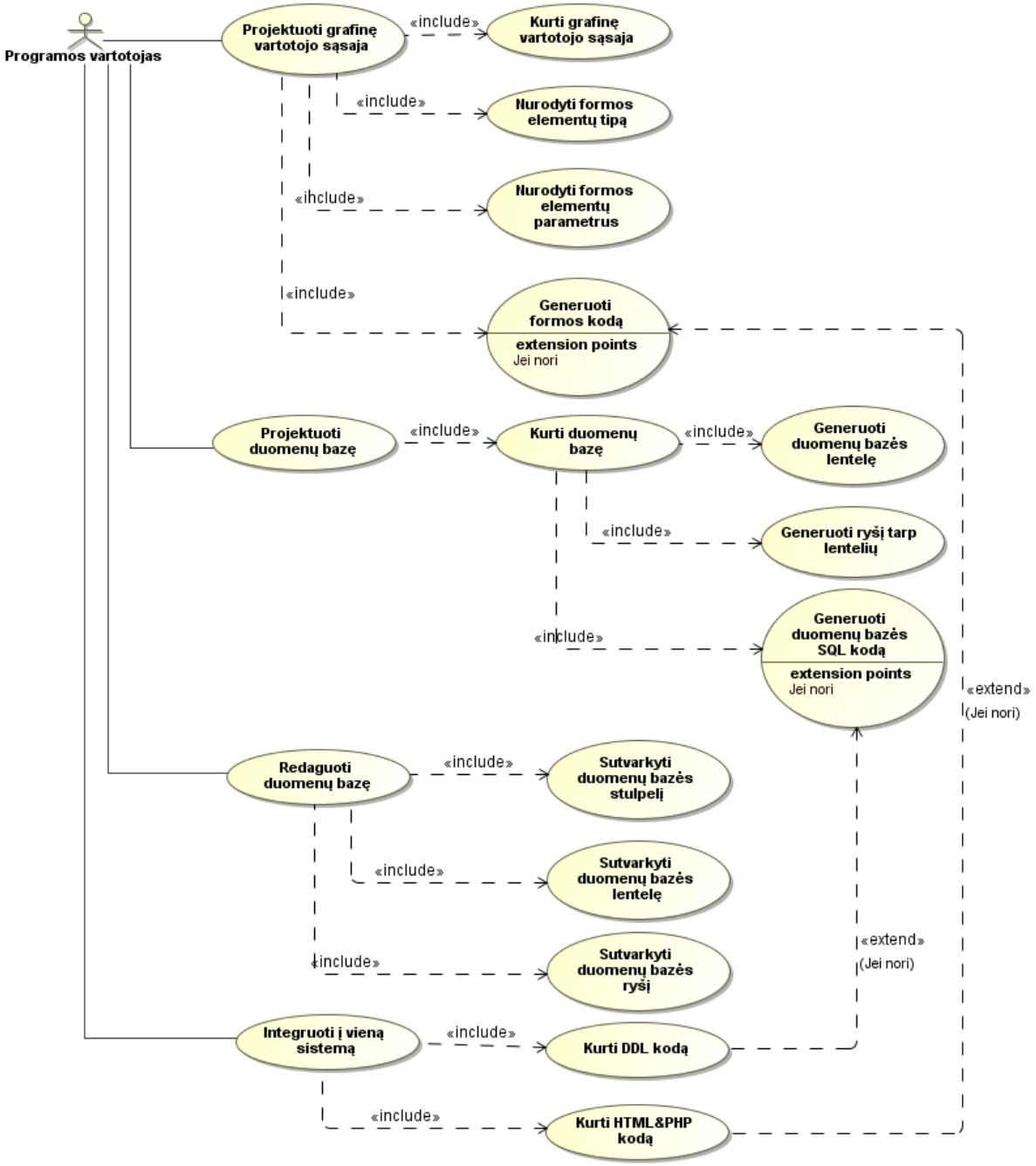
Kuriamas sprendimas turi veikti su *HTML* hiperteksto žymėjimo kalbos pateikiamomis formomis. Generuojant duomenų bazę, programa turėtų nuskaityti pateiktą formą ir pateikti duomenų bazės *SQL* scenarijų, bei kontaktuojančios su duomenų baze, *PHP* klasės kodą.

Taip pat, kuriamo sprendimo programa turi turėti galimybę grafiškai sudėlioti formos elementus savo aplinkoje ir iš sukurtos formos generuoti duomenų bazę. Pakoregavus formą duomenų bazės scenarijus turi būti generuojamas kitas.

3.1.1.1 Sprendimo panaudojimo atvejai

Kuriamas sprendimas turės keletą pagrindinių funkcijų: projektuoti grafinę vartotojo sąsają, kurti duomenų bazę, redaguoti sukurtą duomenų bazę ir visą tai suintegruoti į vientisą sistemą.

Šios funkcijos išreikštos panaudojimo atvejais (3.1 pav.). Sistemos (programos) vartotojas vienas, bet ja naudosis daugelis vartotojų, kuriančių IS, vaidmenų: analitikai, programuotojai, projektuotojai.



3.1 paveikslas. Bendras kompiuterizuojami panaudojimo atvejai

Pagrindiniai sistemos panaudojimo atvejai, kurie atskleidžia magistrinio darbo esmę: „kurti duomenų bazę“ PA, su jį papildančiais generuoti „duomenų bazės lentelę“ PA, „generuoti ryšį tarp lentelių“ PA.

Kiekvienas kuriamos sistemos panaudojimo atvejis specifikuojamas lentelėmis ir sekų diagramomis (9.1 priedas), kurios apibrėžia sistemos elgseną ir vartotojo galimus veiksmus su šiuo sprendimu.

3.1.2 Nefunkciniai sistemos reikalavimai

Reikalavimai sistemos išvaizdai:

- Vartotojo sąsaja turi būti intuityvi, lengvai suprantama, neperkrauta grafiniais elementais, kurie sulėtintų sistemos vykdomas funkcijas.

Reikalavimai vykdymo charakteristikoms:

- Duomenų bazės formavimas turi trukti ne ilgiau kaip 30 sekundžių;
- Greitos atidarymo, išsaugojimo operacijos dirbant su formomis ir ataskaitomis (ne ilgiau kaip 5 sekundės).

Reikalavimai veikimo sąlygoms:

- Kuriamas sprendimas turi veikti įvairioms Windows operacinės versijoms su galimybe veikti Linux operacinės sistemos aplinkoje.

Reikalavimai sistemos patogumui:

- Kuriamas sistemos algoritmas turi būti universalus ir pritaikomas kitiems tyrimams bei bandymams, ne tik su *HTML* formomis ir *MySQL* duomenų baze.

Kultūriniai-politiniai reikalavimai:

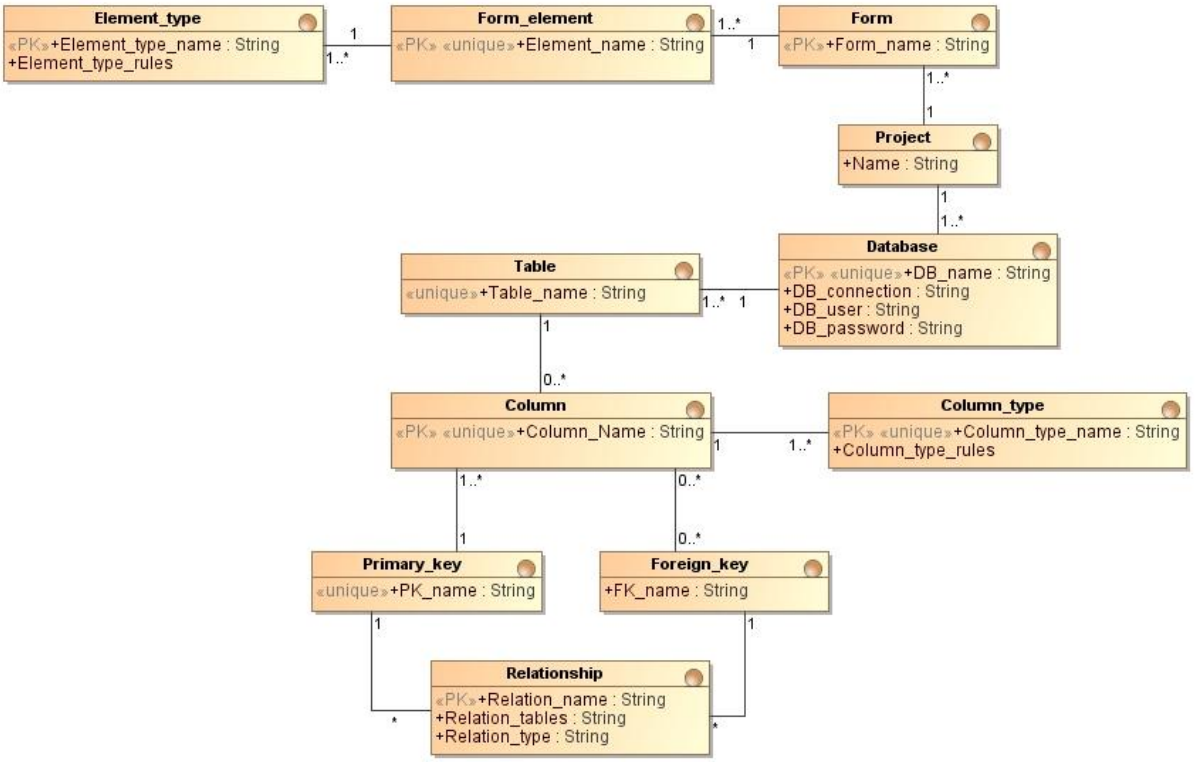
- Sistemoje naudojami kitų gamintojų komponentai turi būti atviro kodo, nemokami nekomerciniams tikslams.

Teisiniai reikalavimai:

- Apribojimų susijusių su įstatymais nėra.

3.2 Dalykinės srities modelis

Pradinis dalykinės srities klasių modelis pateikiamas (3.2 pav.). Šiame modelyje yra aprašomos pagrindinės klasės: Project, Form, Form_element, Element_type, Database, Relationship, Table, Column, Column_type. Šios klasės sujungtos ryšiais.



3.2 paveikslas. Dalykinės srities klasių modelis

3.3 Reikalavimų analizės apibendrinimas

Sistema turi būti paprasta, lengvai suprantama vartotojui. Sistema turi išpildyti pagrindinę funkciją: automatizuotai kurti duomenų bazę iš sukurtų ir sistemai pateiktų *HTML* formų. Kuriama sistema orientuota į patyrusį duomenų bazių specialistą, taigi, grafinė vartotojo sąsaja neturėtų būti perkrauta grafiniais elementais, kurie sulėtintų sistemos atliekamas funkcijas.

4 SISTEMOS PROJEKTAS

Sistemos tikslas, kaip ir pagrindinis darbo tikslas - eksperimentiškai ištirti ar galima generuoti duomenų bazę iš formos, naudojant tik formos elementų taisykles.

4.1 Duomenų bazės generavimo algoritmas

Duomenų bazės algoritmas, generuojantis duomenų bazę veikia pagal 4.1 lentelėje nusakytas taisykles.

4.1 *lentelė. Sprendimų lentelė*

		Veiksmas									
		Kurti formą atitinkančią lentelę	Kurti formos lentelės stulpelį	Kurti formos lentelės stulpelį PK	Kurti formos lentelės stulpelį FK	Kurti nauja lentelė	Kurti naujos lentelės stulpelį	Kurti naujos lentelės stulpelį PK	Kurti naujos lentelės stulpelį FK	Sukurti ryšį tarp formos lentelės ir naujos lentelės.	Sukurti ryšį tarp „Select“ atitinkančios lentelės ir „Group“
Sąlyga	Rasta nauja forma (Form)	X		X							
	Rastas naujas teksto laukas (Text field)		X								
	Rastas naujas slaptažodžio laukas (Password field)		X								
	Rastas naujas pasirinkimo laukas (Radio button)		X								
	Rastas naujas žymimasis laukas (Check box)		X								
	Rasti nauji daugiau negu 2 žymėjimo laukai (Check box)				X	X	X	X	X	X	
	Rastas naujas teksto laukas (Textarea)		X								
	Rastas naujas išsiskleidžiantis sąrašas (Select)				X	X		X		X	
	Rastas naujas išsiskleidžiančio sąrašo elementas (Option)						X				
	Rastas naujas išsiskleidžiančio sąrašo grupavimas Option group					X		X		X	X
	Rasta nauja duomenų lentelė (Grid)*				X	X		X		X	
	Rastas naujas duomenų lentelės stulpelis (Grid column)*						X	X			

Algoritmas analizuoja jam pateiktą formą, joje esančius formos elementus ir jų tam tikrus parametrus - atributus. Formos duomenys saugomi į informacinės sistemos duomenų bazę kuri turi lenteles, bei stulpelius.

Kiekvienas nagrinėjamos formos laukas turi atitinkamai pagal algoritmo taisyklę priskirtą duomenų bazės stulpelį.

4.1.1 Duomenų bazės generavimo algoritmo formalus aprašas

Šiame skyriuje bus formaliai aprašytas duomenų bazių generavimo, iš pateiktų formų, algoritmas. Algoritmas suskirstomas į keturis žingsnius. Kiekvienas jų aprašytas matematiniais modeliais.

4.1.1.1 Duomenų bazės generavimo algoritmo pirmasis žingsnis

Pirmojo žingsnio metu yra nuskaitomas failas su formomis. Gaunama išsami informacija apie formose esančius elementus, kokie jų tipai ir kokiai formai tas elementas priklauso.

Pirmojo žingsnio formalųjį aprašą sudaro šis trejetas: formų aibė, elementų aibė, ir funkcinių priklausomybių aibė.

$$M_1 = < F, E, \Sigma_F > \text{ (4.1)}$$

F – formų aibė; Formų aibė sudaryta iš formų: $F = \{ f_1, f_2, f_3, \dots, f_n \}$. (4.2)

E – formos elementų aibė; Elementų aibė sudaryta iš elementų: $E = \{ e_1, e_2, e_3, \dots, e_n \}$. (4.3)

$\Sigma_F : E \rightarrow F$ (4.4) Formos elementų aibės elementas turi funkcinę priklausomybę su atitinkama formos aibę.

Aibė, sąryšis, atvaizdis	Predikatas
Σ_F	form_content

4.1.1.2 Duomenų bazių generavimo algoritmo antrasis žingsnis

Antrojo žingsnio metu analizuojami paprastieji elementai: įvedimo laukai, žymimieji langeliai, pasirinkimo laikai. Šio tipo elementai, duomenų bazės struktūrai, nėra labai sudėtingi. Antrojo algoritmo žingsnio formalųjį aprašą sudaro 4.5 formulėje nurodytos aibės:

$$M_2 = < M_1, L, S , \Sigma_T , \Delta_{FL}, \Delta_{ES} > \text{ (4.5)}$$

Duomenų bazių aibė sudaryta iš duomenų bazių: $DB = \{ db_1, db_2, db_3, \dots, db_n \}$. (4.6)

Lentelių aibė turi funkcinę priklausomybę nuo atitinkamos duomenų bazės: $L \rightarrow DB$. (4.7)

L – lentelių aibė; Lentelių aibė yra sudaryta iš lentelių: $L = \{ l_1, l_2, l_3, \dots, l_n \}$. (4.8)

S – lentelės stulpelių aibė; Stulpelių aibė sudaryta iš stulpelių: $S = \{ s_1, s_2, s_3, \dots, s_n \}$. (4.9)

Stulpeliai turi funkcinę priklausomybę nuo atitinkamos lentelės: $S \rightarrow L$. (4.10)

Kiekviena sukurta lentelė turi pirminį raktą **pk** kuris priklauso **PK** – pirminių raktų aibei. Pirminių raktų aibę sudaro pirminiai raktai: $PK = \{ pk_1, pk_2, pk_3, \dots, pk_n \}$. (4.11) PK yra stulpelių aibės S poaibiai: $PK \subseteq S$. Taigi pirminių raktų aibė niekada nebūna tuščia aibė $PK \neq \emptyset$ (4.13)

$\Sigma_T : S \rightarrow L$ (4.14) Stulpelių aibės elementas turi funkcinę priklausomybę su atitinkamu duomenų bazės lentelės elementu.

$\Delta_{FL} : F \rightarrow L$ (4.15) Formų aibės elementas, turi funkcinę priklausomybę su duomenų bazės lentelių aibės elementu.

$\Delta_{ES} : E \rightarrow S$ (4.16) Formos elementų aibės elementas, turi funkcinę priklausomybę su duomenų bazės lentelės stulpeliu.

Aibė, sąryšis, atvaizdis	Predikatas
Σ_T	table_content
Δ_{FL}	form_table_relation
Δ_{ES}	column_felement_relation

Jei yra forma ***f*** kuri priklauso formų aibei ***F*** , ir yra elementas ***e,*** kuris priklauso formos elementu aibei ***E***, formos ir elementai bendrai aprašo formų turinį. Šiai formai yra lentelė ***l*** kuri priklauso lentelių aibei ***L*** ir stulpelį ***s***, kuri priklauso stulpelių ***S*** aibei, ir bendrai lentelės su atitinkami stulpeliais sudaro duomenų bazės lentelę. Tarp formos ir lentelių yra funkcinį ryšis.

$$\forall f \in F\ \forall e \in E\ [form\_content(e, f)] \Rightarrow \exists l \in L\ s \in S\ [table\_content(l, s)] \wedge [form\_table\_relation(f,l)] \wedge [column\_felement\_relation(e,s)] \quad (4.17)$$

4.1.1.3 Duomenų bazių generavimo algoritmo trečiasis žingsnis

Trečiais žingsnis analizuoja sudėtingesnius formų elementus išsiskleidžiantį sąrašą (angl. *select*). Tokių elementų duomenims saugoti reikia kurti papildomą duomenų bazės lentelę, kurioje saugomi duomenys apie išsiskleidžiančio sąrašo pasirinkimus.

Šį žingsnio formalųjį aprašą sudaro toks aibių rinkinys:

$$M_3 = <M_2, S_{opt}, E_{opt}, L_f, L_{opt}, E_{sel}, PK, FK, \Sigma_S, \Sigma_{ST}, \Delta_R, \Delta_{FL}, \Delta_{ES}, \Delta_{PL}, \Delta_{IL}> \quad (4.18)$$

Išsiskleidžiančio sąrašo elementų aibė ir jų pasirinkimų aibė yra formos elementų aibės poaibiai: $E_{sel} \subseteq E$ ir $E_{opt} \subseteq E$ **(4.19).**

Lentelės saugančios sąrašo pasirinkimus yra lentelių aibės poaibis: $L_{opt} \subseteq L$ **(4.20)**

Pasirinkimų duomenų bazės lentelės stulpelių aibė yra duomenų bazės lentelės stulpelių poaibis: $S_{opt} \subseteq S$ **(4.21)**

$\Sigma_S : E_{opt} \rightarrow E_{sel}$ **(4.22)** Išsiskleidžiančio sąrašo (angl. *option*) lauko pasirinkimas, turi funkcinę priklausimybę su atitinkamu išsiskleidžiančių sąrašu (angl. *select*).

$\Sigma_{ST} : S_{opt} \rightarrow L_{opt}$ **(4.23)** Išsiskleidžiančio sąrašo stulpelių aibės elementas, turi funkcinę priklausomybę su atitinkamais išsiskleidžiančio sąrašo lentelių aibės elementais.

$\Delta_R : FK \rightarrow PK$ **(4.24)** Išorinių raktų aibės elementas, turi funkcinę priklausomybę su pirminių raktų aibės elementais.

$\Delta_{FL} : F \rightarrow L$ **(4.25)** Fromų aibės elementas turi tam tikrą funkcinę priklausomybę su duomenų bazės lentelių aibės elementu.

$\Delta_{ES} : E \rightarrow S$ **(4.26)** Formos elementų aibės, turi tam tikrą funkcinę priklausomybę su duomenų bazės lentelės stulpeliu.

$\Delta_{PL} : PK \rightarrow L_f$ **(4.28)** Pirminis raktas turi funkcinę priklausomybę su duomenų bazės formos lentele.

$\Delta_{IL} : FK \rightarrow L_{opt}$ **(4.29)** Išorinis raktas turi funkcineę priklausomybę su duomenų bazės išsikleidžiančio sąrašo pasirinkimų lentele.

Aibė, sąryšis, atvaizdis	Predikatas
Σ_S	describe_select
Σ_{ST}	describe_select_table
Δ_R	describe_relationship
Δ_{FL}	form_table_relation
Δ_{ES}	column_felement_relation
Δ_{PL}	describe_main_table
Δ_{IL}	describe_foreign_table

Jei forma ***f***, turi išsikleidžiančio sąrašo elementą e_{sel} , ir šis elementas turi jam priskirtu pasirinkimo elementų e_{opt}. Šie elementai yra formos turinio dalis ir aprašo išsikleidžiančio sąrašo elementą, šiai formai turi egzistuoti tokia lentelė l_f ir tokie stulpeliai stulpeliai s_{opt}, kurie priklauso lentelei l_{opt}. Pirminis raktas ***pk***, kuris priklauso lentelė l_f, ir išorinį raktas ***fk***, kuris priklauso lentelei l_{opt} ir kuris rišas su pirminiu raktu ***pk***. Kiekvienas formos elementas turi funkcinį ryši su duomenų bazės lentelių stulpeliais.

$$\forall f \in F\ \forall e_{sel} \in E_{sel}\ \forall e_{opt} \in E_{opt}\ [form\_content(e_{sel}, f)] \wedge [describe\_select(e_{sel}, e_{opt})] \Rightarrow \exists\, l_f \in L\ l_{opt} \in s_{opt} \in S_{opt}\ [describe\_select\_table(l_{opt}, s_{opt})] \wedge [column\_felement\_relation(e_{opt}, s_{opt}) \wedge pk \in PK\ fk \in FK\ [describe\_relationship(pk, fk)] \wedge [describe\_main\_table(l_f, pk)] \wedge [describe\_foreign\_table(l_{opt}, fk)] \wedge [form\_table\_relation(f, l_{opt})] \quad (4.30)$$

4.1.1.4 Duomenų bazių generavimo algoritmo ketvirtas žingsnis

Ketvirtas algoritmo žingsnis tai trečio žingsnio tęsinys. Šis žingsnis analizuoja sugrupuotus išsiskleidžiančius sąrašus (4.1 pav.).

4.1 Paveikslas. Sugrupuotas išsiskleidžiantis sąrašas

Šio elemento duomenims saugoti reikia dviejų papildomų duomenų bazės lentelių: vienoje saugome grupes , antroje lentelėje saugomi grupės pasirinkimai. Taigi, aptikus tokį elementą duomenų bazės schemoje bus kuriamos dvi lentelės, kurios ryšiais bus sujungtos su pagrindine formos DB lentele.

Ketvirto žingsnio matematinį modelį sudaro 4.31 formulėje pateiktos aibės.

$$M_4 = <M_3, L_{optg}, S_{optg}, \Sigma_{OG}, \Sigma_{SG}, \Sigma_{GF}, \Sigma_G, \Sigma_P, \Delta''_R, \Delta''_{IL} > \textbf{(4.31)}$$

Pasirinkimų grupės elementų aibė, yra formos elementų aibės poaibis: $E_{optg} \subseteq E$ **(4.32)**

Pasirinkimus grupes saugančių duomenų bazės lentelių aibė yra lentelių aibės poaibis:

$L_{optg} \subseteq L$ **(4.31)**

Pasirinkimų grupių duomenų bazės lentelės stulpelių aibė yra duomenų bazės lentelės stulpelių aibės poaibis: $S_{optg} \subseteq S$ **(4.33)**

$\Sigma_{OG} : E_{opt} \rightarrow E_{optg}$ **(4.34)** – Išsiskleidžiančio sąrašo (angl. *select*) pasirinkimo (angl. *option*) elementas, turi funkcinę su atitinkamą pasirinkimų grupe(angl. *option group*).

$\Sigma_{SG} : \Sigma_{OG} \rightarrow E_{sel}$ **(4.35)** – Pasirinkimų grupė turi funkcinę priklausomybę su išsiskleidžiančiu sąrašu.

$\Sigma_{GF} : L_{optg} \rightarrow L_f$ **(4.36)** Pasirinkimų grupes saugantys duomenų bazės lentelių aibės elementai, turi funkcinę priklausomybę su pagrindiniais formos lentelių aibės elementais.

$\Sigma_G : L_{opt} \rightarrow L_{optg}$ **(4.37)** Pasirinkimų duomenų bazės lentelės aibės elementai, turi funkcinę priklausomybę su pasirinkimų grupių duomenų bazės lentelės aibės elementais.

$\Sigma_P : S_{opt} \rightarrow L_{opt}$ **(4.38)** Pasirinkimų stulpelių aibės elementai, turi funkcinę priklausomybę su pasirinkimų duomenų bazės lentelių aibe.

$\Delta''_R : FK_{opt} \rightarrow PK_{optg}$**(4.39)** – Išorinių raktų aibės elementas, turi funkcinę priklausomybę su pirminių raktų aibės elementais.

$\Delta''_{IL} : FK_{opt} \rightarrow L_{opt}$**(4.40)** Išorinis raktas turi funkcinę priklausomybę su duomenų bazės išsikleidžiančio sąrašo pasirinkimų lentelių aibe.

Aibė, sąryšis, atvaizdis	**Predikatas**
Σ_{OG}	*describe_selectgroup*
Σ_{SG}	*describe_selectgroup''*
Σ_{GF}	*describe_select_table*
Σ_G	*describe_groups_table*
Σ_P	*describe_option_table*
Δ_R	*describe_relationship*
Δ''_R	*describe_relationship''*
Δ''_{IL}	*describe_foreign_table''*

Jei forma ***f***, kuri turi išsiskleidžiantį sąrašą e_{sel}, ir šio sąrašo pasirinkimai e_{opt}, yra sugrupuoti į e_{optg} grupes. Šie elementai yra formos turinio dalis ir aprašo sugrupuota išsiskleidžiantį sąrašą, tai tokiai forma turi egzistuoti duomenų bazės lentelę l_f, grupių duomenų bazės lentelę l_{optg} ir pasirinkimų duomenų bazės lentelė l_{opt}, šios duomenų bazės lentelės, Pasirinkimų gupes sauganti lentelė l_{optg} turi pirmini raktą pk_{optg}, pagrindinė formos lentelė l_f turi išorini raktą fk_{optg}, kuris rišasi su pirminis pk_{optg}, ir yra ryšis, sudarytas iš priminio rakto pk_{opt}, kuris priklauso l_{opt} lentelei ir išorinis raktas fk_{opt}, kuris priklauso lentelei l_{optg}. Pirminiai raktai aprašo pagrindines ir išorinio rakto lenteles. Šių aibių elementai apibrėžia duomenų bazės struktūra, grupuotam išsikleidžiančio sąrašo formos elementui, kiekvienas formos elementas turi funkcinį ryši su duomenų bazės lentelių stulpeliais.

$$\forall f \in F\ \exists e_{sel} \in E_{sel}\ e_{optg} \in E_{optg}\ e_{opt} \in E_{opt}\ [form\_content(e_{sel}, f)]$$
$$\wedge [describe\_selectgroup''([describe\_selectgroup(e_{opt}, e_{optg})], e_{sel})] \Rightarrow \exists l_f \in L_f\ l_{optg} \in L\ l_{opt} \in L\ pk_{optg} \in$$

$$PK_{optg}\ fk_f \in FK_f\ [describe\_relationship(pk_f, fk_{optg})] \wedge \exists\, pk_{optg} \in PK_{optg}\ fk_{opt} \in FK_{opt}\ [describe\_relationship``(pk_{optg}, fk_{opt})] \wedge [describe\_main\_table(l_f,pk)] \wedge [describe\_foreign\_table(fk_{optg}, l_f)] \wedge [describe\_foreign\_table``(fk_{opt}, l_{optg})] \wedge [describe\_select\_table(l_f, l_{optg})] \wedge [describe\_grouped\_select\_tables(l_{optg}, l_{opt})] \wedge [describe\_option\_table(l_{optg}, s_{opt})] \wedge [form\_table\_relation(e_{opt}, s_{opt})] \wedge [column\_felement\_relation(e_{opt}, s_{opt})] \quad (4.41)$$

Vienareikšmiškai vadovautis ankščiau surašytomis matematinėmis išraiškomis arba duomenimis 4.1 lentelėje, negalima. Tai tik idealizuotas modelis. Pagal situaciją ir turimą užduotį vartotojas turėtų duomenų bazės schemą keisti, kad sistema atitinkamų tam tikrus užsibrėžtus reikalavimus ir tikslus.

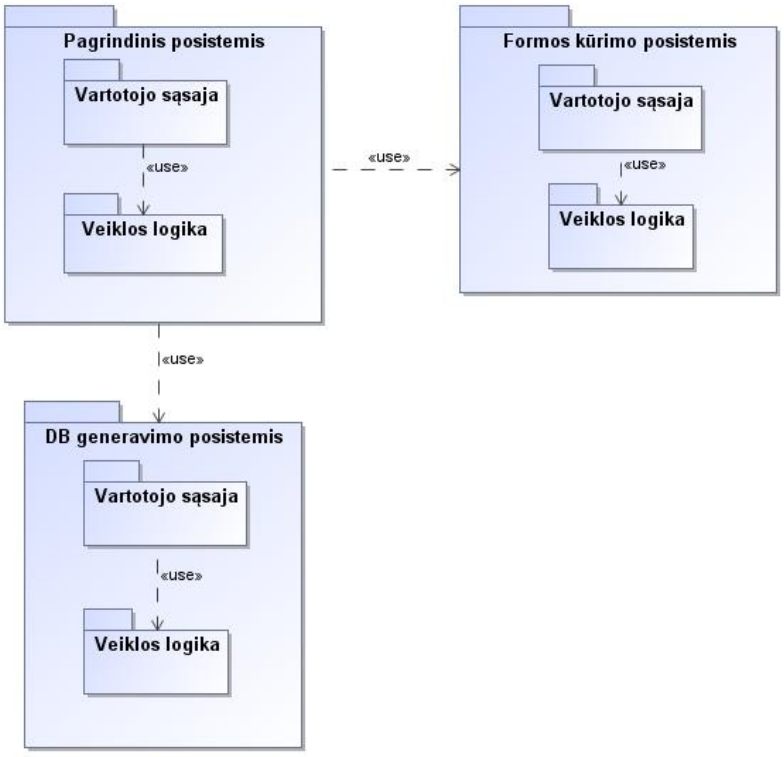
4.2 Sistemos loginė architektūra

Sistema buvo išskirstyta į tris posistemius (4.2 pav.): pagrindinį, formos kūrimo, duomenų bazės generavimo. Šie posistemiai yra susieti su atitinkamomis sistemos veiklos funkcijomis. Visi posistemiai pasiekiami vienam vartotojui.

Pagrindinis posistemis. Pagrindinis posistemis turi visą veiklos meniu, kuriamos duomenų bazės, vartotojo sąsajos ir integravimo į vieną sistemą funkcijos.

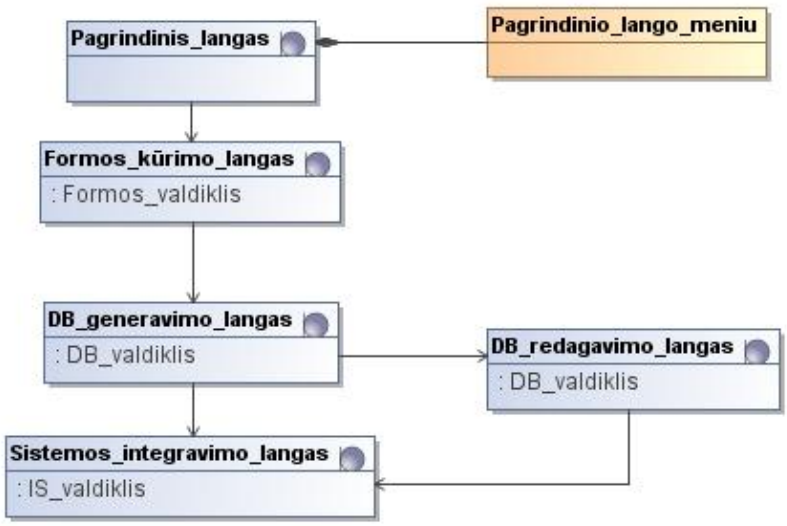
Formos kūrimo posistemis. Posistemis skirtas kurti formas. Jame esančiose klasėse yra aprašomi *HTML* formų elementai, bei jų tipai. Taip pat, įrankiai padėsiantys suskurti paprastą formą.

DB generavimo posistemis. Posistemis skirtas duomenų bazės generavimui iš jau sukurtos formos. Forma sukuriama trečiųjų šalių programinės įrangos pagalba arba formos kūrimo posistemyje. Šiame posistemyje aprašytos klasės apibudina duomenų bazės logiką ir duomenų bazės generavimui iš jau sukurtos formos metodus.



4.2 paveikslas. Sistemos loginė architektūra

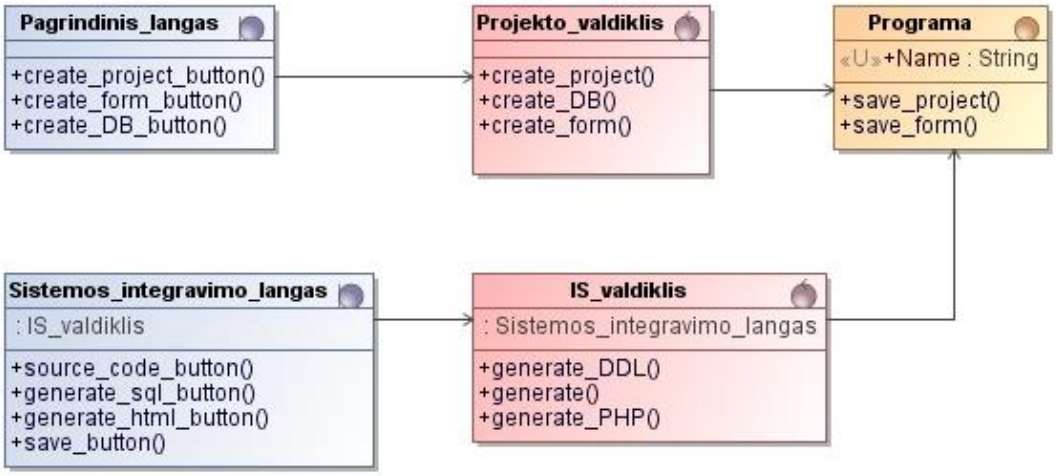
Visi langai pasiekiami iš pagrindinio posistemio vienam sistemos vartotojui. Iš sistemos navigavimo plano (4.3 pav.) galima matyt, kad visi kiti langai yra pasiekiami iš pagrindinio lango, su pagrindinio lango meniu. Skirtingi langai valdo skirtingus sistemos posistemius.



4.3 paveikslas. Sistemos navigavimo planas

4.2.1 Pagrindinis posistemis

Pagrindinis posistemį aprašančios klasės pateiktos (4.4 pav.). Šiame posistemyje valdomas visas kuriamas projektas. Iš šio posistemio galima pasiekti kitus sistemoje esančius posistemius: formos kūrimo ir duomenų bazės generavimo. Šiame posistemyje aprašytos, sistemos integravimo į vieną sistemą, klasės.



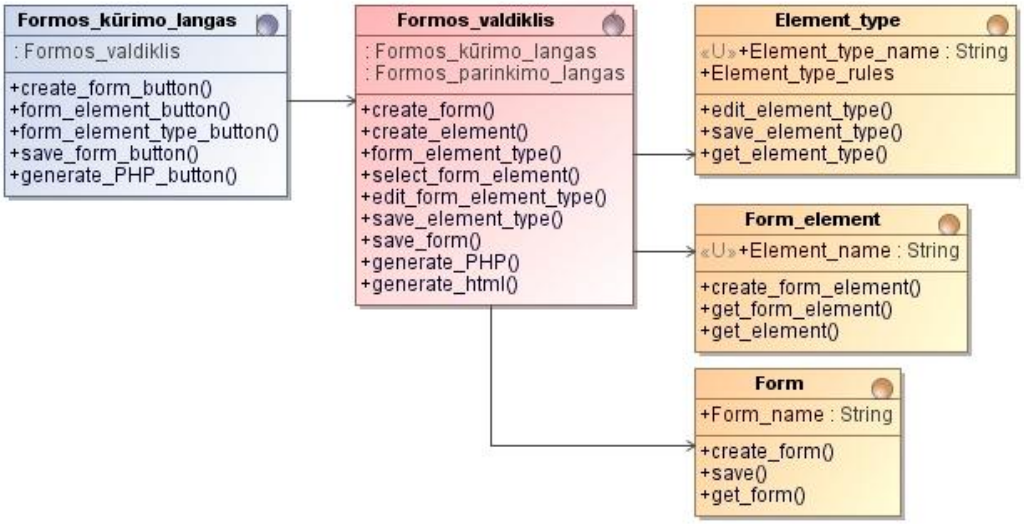
4.4 paveikslas. Pagrindinio posistemio klasių diagrama

Posistemio panaudojimo atvejus realizuojančios klasių diagramos, pateiktos 9.3 priede.

Posistemio panaudojimo atvejų sekų diagramos (9.4 priede). Sekų diagramos apibūdina objektų – klasių veiksmų lygiagretų išsidėstymą laike.

4.2.2 Vartotojo sąsajos kūrimo posistemis

Vartotojo sąsajos kūrimo posistemį aprašančios klasės pateiktos (4.5 pav.). Ši posistemė pasiekiama iš pagrindinio posistemio, meniu pagalba. Vartotojo kūrimo posistemyje aprašytos formos kūrimo klasės.

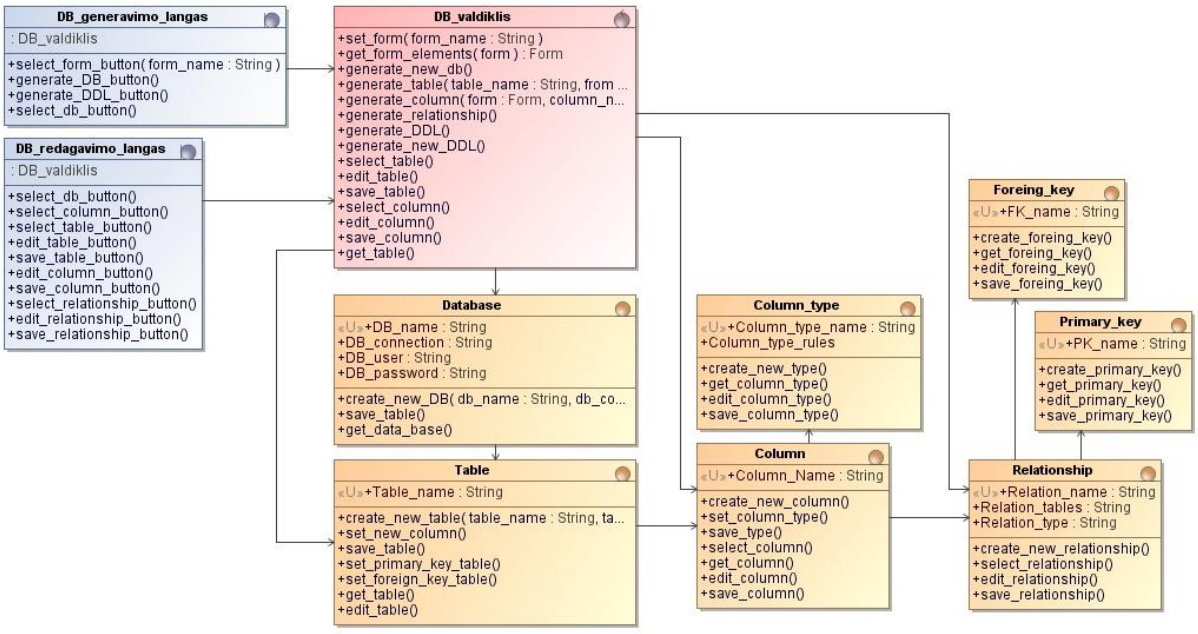


4.5 paveikslas. Vartotojo sąsajos kūrimo posistemis

Panaudojimo atvejų realizacijos pavaizduotos 9.11 paveiksle.
Posistemio panaudojimo atvejus realizuojančios klasių diagramos, pateiktos 9.3 priede.
Posistemio panaudojimo atvejų sekų diagramos pateiktos 9.4 priede.

4.2.3 Duomenų bazės generavimo posistemis

Duomenų bazės generavimo posistemį apibrėžia 4.6 pav. pavaizduotos klasės. Šiose klasėse yra pavaizduoti metodai, kurie atlieka duomenų bazės generavimo veiksmus. Duomenų generavimo posistemyje bus aprašytas metodas, kuris iš jai pateiktos formos generuos duomenų bazę su jai priklausančiomis lentelėmis, stulpeliais ir ryšiais.

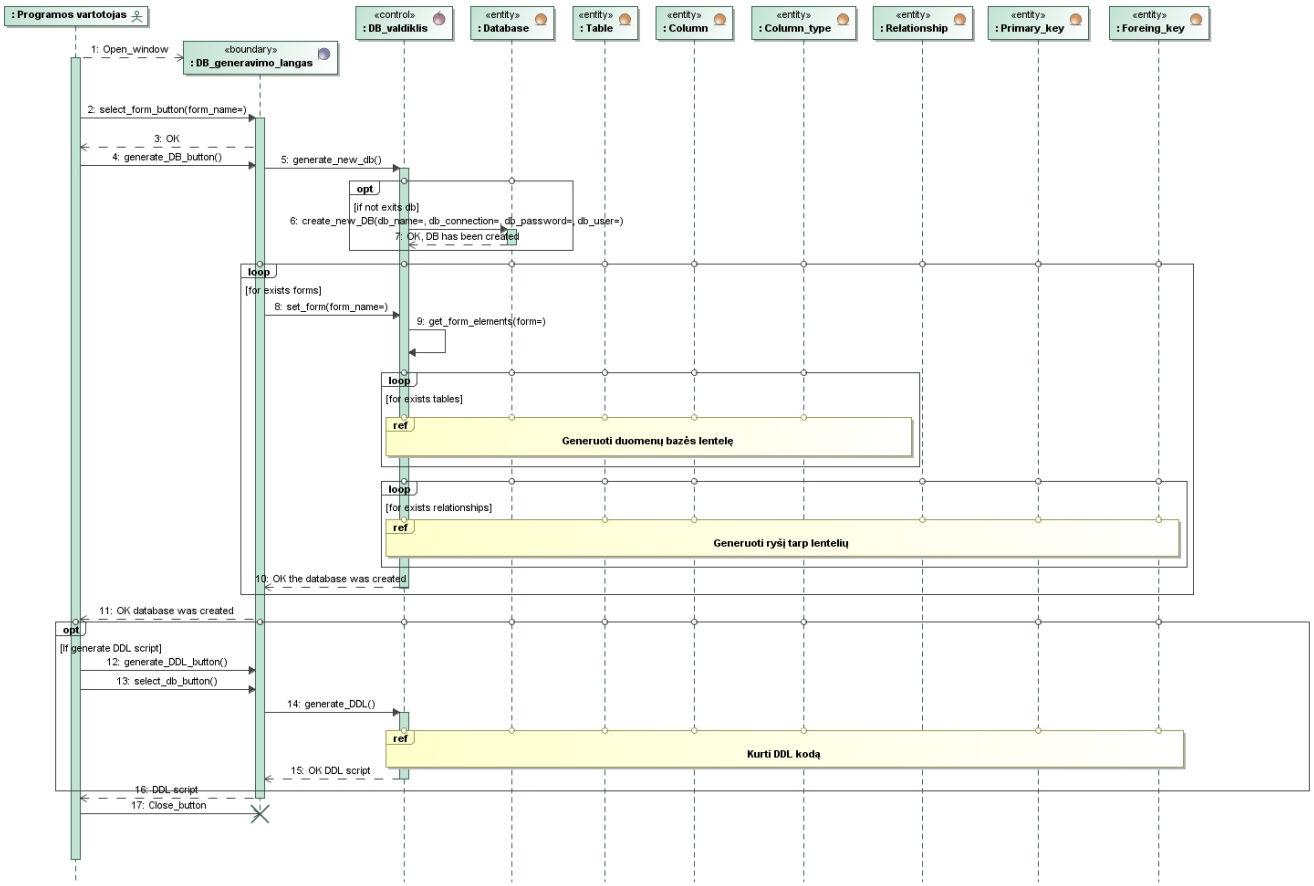


4.6 paveikslas. Duomenų bazės generavimo posistemio klasių diagrama

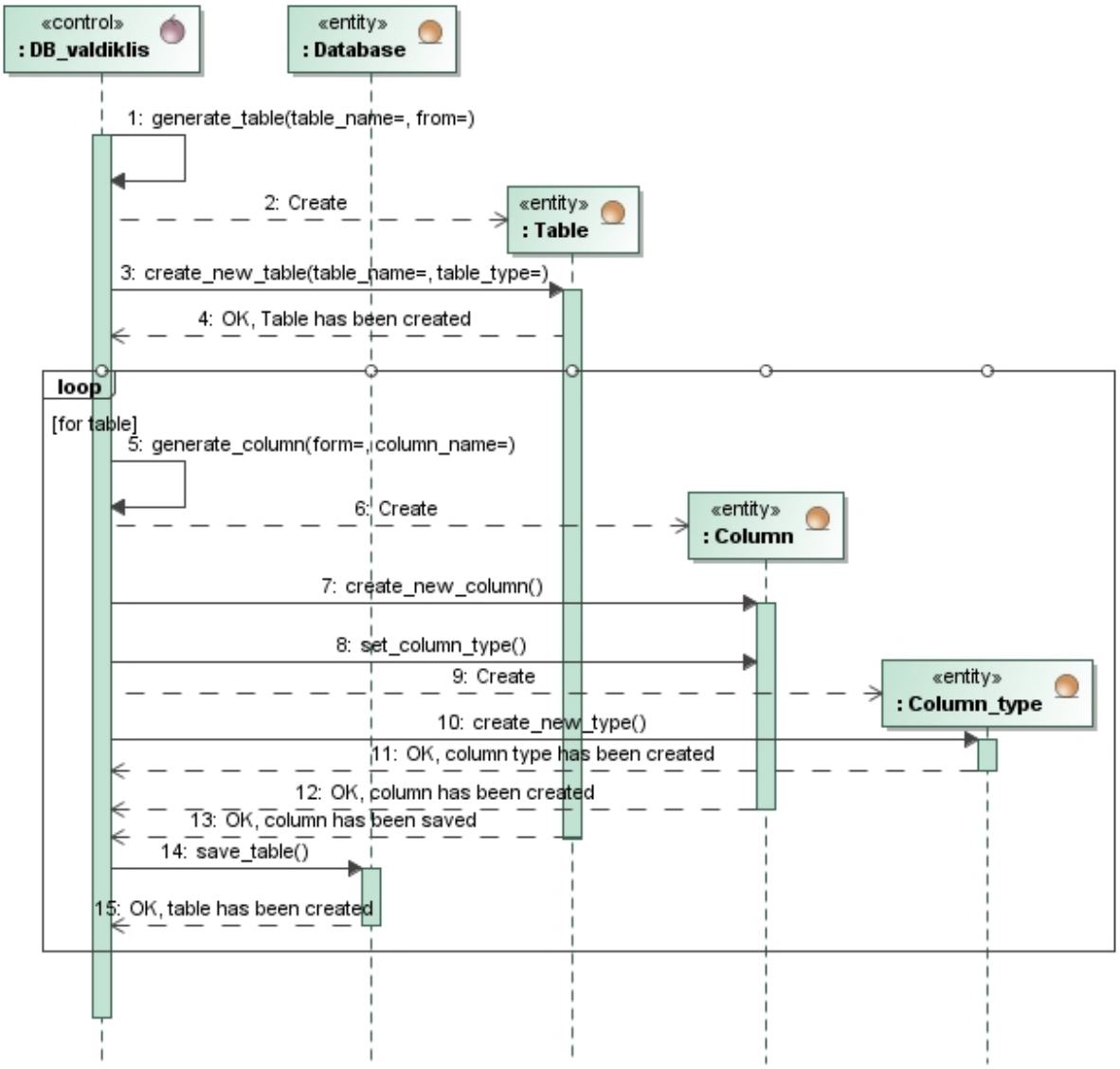
Posistemis realizuoja 9.2 priede pavaizduotus panaudojimo atvejų realizacijos diagramas. Posistemio panaudojimo atvejus realizuojančios klasių diagramos, pateiktos 9.13 priedė.

Posistemio panaudojimo atvejų sekų diagramos pateiktos 4.7 - 4.13 paveiksluose. Pagrindinis panaudojimo atvejis (toliau PA), duomenų generavimo posistemyje, yra „Kurti duomenų bazę“ (4.7 pav.) ir kartu su šiuo panaudojimo atveju susijusiais PA: „Generuoti duomenų bazės lentelę“ (4.8 pav.)

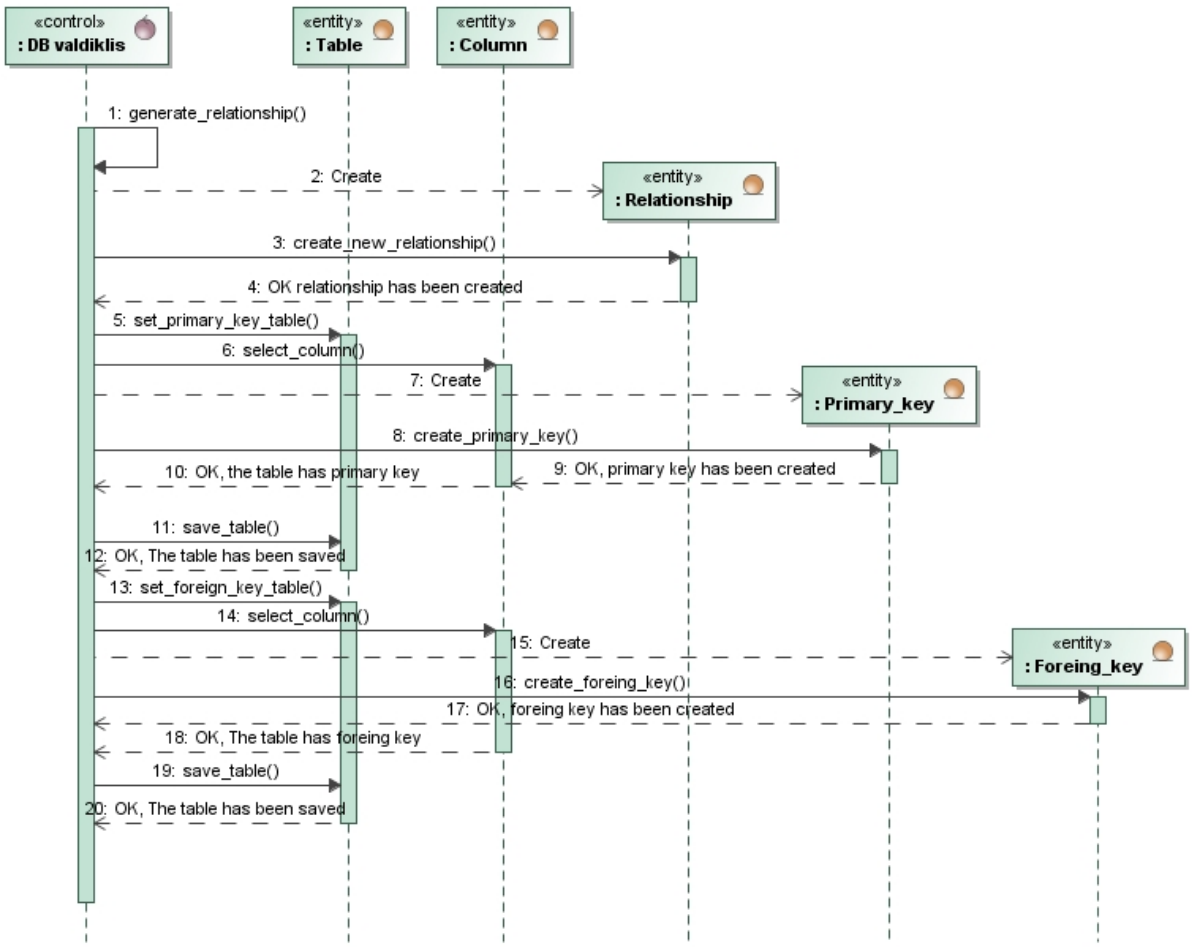
ir „Generuoti ryšį tarp lentelių“ (4.9 pav.). Šiuose PA atskleidžiama, sistemos elgsena generuojant duomenų bazę.



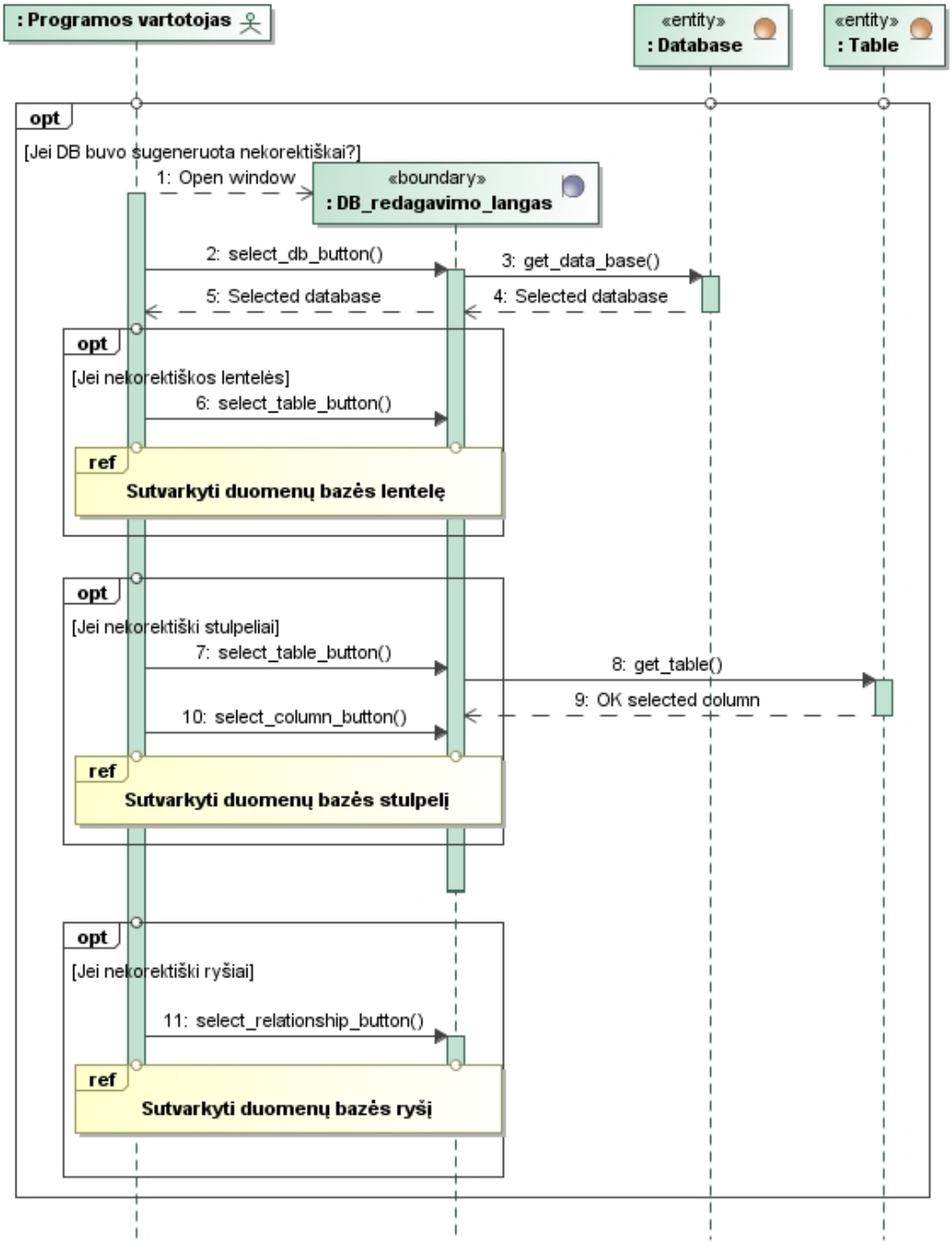
4.7 paveikslas. PA „Kurti duomenų bazę“ sekų diagrama



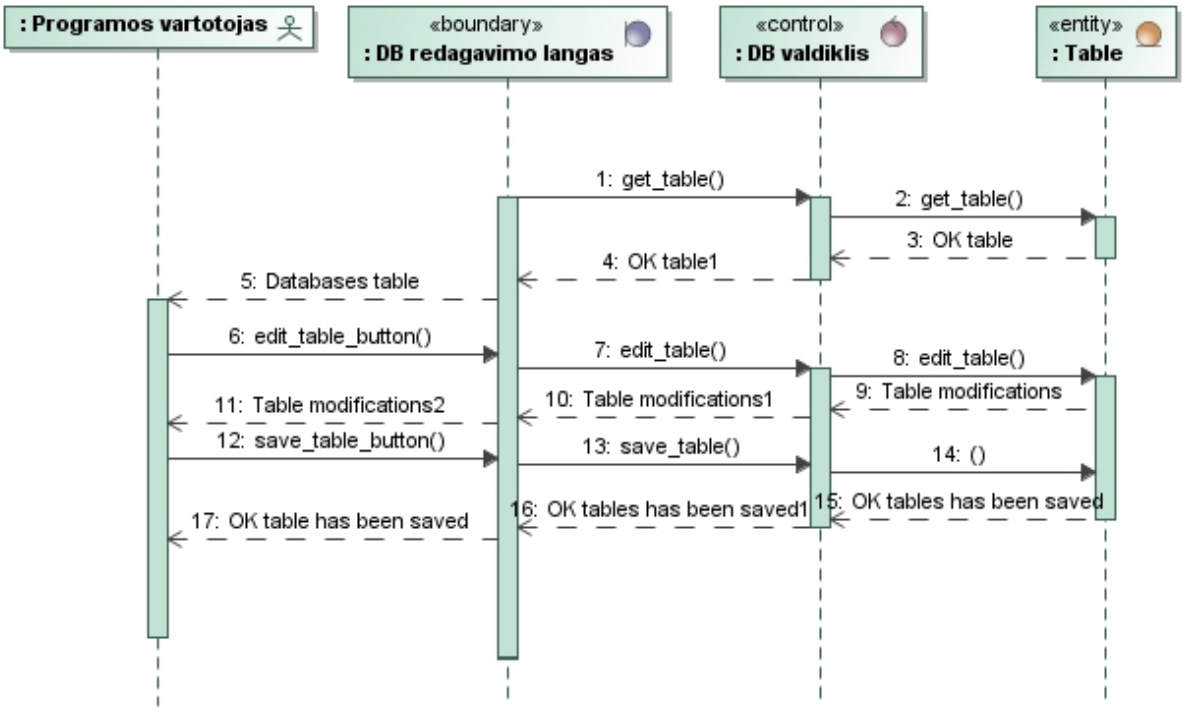
4.8 paveikslas. PA „Generuoti duomenų bazės lentelę“ sekų diagrama



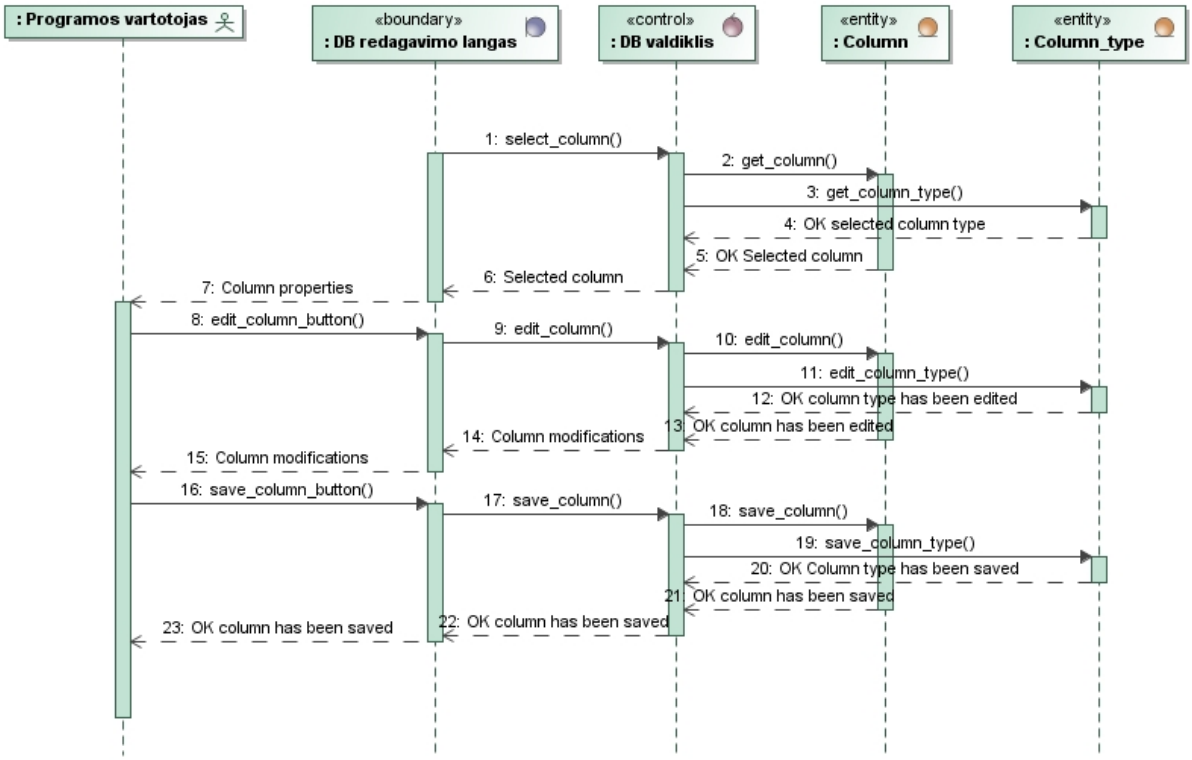
4.9 paveikslas. PA „Generuoti ryšį tarp lentelių“ sekų diagrama



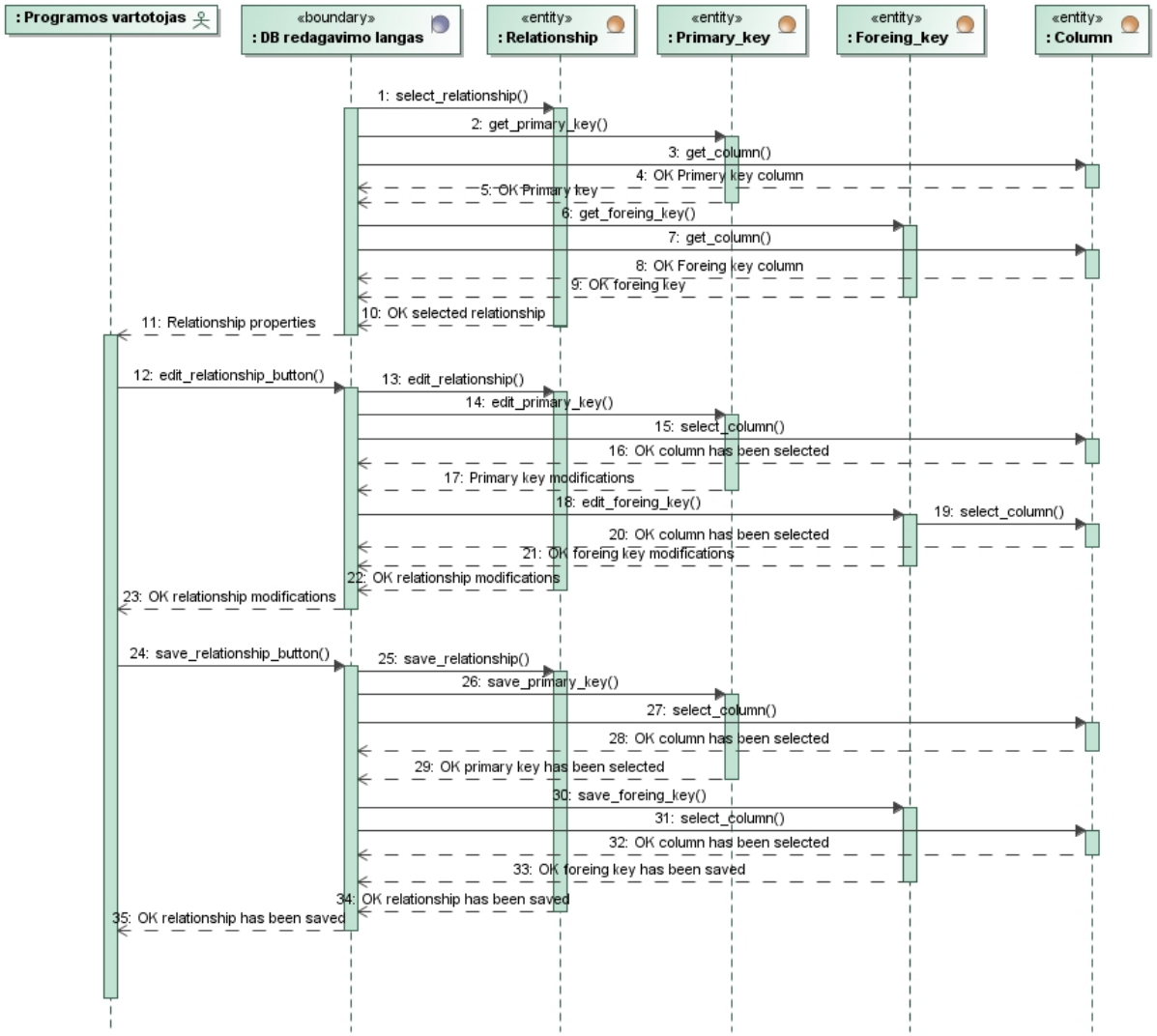
4.10 paveikslas. PA „Redaguoti duomenų bazę“ sekų



4.11 paveikslas. PA „Sutvarkyti duomenų bazės lentelę“ sekų diagrama



4.12 paveikslas. PA „Sutvarkyti duomenų bazės stulpelį“ sekų diagrama



4.13 paveikslas. PA „Sutvarkyti duomenų bazės ryšį“ sekų diagrama

Iš klasių realizacijos ir panaudojimo atvejų sekų diagramų nėra matoma kaip generuojama duomenų bazė, iš sistemai pateiktos formos. Dėl to buvo išskirti metodai susiję su magistrinio darbo tikslais: *get_form_elements*, *generate_table*, *generate_column*, *generate_relationship*. Šiems metodams buvo nubraižytos atskiros veiklos diagramos (4.14 - 4.16 pav.), kuriose atsiskleidžia sistemos elgsena, skaitant formą ir kuriant duomenų bazės schemą.

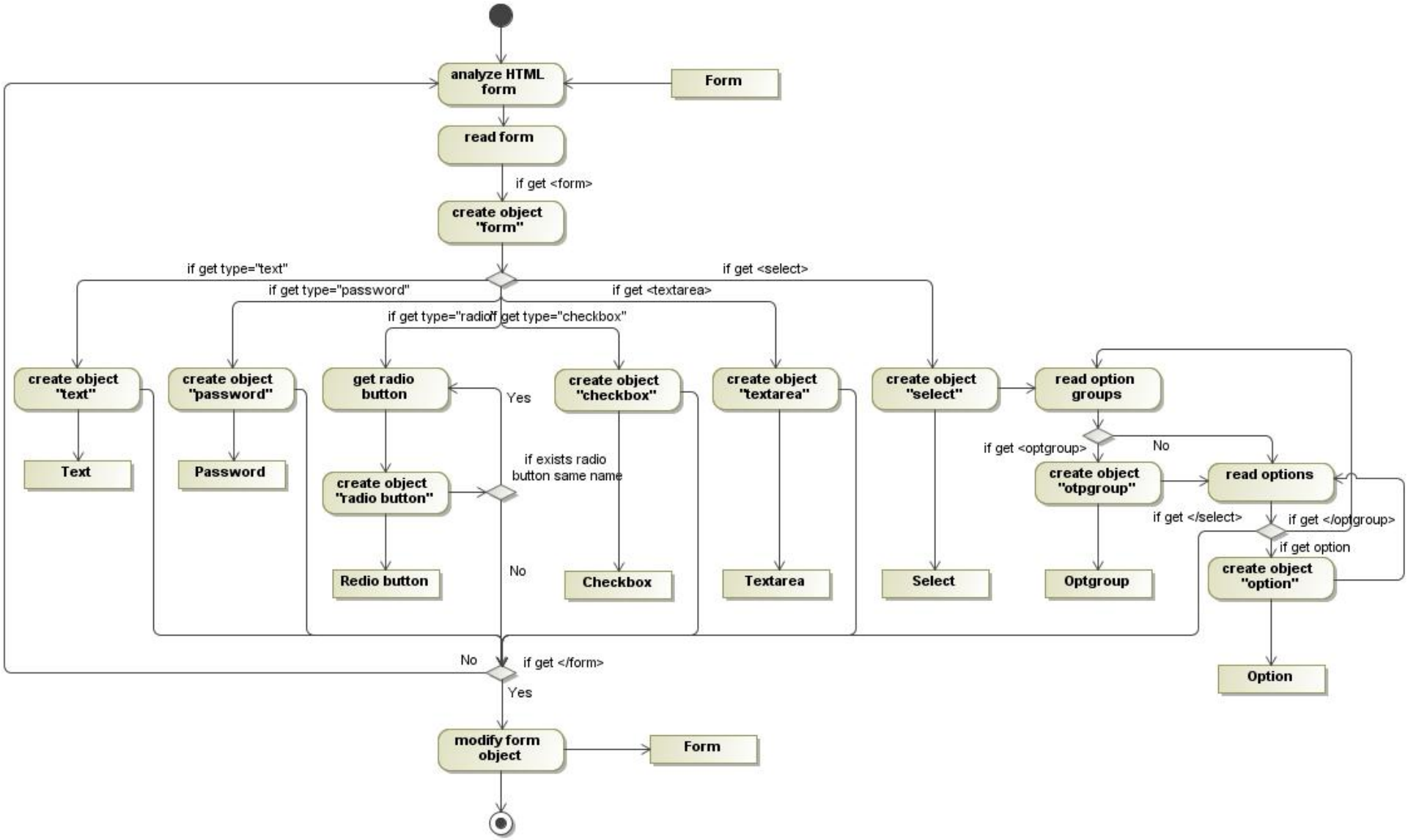
4.3 Duomenų bazių generavimo metodai

Metodas *get_form_elements*.

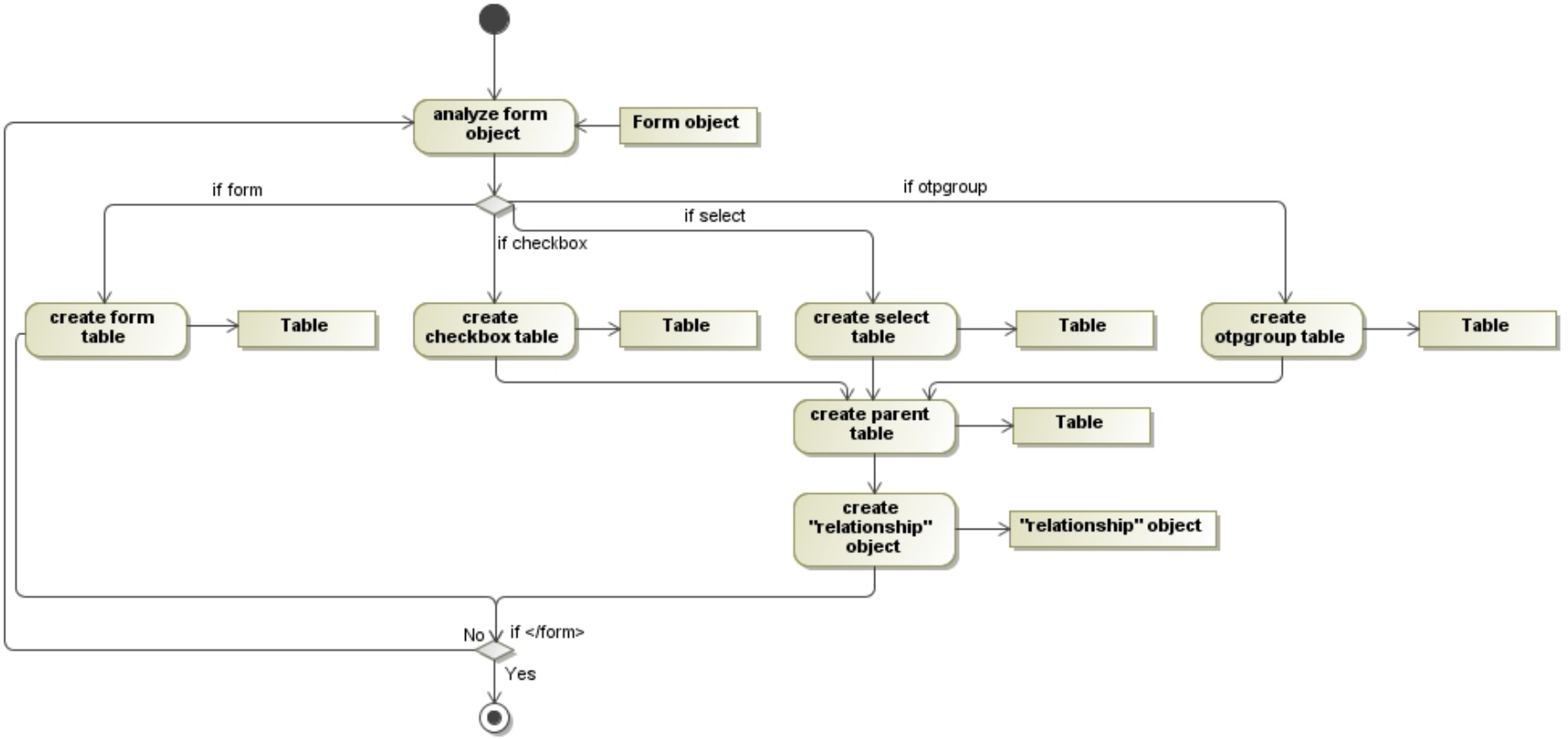
Metodas nuskaito formą ir jose esančius elementus ir sukuria objektus, kurie atitinka formos elementus. Šie objektai perduodami kitiems sistemos metodams tolesniam apdorojimui. Metodo veiklos diagrama pavaizduota 4.14 paveiksle.

Metodas *generate_table*

Metodas, sekų diagramose, vykdomas po formos elementų išgavimo iš pateiktos formos. Šiame metode sukuriami lentelių objektai. Jei elementas yra su daugiau reikšmių pavyzdžiui: „*combo box*“ *HTML* formose aprašomas, kaip <*select*>. Metodas dar sukuria ir ryšio objektą kuriame pagrindinė – „tėvinė“ lentelė yra pirma sukurta formos lentelė. O vaikinė lentelė sukuriama iš <*option*> atributų. Visoms lentelėms ir ryšiams sukuriami objektai, kurie vėliau naudojami kitų metodų. Metodo veiklos diagrama pavaizduota 4.15 pav.



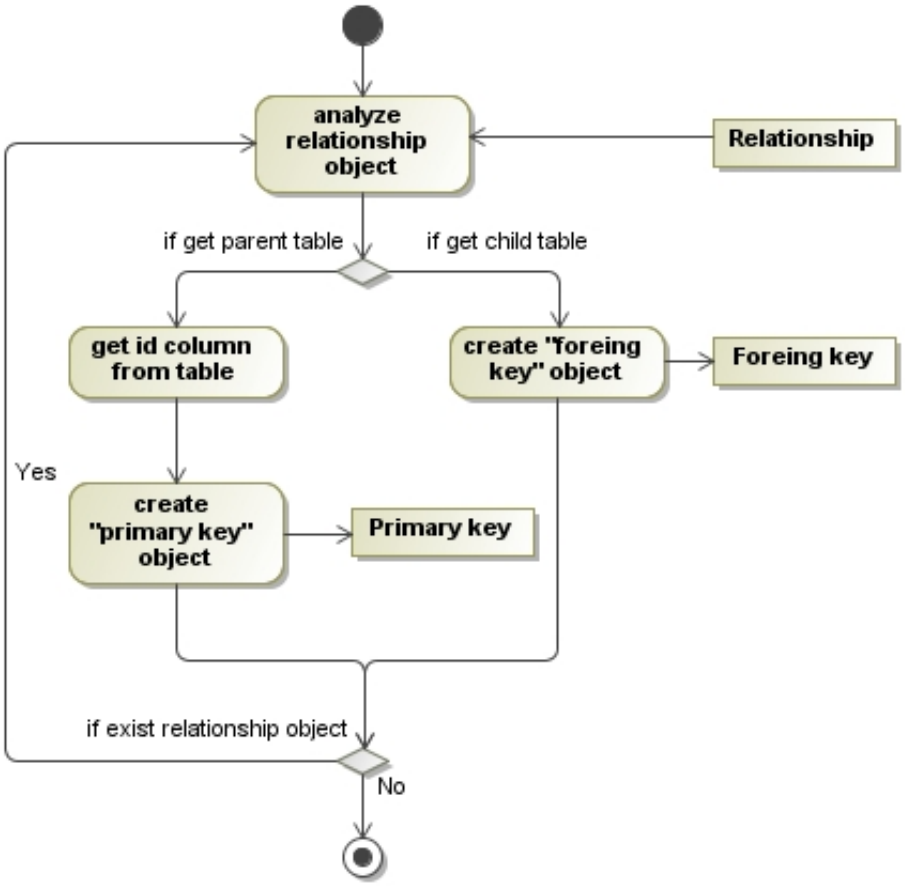
4.14 paveikslas. Metodo „get_form_elements“ veiklos diagrama



4.15 paveikslas. Metodo „generate_tables“ veiklos diagrama.

Metodas *generate_relationship*

Metodas skirtas kurti duomenų bazės ryšius. Metodui perduodamas *relationship* objektas jo pagalba sukuriami pirminio ir išorinio rakto objektai. Šie objektai sukuria pirminius ir išorinius raktus duomenų bazės raktus. Taip lentelės yra sujungiamos ryšiu. Sekų diagramoje matome, kad metodas vykdomas po lentelių ir stulpelių sukūrimo. Metodo veikimas pavaizduotas veiklos diagrama 4.16 pav.



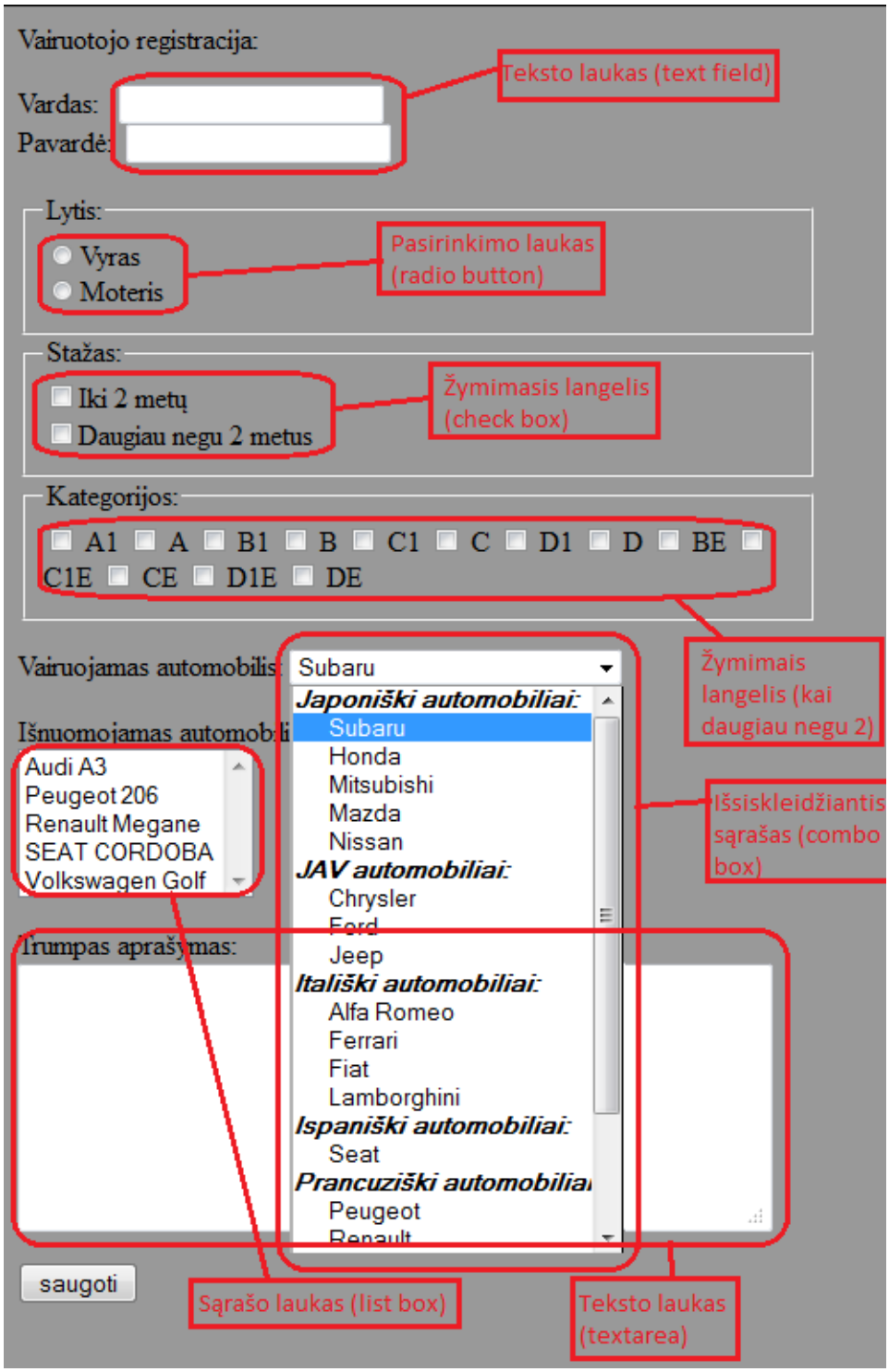
4.16 paveikslas. Metodo „generate_relationship“ veiklos diagrama

4.4 Duomenų bazės generavimo metodų apibendrinimas

Metodų veikimo principą apibūdina ir patikslina, pateikta 4.1 sprendimų lentelė (angl. decision table). Lentelėje yra surašyti visi galimi *HTML* formos elementai ir sprendimai kaip tūrėtų elgtis sistema aptikus vieną ar kitą elementą. Kadangi *grid* tipo elementai yra *HTML* formose pavaizduojami kaip lentelė (angl. *table*), dėl to *gird* tipo elementas sprendimų lentelėje yra pažymėtas.

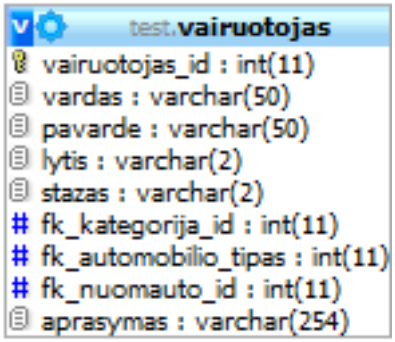
Tarp galimų sprendimų pateikiami galimi du ryšiai: formos lentelė – nauja lentelė ir nauja lentelė – nauja lentelė. Sukūrus naują lentelę, kuri jungiasi su jau sukurta kita lentele (ne formos lentele). Sprendimų lentelėje pažymėta „Naujos lentelės stulp. PK (angl. *primary key*) ir naujos lentelės stulp. FK (angl. *foreign key*)., tai šiuo atveju PK yra naujos lentelės pirminio rakto stulpelis, o FK yra senos lentelės išorinio rakto stulpelis.

Siekiant patikslinti Sprendimų lentelės (4.1 lentelė) surašytus sprendimus ir sistemos veiksmus, buvo sukurta pavyzdinė vairuotojo registravimo forma (4.17 pav. ir 9.6 priedas) su jos duomenis saugančia duomenų baze (4.22 pav.).



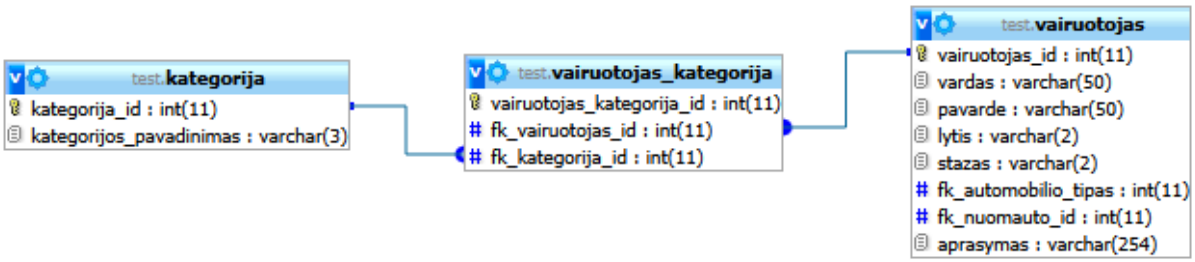
4.17 paveikslas. HTML formos pavyzdys "vairuotojų registracija"

Sistema atpažinus formą, sukuria jai lentelę (4.18 pav.), pavadinimu nuskaitytu iš kodo, ir sukuria jai pirminio rakto atributą. Atpažinus formoje esančius teksto laukus (angl. *text field*), formos lentelėje sukuriami atributai „*vardas*“ ir „*pavardė*“. Atributams suteikdama vardus, kuriuos atpažįsta iš laukams kode suteiktų pavadinimų (angl. *name*). Aptikus sekantį elementą, pasirinkimo laukus, sistema turėtų juos identifikuoti kaip esančius iš vienos grupės. Tai padeda nustatyti, taip pat jiems kode suteikti vardai. Jei vardai vienodi, pavyzdžiui *name=”lytis”*, tai šiems formos elementams sukuriama duomenų bazės atributas lytis su galimomis reikšmėmis (angl. *value*): vyras ir moteris. Atpažinus žymimojo lauko formos elementą, pavadinimu „stazas“ duomenų bazės lentelėje sukuriamas stulpelis, kurio galimos reikšmes yra „iki2metu“ ir „daugiau“.



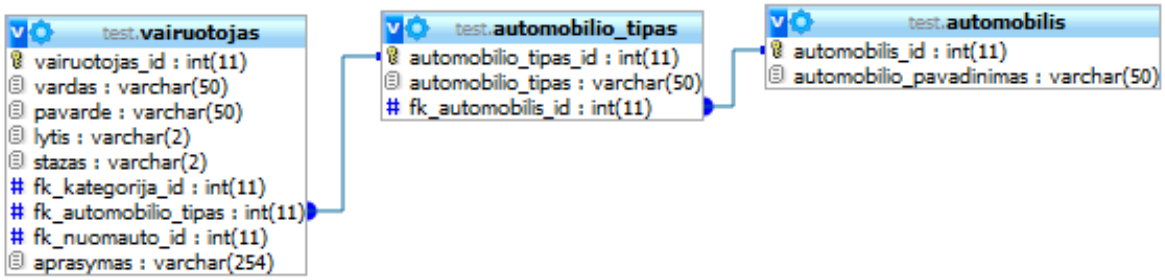
4.18 Paveikslas. Vairuotojas lentelė

Jeigu žymimojo lauko elementų tuo pačiu pavadinimu yra daugiau negu 2 ir juos galima pažymėti du ir daugiau. Tai šiam elementui kuriamos dvi naujos lentelės (4.19 pav.) su visais atitinkančiais stulpeliais, bei pirminiais raktais. Formos lentelėje sukuriamas naujas išorinio rakto stulpelis jungiantis formos lentelę *„vairuotojas“* su nauja lentele *„vairuotojas_kategorija“*. *„vairuotojas_kategorija“* - tarpinė lentelė , turi du išorinius raktus, kurie sujungia *„kategorijos“* ir *„vairuotojas“* lenteles. Šias lenteles, formos lentelė ir naujas sukurtas, sujungiant ryšiais.



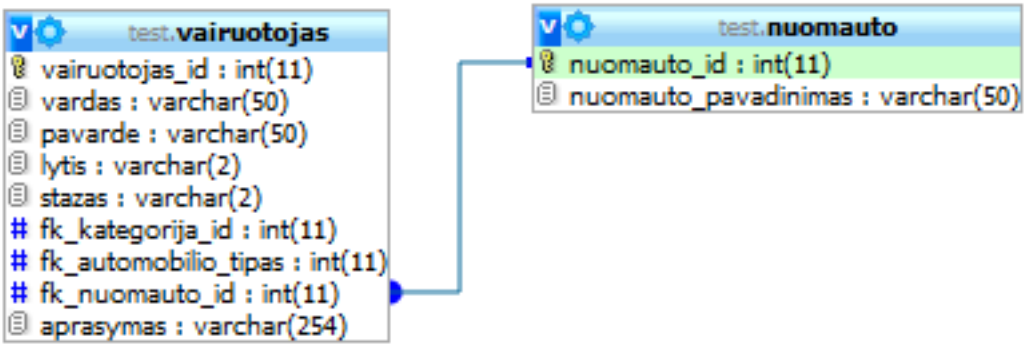
4.19 paveikslas. Lentelė „vairuotojas“ ir lentelė „kategorija“

Jei sistema aptinka išsiskleidžiantį sugrupuotą sąrašą (4.17 pav.), duomenų bazėje kuriamos dvi lentelės, duotu atveju: *„automobilio_tipas“* ir *„automobilis“* (4.20 pav.). Lentelėse sukuriami atitinkami pirminiai raktai ir kiti reikalingi atributai. Formos lentelėje „vairuotojas“, sukuriama išorinis raktas jungiantis su automobilio tipo lentele. „Automobilio_tipas“ lentelėje analogiškai, sukuriamas išorinis raktas jungiantis su lentele „automobilis“.



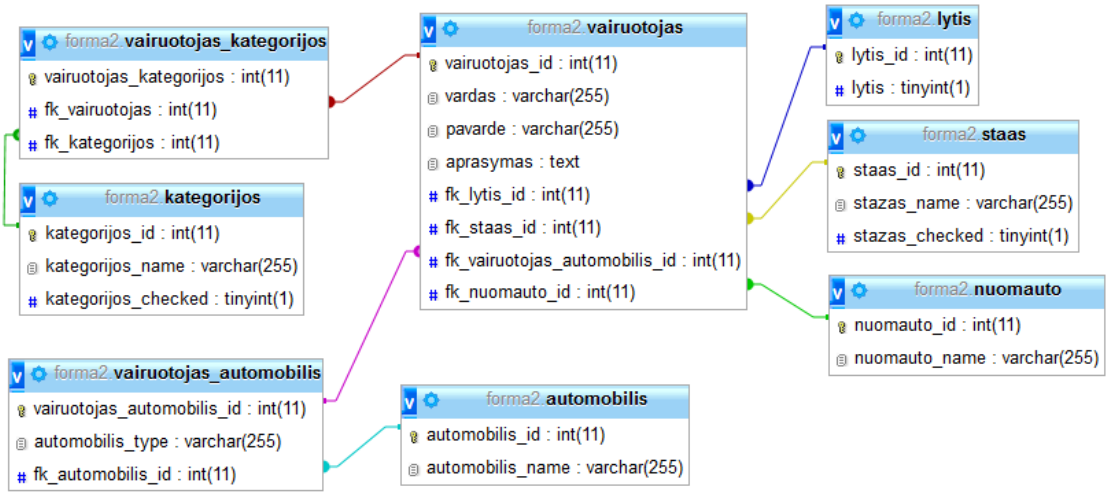
4.20 paveikslas. Lentelės „vairuotojas“, „automobilio_tipas“, „automobilis“

Sistemai aptikus sąrašo lauką, kuriama lentelė (4.21 pav.) tokiu pat pavadinimu kaip nurodyta kode, su jai atitinkamais laukais: pirminio rakto ir sąrašą apibudinančiais laukais. Tarp formos ir naujai sukurtos lentelės sukuriamas ryšys, į formos lentelę įrašant išorinio rakto stulpelį.



4.21 paveikslas. Lentelė „vairuotojas“ ir lentelė „nuomauto“

Formos duomenis saugančios visa duomenų bazės schema pateikta 4.22 paveiksle.



4.22 paveikslas. Formos „vairuotojų registracija“ duomenų bazės schema

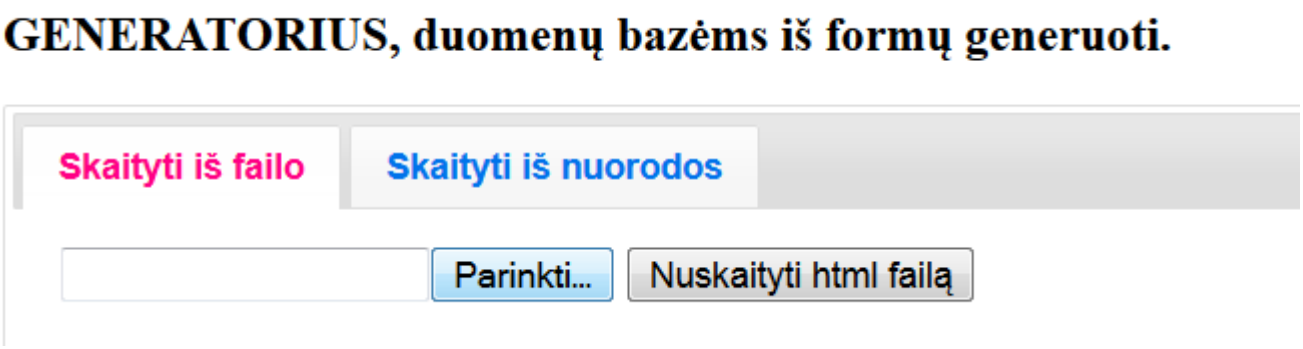
5 DUOMENŲ BAZĖS GENERAVIMO IŠ EKRANINĖS FORMOS SISTEMOS REALIZACIJA

5.1 Sistemos veikimo aprašas

Sukurtos duomenų bazės generavimo iš formos sistemos veikimas susideda iš trijų etapų:

1) nuskaitoma forma;
2) generuojama duomenų bazė;
3) sugeneruota duomenų bazė pateikiama vartotojui patogia forma (SQL script, sukuriama duomenų bazė pasirinktoje duomenų bazių valdymo sistemoje).

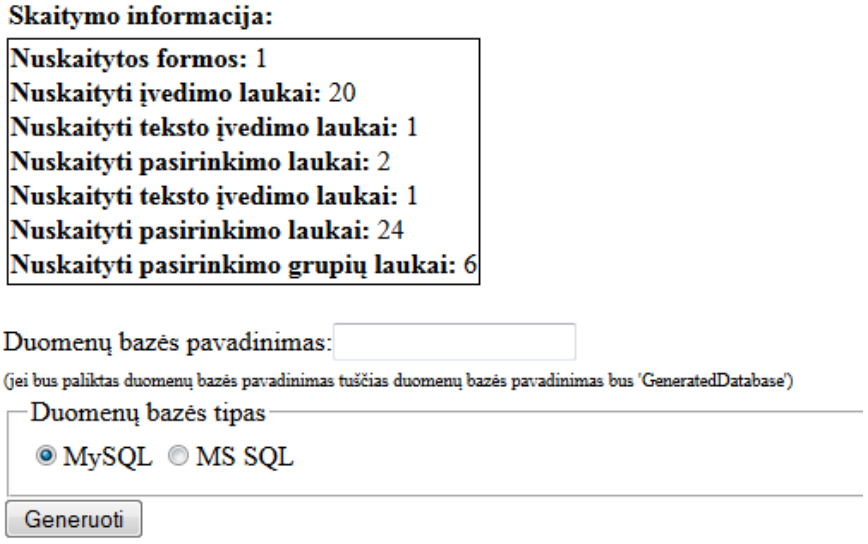
Visi etapai atliekami iš eilės.



5.1 paveikslas. Programos valdymo menių

Programa valdoma nesudėtingai (5.1 pav.). Galima skaityti formą tiek iš HTML failo, tiek iš pateiktos nuorodos. Įvedus formą spaudžiame mygtuką „Nusktaityti html failą“ arba „Nuskaityti html linką“.

Nusktaičius formą gaunama suvestinė (5.2 pav.) ir sekančių žingsnių nurodymai.



5.2 paveikslas. Nuskaitytos formos suvestinė

Į duomenų bazės pavadinimo lauką turi būti nurodoma duomanų bazės schemos pavadinimas. Nenurodžius šio parametro sistema sukurs numatytąjį pavadinima – „GeneratedDatabase”. Pasirinkus kokio tipo duomenų bazės schema bus generuojama spaudžiame mygtuką „Generuoti“. Kai bus sugeneruota duomenų bazės schema atsiras mygtukas „Gauti SQL kodą“. Paspaudus šį mygtuką bus pavaizduotas *SQL* kodas.

SQL kodo apačioje yra mygtukas „Sukurti SQL failą“. Šio mygtuko pagalba vartotojas gali susikurti, prieš tai sugeneruoto, *SQL* kodo failą, kurį skaito pasirinkto tipo DBVS.

5.1.1 Formos skaitymas

Puslapio nuskaitymui buvo pasirinkta *Document Object Model* (*DOM*) biblioketą[23]. Ši biblioteka pasirinkta, nes yra paprasta, nesudėtinga ir lengvai valdoma. *DOM* naudojama daugelyje interneto sistemų, dėl to, sukurtas įrankis veikia *HTTP* protokolo pagalba ir paprastas vartotojas gali pasiekti šią eksperimentinę sistemą.

HTML failas pirmiausiai yra nuskaitomas. Formos elementai yra filtruojami nuo kitų, programiniame kode esančių elementų. Išfiltravus elementus, juos saugome objektuose, kuriuose yra nurodomas šio elemento tipas, pavadinimas ir kokio tipo duomenys yra saugomi.

5.1.2 Duomenų bazės generavimas

Buvo sudarytas algoritmas, šiam uždaviniui išspręsti. Algoritme nuskaityti formos elementai įrašomi į sąrašus elementų grupių, pagal jų tipus. Šie sąrašai naudojami atskirų elementų lentelėms formuoti.

Aptikęs faile formą, sugeneruojamas SQL kodas, duomenų bazei sukurti. Pradžioje iš formos išrenkami elementai:

- *INPUT* - įvedimo laukai:
  - tekstas (angl. *text*);
  - kodas (angl. *password*);
  - žymimasis (angl. *checkbox*) jei yra mažiau negu 2;
  - pasirinkimo (angl. *radio*);
- *TEXTAREA* – teksto lauko elementai.

  Visų kitų tipų elementams kuriamos naujos lentelės, su juose esančiais stulpeliais.
- Išsiskleidžiantis arba sąrašo laukas (angl. combo box or list box);
- Sąrašo grupavimas (angl. option group);

Sekančiu veiksmu įrankis, pagal 4.1 lentelės taisykles, generuoja duomenų bazę: sukuria lenteles, stulpelius ir ryšius.

6 EKSPERIMENTINIS GENERATORIAUS TYRIMAS

6.1 Eksperimento planas

Eksperimento tikslas yra ištirti duomenų bazių generavimo algoritmą prie įvairių sąlygų. Norint ištirti sukurtą programą buvo pagamiti keli HTML failai su formomis:

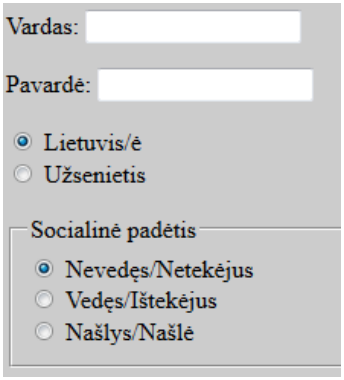
1. Paprastas HTML failas su viena paprasta forma.
2. Sudėtingesnis HTML failas su viena forma, išnaudoti visi įmanomi formos elementai.
3. Sudėtingas HTML failas su keliomis formomis, panaudojant visus įmanomus elementus.
4. Du HTML failai su formomis.

Kiekvienai formai, su eksperimentine sistema, bus sugeneruotas *SQL* kodas. Kodas paleistas ant atitinkamo *DBVS*. Gauta schema bus lyginama su eksperto sukurta schema. Taip bus nustatyta kiek eksperimentinė sistema tiksliai sugeneruoja duomenų bazės schemą.

6.2 Eksperimento rezultatai

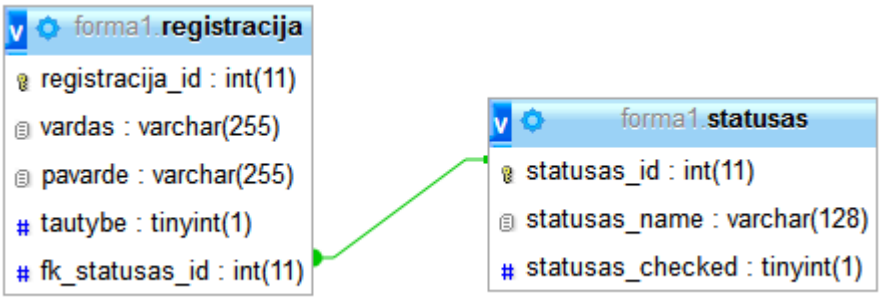
Buvo atlikti trys eksperimentai su trimis skirtingomis formomis, kuriose panaudoti įvairūs elementai.

Pirmuoju atveju paduota 6.1 paveiksle pavaizduota forma. Forma yra paprasta, nesudėtinga, naudojama nedaug elementų(9.5 priedas). Eksperimento tikslas patikrinti kaip veikia eksperimentinė sistema.



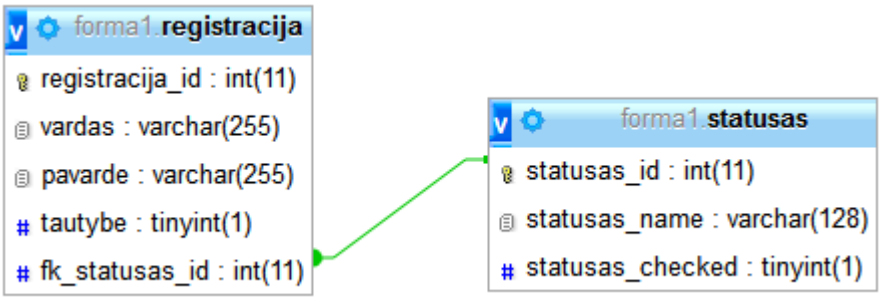
6.1 paveikslas. Paprasta forma

Gautas rezultatas 6.2 paveiksle pavaizduota duomenų bazės schema. Schema susideda iš dviejų lentelių, „registracija“ ir „statusas“.



6.2 paveikslas. Iš paprastos formos sugeneruota duomenų bazės schema

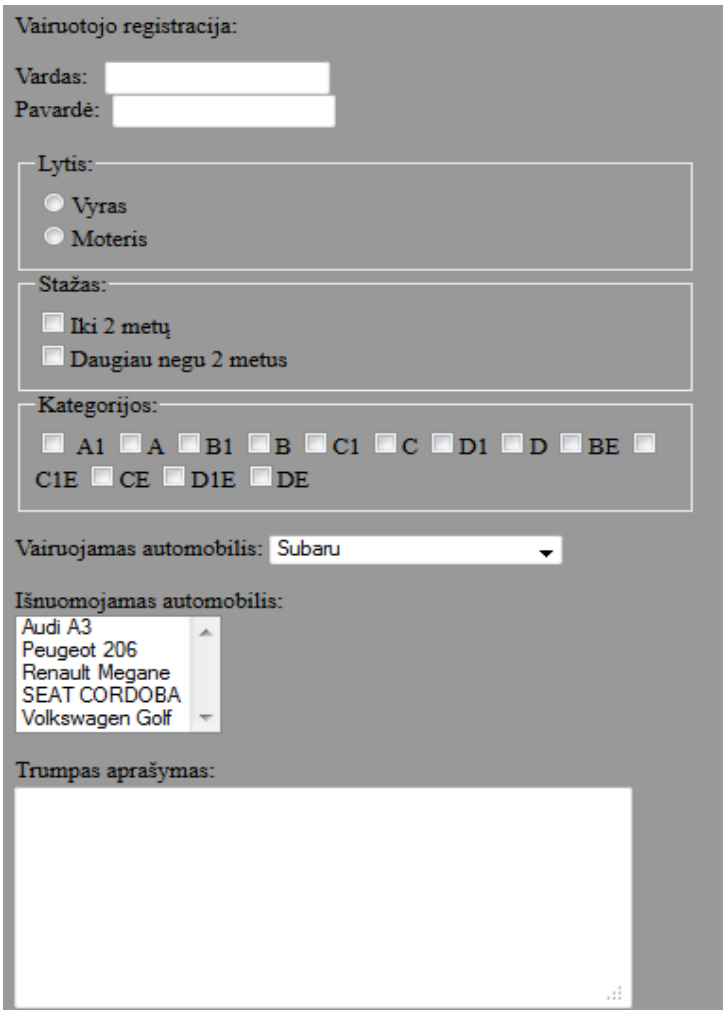
Duomenų bazės schema sukurta eksperto (6.3 pav.).



6.3 paveikslas. Eksperto suprojektuota duomenų bazės schema.

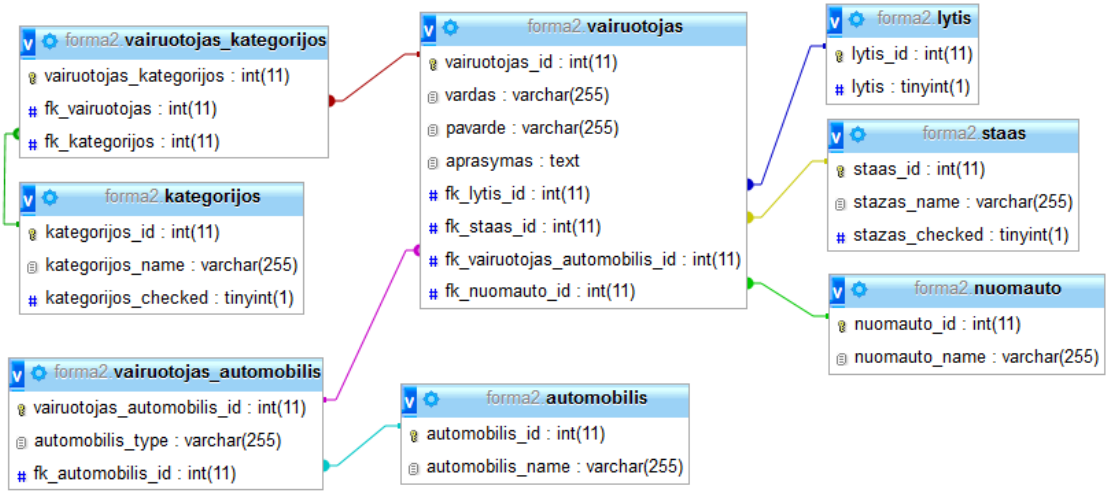
Eksperto suprojektuota duomenų bazės schema, nieko nesiskiria nuo eksperimentinės sistemos suprojektuotos duomenų bazės schemos. Forma nebuvo sudėtinga, dėl to nėra didelių skirtumų.

Antruoju bandymu buvo panaudota 6.4 paveiksle pavaizduota forma (formos kodas yra 9.6 priede).



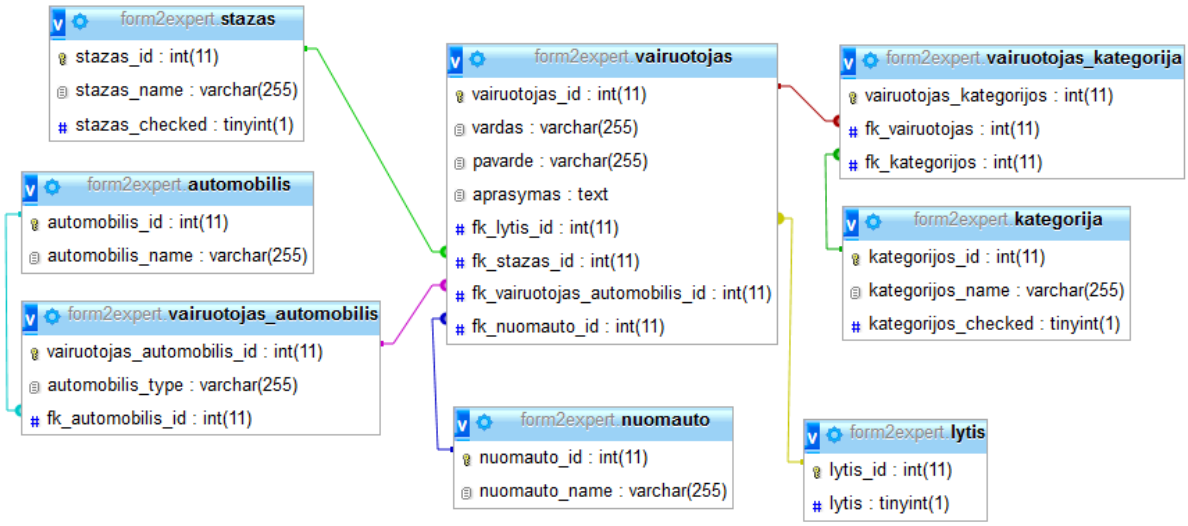
6.4 paveikslas. Forma „vairuotojo registracija“

HTML failas yra sudėtingesnis, nes panaudoti visi įmanomi *HTML* formų elementai. Gautas rezultatas 6.5 paveiksle pavaizduota duomenų bazės schema. Sugeneruotos 7 lentelės: „lytis“, „staas“, „vairuotojas_kategorijos“, „kategorijos“, „vairuotojas_automobilis“, „automobilis“, „nuomauto“. Tarp atitinkamų lentelių sukuriami ryšiai.



6.5 paveikslas. Iš formos „vairuotojo registracija“ sugeneruota duomenų bazės schema.

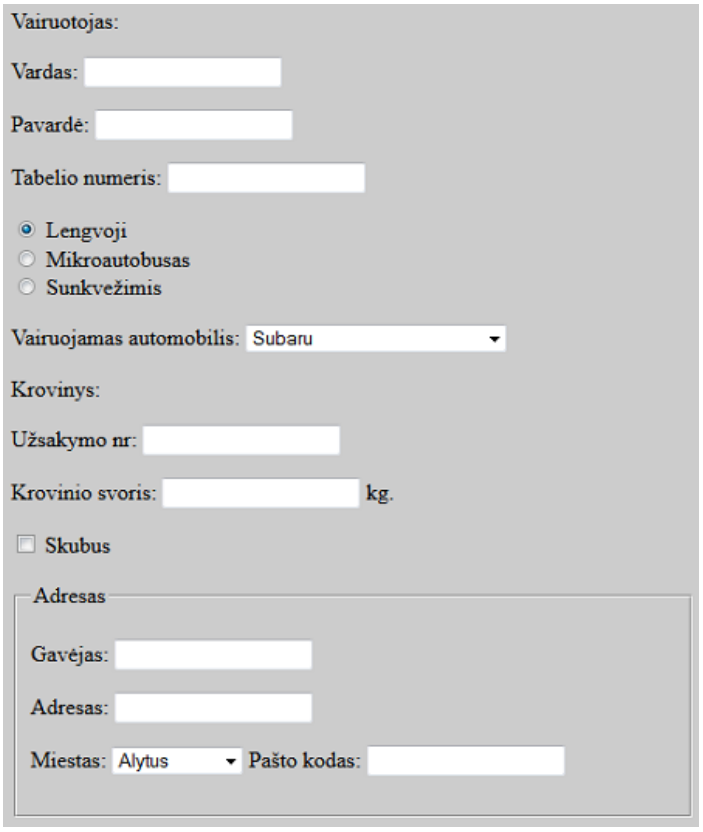
Duomenų bazės schema sukurta eksperto (6.6 pav.). Suprojektuotos 7 lentelės: „lytis“, „stazas“, „vairuotojas_kategorija“, „kategorija“, „vairuotojas_automobilis“, „automobilis“, „nuomauto“. Tarp atitinkamų lentelių sukuriami ryšiai.



6.6 paveikslas. Eksperto suprojektuota duomenų bazės schema.

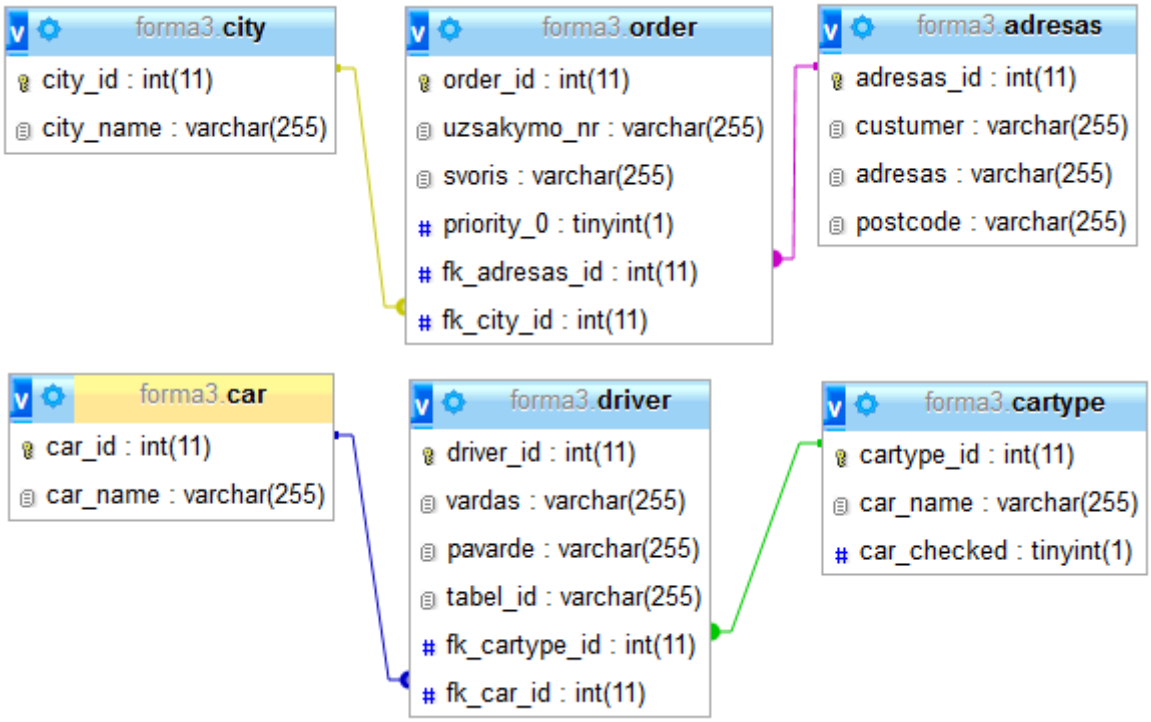
Sugeneruota duomenų bazė schema mažai skiriasi nuo eksperto suprojektuotos schemos. Eksperto projektuojamoje duomenų bazės schemoje galima rinktis kitus stulpelių tipus. Taip pat pavadinimai projektuojami pilnesni, nenaudojami iš kodo paimti pavadinimai.

Trečiuoju bandymu panaudota 6.7 paveiksle pavaizduota forma (formos kodas 9.7 priede) Šitame faile yra panaudotos dvi formos. Tarp kurių turėtų būti ryšys.



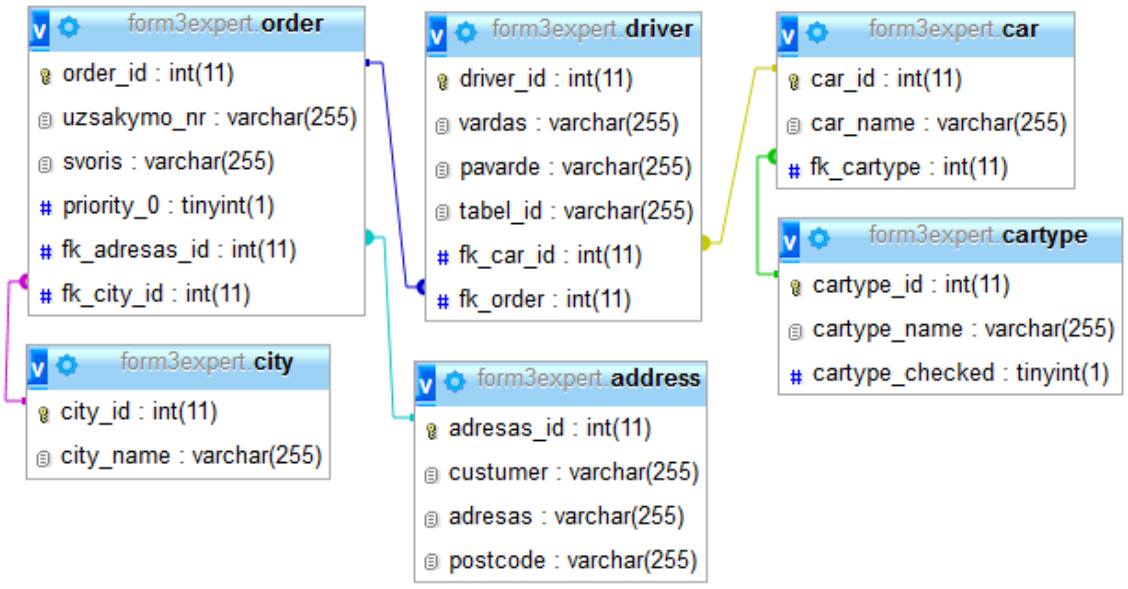
6.7 paveikslas. „Vairuotojo ir krovinio registracija“ formos pavyzdys.

Gautas rezultatas 6.8 paveiksle pavaizduota duomenų bazės schema. Sugeneruota duomenų bazė susideda iš 6 lentelių: „adresas“, „order“, „city“, „cartype“, „driver“, „car“. Tarp lentelių sukurti ryšiai.



6.8 paveikslas. Eksperimentinės sistemos sugeneruota duomenų bazės schema.

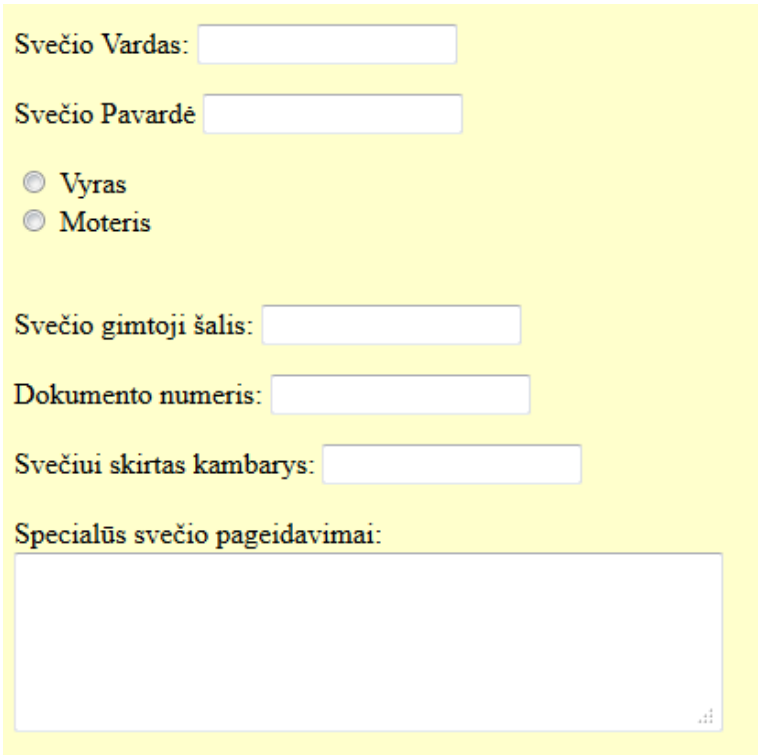
Duomenų bazės schema sukurta eksperto (6.9 pav.). Sukurta duomenų bazė susideda iš 6 lentelių: „address“, „order“, „city“, „cartype“, „driver“, „car“. Sukurti ryšiai tarp lentelių.



6.9 paveikslas. Eksperto suprojektuota duomenų bazės schema.

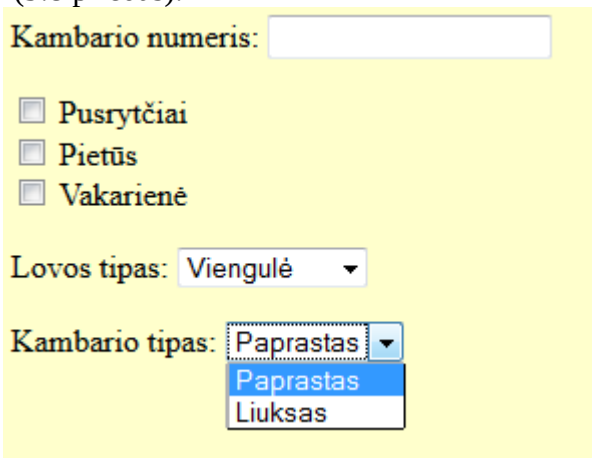
Esminis trečiojo tyrimo skirtumas tarp sugeneruotos duomenų bazės schemos ir eksperto sukurtos duomenų bazės schemos – ekspertinė sistema negali atpažinti ryšių tarp formų. Taip pat, jei naudojami elementai, tarp kurių irgi gali būti nustatytas ryšys. Programa negali jų nustatyti.

Ketvirtuoju bandymu panaudotos dvi formos, skirtinguose *HTML* dokumentuose. Pirmoji „svečio“ forma (6.10 pav.). Tai paprasta forma. Formos elementai nėra sudėtingi (9.8 priedas).



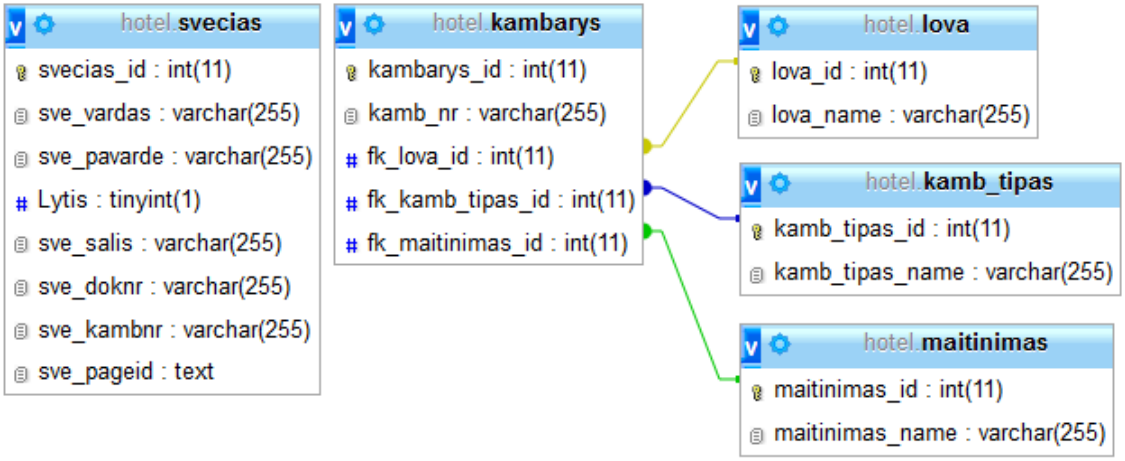
6.10 paveikslas.„Svečias“ formos pavyzdys

Antroji forma (6.11 pav.) sudėtingesnė forma. Joje panaudoti išsikleidžiančio sąrašo, pažymimojo lauko elementai (9.8 priedas).



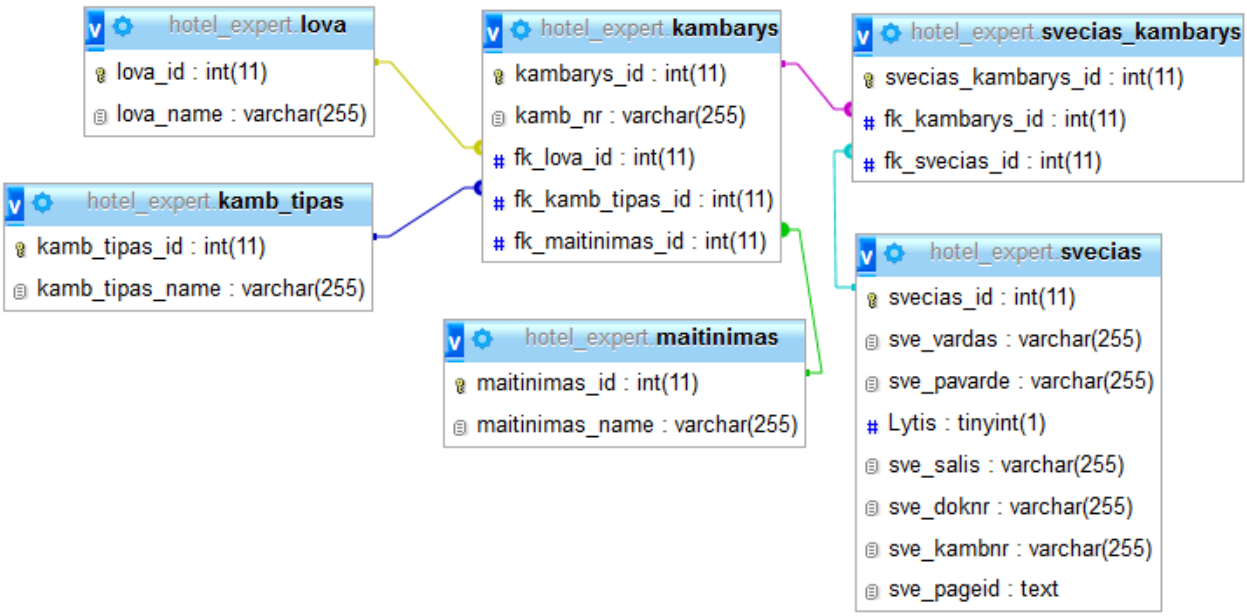
6.11 paveiklas.„Kambarys“ formos pavyzdys

Pasinaudojus eksperimentine programa buvo gauta paveiksle pavaizuoda schema.



6.12 paveikslas. Eksperimentinės sistemos suprojektuota duomenu bazės schema

Analogiškai kaip ir praeituose eksperimentuos, ekspertas suprojektavo savo duomenų bazės schemą (6.13 pav.).



6.13 paveikslas. Eksperto suprojektuota duomenų bazės schema.

Paskutiniu atveju sukurta ir sugeneruota duomenų bazės schemos skiriasi stipriai. Programa iš atskirų failų negali identifikuoti, ar dvi formos atskiruose failuose turi ryši tarpusavyje. Tai parodo programos ribas ir netobulumą. Iš tokių atskirų fromų sugeneruota duomenų bazė yra labai netiksli dėl to vartotojas turėtų įsikišti į šios duomenų bazės projektavimo darbus ir pasikoreguoti ją pagal turimą dalykinę sritį.

Programuotojai turi pakankamai daug laisvės programuodami *HTML* formas ir rasti identifikuojančius požymius, kad dviejų formų duomenų bazės schemos turi būti sujungtos ryšiais, yra gana sudėtinga.

6.2.1 Eksperimento apibendrinimas

Sukurta eksperimentinė programa nėra iki galo išbaigta, automatiškai sugeneruotos duomenų bazės – kartais gaunasi netikslios, jas reikia taisyti rankiniu būdu. 6.1 lentelėje yra aprašyti eksperimentų rezultatai: paduoti formų ir gauti duomenų bazių duomenys. Taip pat, apibendrinimui pateikiami ir eksperto sukurtų duomenų bazių duomenys.

6.1 *lentelė. Eksperimento apibendrinimo lentelė*

	Eksperimentas I	Eksperimentas II	Eksperimentas III	Eksperimentas IV
Formų skaičius	1	1	2	2
Formose esančių elementų skaicius	7	22	14	14
Sugeneruota lentelių	2	8	6	5
Sugeneruota stulpelių	8	26	24	19
Sugeneruota ryšių	1	7	4	3
Ekspertas sukurė lentelių	2	8	6	6
Ekspertas sukurė stulpelių	8	26	24	22
Ekspertas sukūrė ryšių	1	6	5	5

6.3 Eksperimentinės sistemos veikimo ir savybių analizė

Duomenų bazė generuota su generatoriumi ne visais atvejais gaunama tiksli. Tai priklauso nuo formų sudėtingumo ir jų skaičiaus, ryšio tarp formų. Dėl tos priežasties vartotojas sugeneravęs duomenų bazę turėtų ją peržiūrėti ir jei reikia ją keisti, priklausomai nuo dalykinės srities reikalavimų.

Iš ekspertimentinės sistemos nefunkcinių reikalavimų sistemos veikimo ir savybių analizei buvo išskirta šios veikimo sąvybės:

- Eksperimentinės sistemos veikimas yra nesudėtingas: valdymo meniu nėra sudėtingas, keliais paspaudimais vartotojui pateikiama schema.
- Sistema veikia su visom operacinėm sistemomis, kurios turi prieigą prie interneto.
- Sistema veikia greitai, nesudėtingas formas duomenų bazės schema sugeneruoja greičiau nei per 10 s.
- Eksperimentinė sistema *SQL* kodą generuoja populiariai naudojamomis duomenų bazių valdymo sistemomis: *MySQL* is *Microsoft SQL* server.

6.4 Eksperimentinės sistemos taikymo rekomendacijos

Sistema gali būti naudojama projektuojant įvairaus tipo WEB informacines sistemas. Programa nesugeneruoja pilnai paruoštos sistemos, vartotojas pagal savo ir keliamus reikalavimus koreguoja duomenų bazės schemą, jau perkėlus ją į atitinkamą naudojamą DBVS. Taip pat vartotojas turi padaryti formos surišimą su sugeneruota ir patalpinta duomenų bazės schema.

7 IŠVADOS

1. Išanalizavus esamas duomenų bazių valdymo sistemas buvo nustatyta, kad dabartinės DBVS turi vedlius, kurie iš pateiktos duomenų bazės generuoja ekraninę formą. Nuspręsta, kad reikia sukurti algoritmą ir juo paremtą programą, kuris iš pateiktos ekraninės formos generuoja duomenų bazės schemą.
2. Buvo sukurta eksperimentinė programa generuojanti duomenų bazes, iš pateiktų vaizdinių formų. Programa generuoja duomenų bazės schemą ir atiduoda ją SQL kodo formatu.
3. Eksperimentinei programai realizuoti buvo pasirinkta *PHP* programavimo kalba ir *HTML* ekraninės formos. *HTML* ekraninės formos buvo pasirinktos dėl populiarumo WEB informacinių sistemų projektuose.
4. Eksperimentinė sistema buvo įdiegta į serverį, bei buvo atlikti eksperimentai su pasirinktomis ekraninėmis formomis.
5. Atlikus eksperimentus su eksperimentine sistema buvo nustatyta:
   - Algoritmas generuoja duomenų bazę automatiškai, bet norint gauti tikslų rezultatą reikalingas nedidelis vartotojo įsikišimas sutvarkant duomenų bazės stulpelių tipus, ryšius.
   - Eksperimentinė sistema yra sukurta eksperimento tikslais, dėl to ją galima tobulinti, prijungti papildomus formos kūrimo ir vaizdavimo posistemius, kitų duomenų bazių *SQL* kodo generavimo pasirinkimus.
6. Ateityje, norint, kad eksperimentine sistema naudotųsi platesnis ratas vartotojų, reikia tobulinti, prijungti papildomus posistemius.

8 LITERATŪRA

[1] Benkt Wangler, „Contributions to Functional Requirements Modelling“ 1993 m. Edsbruk. pp.189-220

[2] Rita Butkiene, Daktaro disertacija „Informacijos sistemai keliamų funkcinių reikalavimų specifikavimo metodas“ 2002 m., Kaunas.

[3] Simon R. Rollinson ,Stuart A. Roberts, „Formalizing the Informational Content of Database User Interfaces“ 1998 m. Leeds.

[4] John C. Hancock, Roger Toren „Practical Business Intelligence with SQL Server 2005“ 2006m. Crawfordsville.

[5] Oracle Database Online Documentation 10g Release 2 (10.2). [Tinkle]. Available: http://www.oracle.com/pls/db102/portal.portal_db?selected=2 [Kreiptasi 2010-10-20]

[6] OpenBase SQL Overview. [Tinkle]. Available: http://openbase.wikidot.com/ [Kreiptasi 2010-10-20]

[7] Postgresql wiki Documentation. [Tinkle]. Available: http://wiki.postgresql.org/wiki/Main_Page [Kreiptasi 2010-10-20].

[8] Michael J. Prietula, Salvatore T. March „Form and Substance in Physical Database Design: An Empirical Study“ 1991 m.

[9] About OpenOffice.org, [Tinkle]. Available: http://about.openoffice.org/index.html [Kreiptasi 2010-10-20].

[10] MySQL AB „MySQL® Administrator's Guide and Language Reference“ 2006 m.

[11] Guide to MySQL for Microsoft Windows. [Tinkle]. Available: http://www.mysql.com/why-mysql/white-papers/mysql_wp_onwindows.php [Kreiptasi 2010-10-20].

[12] Microsoft Visual FoxPro 9.0 SP2. [Tinkle]. Available: http://msdn.microsoft.com/en-US/library/724fd5h9%28v=VS.80%29.aspx [Kreiptasi 2010-11-07]

[13] „XML support in Microsoft SQL Server 2005“ [Tinkle]. Available: http://www.imamu.edu.sa/DContent/IT_Topics/xml_support_in_sql.doc [Kreiptasi 2010-12-14]

[14] Andrew Watt, Microsoft SQL Server 2005 For Dummies. Wiley Publishing. 2006.

[15] Oracle Designer tutorial. Oracle Corporation, 2007 [Tinkle]. Available: http://www.oracle.com/technetwork/testcontent/designer-tutorial-129019.zip [kreiptasi 2010-12-14]

[16] Otey, Michael, „Visual Studio 2010“, 2010 m., SQL Server Magazine.

[17] Visual Studio MSDN. [Tinkle]. Available: http://msdn.microsoft.com/en-us/vstudio/default [Kreiptasi 2011-01-14]

[18] PHP: Documentation. [Tinkle]. Available: http://www.php.net/docs.php [Kreiptasi 2011-01-15]

[19] Jeremy Allen, Charles Hornberger PHP 4 vadovas, 2003 m. „Smaltijos“ leidykla , Kaunas

[20] World Wibe Web Consortium (W3C). [Tinkle]. Available: http://www.w3.org [Kreiptasi 2011-01-16]

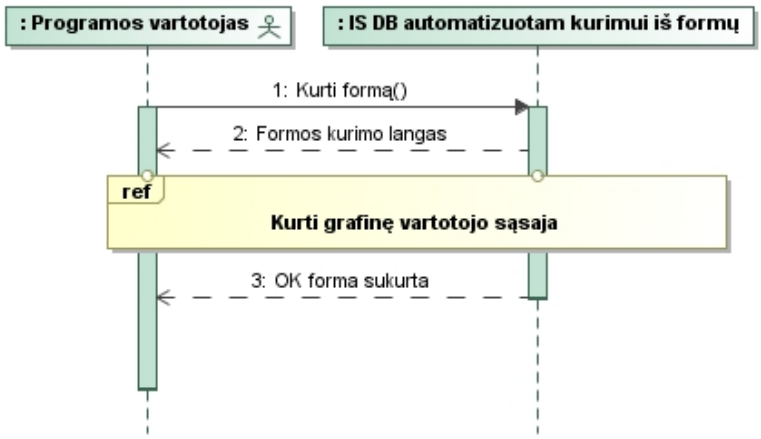
[21] Lietuvių kalba informacinėse technologijose. [Tinkle]. Available: http://www.litkit.lt/ [Kreiptasi 2011-01-17]

[22] HTML parser. [Tinkle]. Available: http://htmlparser.sourceforge.net [Kreiptasi: 2011-07-06]

[23] PHP: DOM – Manual. [Tinkle]. Available: http://php.net/manual/en/book.dom.php [Kreiptasi: 2012-10-06]

9 PRIEDAI

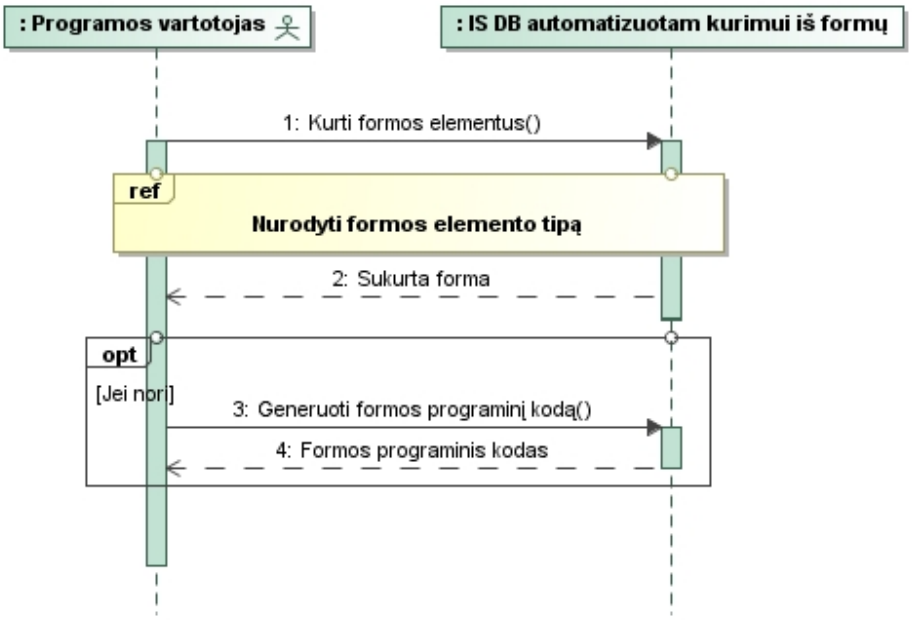
9.1 Priedas. Sekų diagramos ir jų specifikacijų lentelės



9.1 paveikslas. PA „Projektuoti grafinę vartotojo sąsają“ sekų diagrama

9.1 *lentelė. PA „Projektuoti grafinę vartotojo sąsają“ specifikacija*

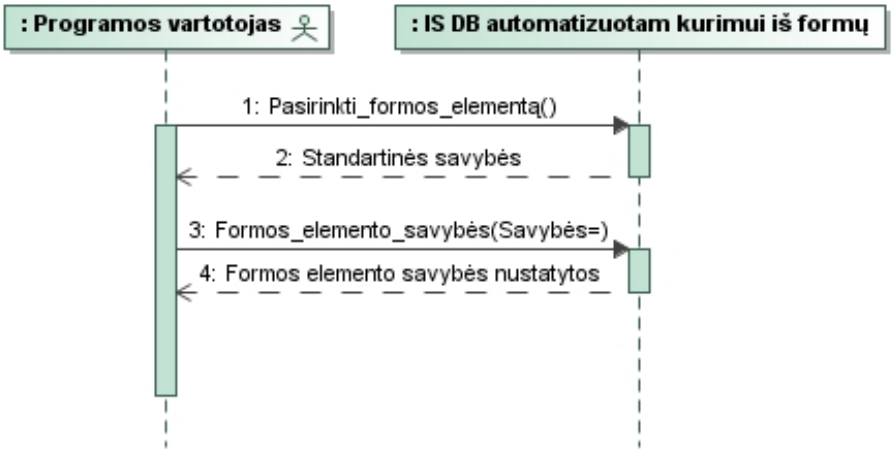
PA „Projektuoti grafinę vartotojo sąsają“		
Tikslas. Sukurti grafinę vartotojo sąsają kuriamai sistemai. Arba pateigti programai sukurtos grafinės vartotojo sąsajos programinį kodą suprogramuot trečių šalių programine įranga.		
Aprašymas.		
Prieš sąlyga		
Aktorius		Programos vartotojas
Sužadinimo sąlyga		Vartotojas nori sukurti grafinę vartotojo sąsają
Susiję panaudojimo atvejai	**Išplečia PA**	
	Apima PA	„Kurti grafinę vartotojo sąsają“
	Specializuoja PA	
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Vartotojos nori kurti formą.		Sistema atidaro formos kūrimo langą.
2. Vartotojos kuria formą.		Sistema įvykdo PA „Kurti grafinę vartotojo sąsają“. Ir pereina į 3 žingsnį.
3. Sistema baigia PA.		Sistema išsaugo atmintyje formą.
Po sąlyga:		
Alternatyvūs scenarijai		
Pateikiamas grafinės vartotojo sąsajos programini kodas suprogramuotas trečių šalių programine įranga.		



9.2 paveikslas. PA „Kurti grafinę vartotojo sąsaja“ sekų diagrama

9.2 *lentelė. PA „Kurti grafinę vartotojo sąsają“ specifikacija*

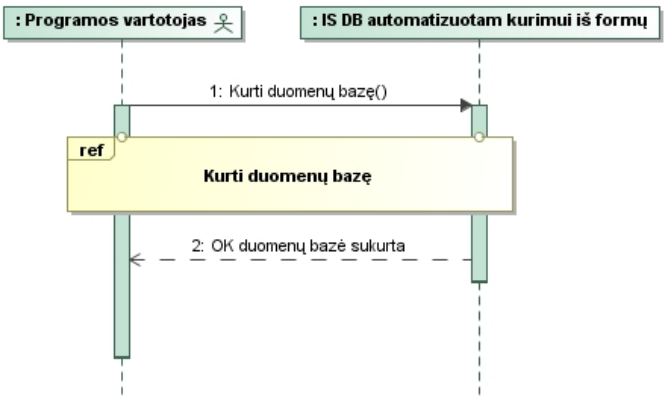
PA „Kurti grafinę vartotojo sąsają“		
Tikslas. Nurodyti formoje esančius elementus.		
Aprašymas. Šis PA yra Projektuoti grafinę vartotojo sąsaja PA dalis. Skirtas formos elementams nustatyti.		
Prieš sąlyga		
Aktorius		Programos vartotojas
Sužadinimo sąlyga		
Susiję panaudojimo atvejai	**Išplečia PA**	
	Apima PA	
	Specializuoja PA	„Projektuoti grafinę vartotojo sąsają“
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Vartotojos nori pridėti naują elementą.		Sistema į naujos formos langą įdedą elementą ir sukuria jai HTML kodo atitikmenį.
2. Vartotojos nori nustatyti apsirinkto lemento tipą.		Sistema įvykdo PA „Nurodyti formos elemento tipus“
3. Sistema baigia PA		Sistema išsaugo formą.
Po sąlyga:		
Alternatyvūs scenarijai		
Jei vartotojas nori sugeneruoti formos HTML kodą.		
1.Vartotojos nori sugeneruoti HTML kodą.		Sistema sugeneruoja HTML kodą sukurtai sistemai.



9.3 paveikslas. PA „Nurodyti formos elemento tipą“ sekų diagrama

9.3 *lentelė. PA „Nurodyti formos elemento tipą“ specifikacija*

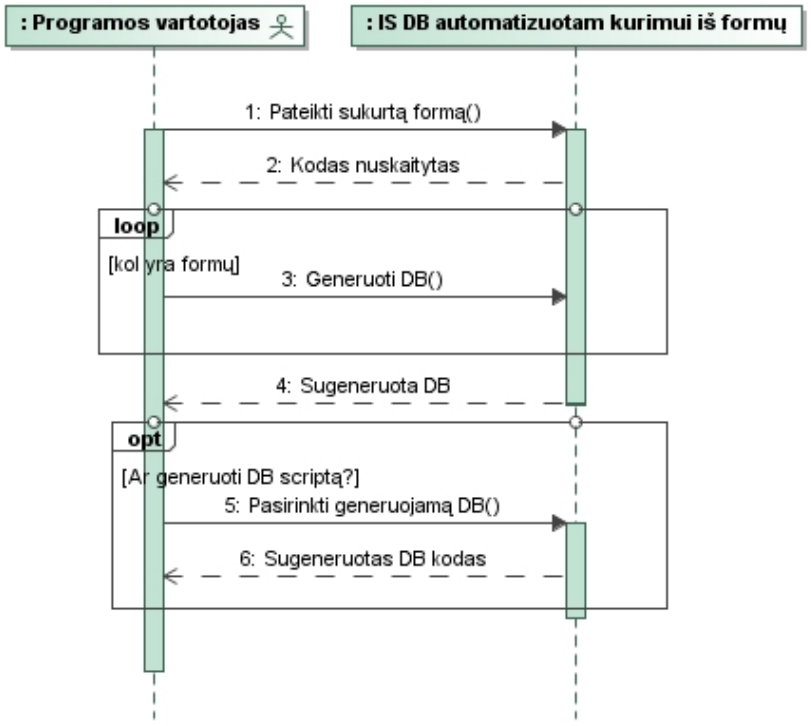
PA „Nurodyti formos elemento tipą“		
Tikslas. Nustatyti formos elemento tipą, jį pakeisti.		
Aprašymas. Šis PA yra Kurti grafinė vartotojo sąsają PA dalis. Skirtas formos elementams parametrizuoti. Nurodyti eilutės ilgi, įrašomų simbolių tipą, skaičių.		
Prieš sąlyga		
Aktorius		Programos vartotojas
Sužadinimo sąlyga		
Susiję panaudojimo atvejai	**Išplečia PA**	
	Apima PA	
	Specializuoja PA	„Kurti grafinė vartotojo sąsają“
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Vartotojas nori pakeisti elemento tipą.		Sistema pateikia standartines elemento savybes.
2. Vartotojas nurodo naują elemento tipą.		Sistema pakeičia pasirinkto elemento savybes. Ir išsaugo jas.
3. Sistema baigia PA.		
Po sąlyga:		
Alternatyvūs scenarijai		



9.4 paveikslas. PA „Projektuoti duomenų bazę“ sekų diagrama

9.4 *lentelė. PA „Projektuoti duomenų bazę“ specifikacija*

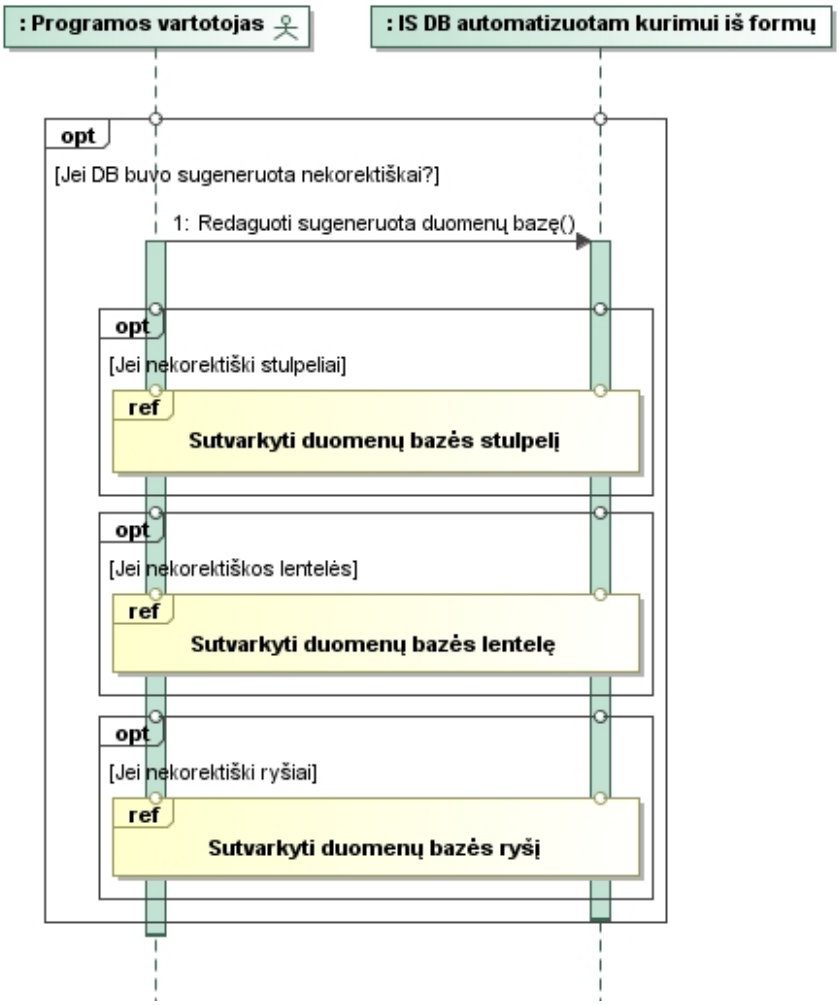
PA „Projektuoti duomenų bazę“		
Tikslas. Suprojektuoti ir iš formos sukurti duomenų bazę.		
Aprašymas. Šis PA skirtas kurti duomenų bazę iš pateiktos formos.		
Prieš sąlyga		
Aktorius		Programos vartotojas
Sužadinimo sąlyga		
Susiję panaudojimo atvejai	**Išplečia PA**	
	Apima PA	„Kurti duomenų bazę“,“ Redaguoti duomenų bazę“
	Specializuoja PA	
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Vartotojos nori generuoti duomenų bazę		Sistema įvykdo Kurti duomenų bazę PA ir pereina prie 2 žingsnio.
2. Sistema baigia PA.		Sistema išsaugo sugeneruota duomenų bazė ir baigia PA.
Po sąlyga:		
Alternatyvūs scenarijai		



9.5 paveikslas. PA „Kurti duomenų bazę“ sekų diagrama

9.5 *lentelė. PA „Kurti duomenų bazę“ specifikacija*

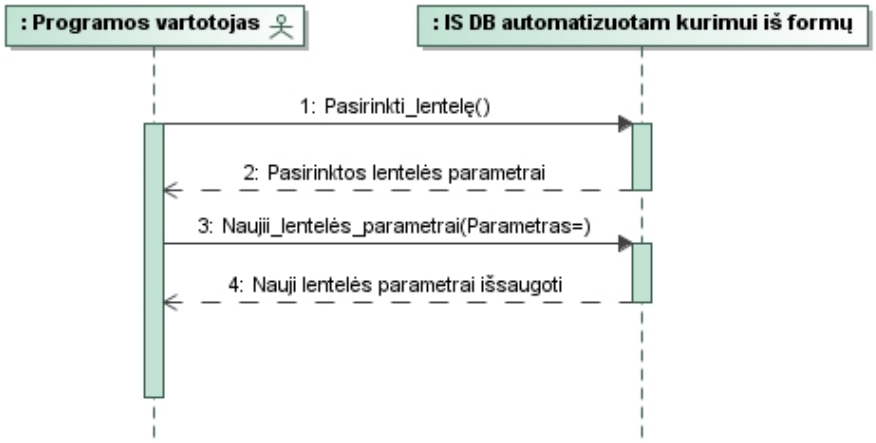
PA „Kurti duomenų bazę“		
Tikslas. Generuoti duomenų bazę iš sukurtos formos.		
Aprašymas. Iš sukurtos arba pateiktos trečiųjų asmenų suprogramuotos formos generuojama duomenų bazė pagal taisykles nurodytas generavimo algoritme. Duomenų bazė iškart susiejama su sukurta formą		
Prieš sąlyga		Turi būti sukurta arba pateikta forma.
Aktorius		Programos vartotojas
Sužadinimo sąlyga		
Susiję panaudojimo atvejai	**Išplečia PA**	
	Apima PA	
	Specializuoja PA	„Projektuoti duomenų bazę“
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Vartotojos pateikia duomenų sukurtą formą.		Sistema nuskaito pateiktą formą.
2. Vartotojos nori generuoti DB iš pateiktos formos.		Sistema generuoja formą. Kol yra pateiktų formų.
3. Sistema baigia PA		Sistema baigia PA
Po sąlyga:		
Alternatyvūs scenarijai		
Jei vartotojas nori sugeneruoti duomenų bazės DDL kodą.		
1. Vartotojas nori sugeneruoti DDL kodą.		Sistema sugeneruoja DDL kodą.



9.6 paveikslas. PA „Redaguoti duomenų bazę“ sekų diagrama

9.6 *lentelė. PA „Redaguoti duomenų bazę“ specifikacija*

PA „Redaguoti duomenų bazę“		
Tikslas. Sutvarkyti sugeneruota duomenų bazę.		
Aprašymas. Sugeneruotos duomenų bazės redagavimui skirtas PA. (Jei reikalinga redaguoti)		
Prieš sąlyga		1. Turi būti sukurta arba pateikta forma. 2. Sugeneruota duomenų bazė.
Aktorius		Programos vartotojas
Sužadinimo sąlyga		
Susiję panaudojimo atvejai	**Išplečia PA**	
	Apima PA	„Sutvarkyti duomenų bazės ryšį“, „Sutvarkyti duomenų bazės lentelę“, „Sutvarkyti duomenų bazės lauką“
	Specializuoja PA	
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Vartotojos nori redaguoti sukurtą duomenų bazę		Sistema pateikia sukurtos duomenų bazės schemą. Ir pereina prie alternatyvaus scenarijaus 1, 2, 3 žingsnių pasirinktinai.
2. Sistema baigia PA.		
Po sąlyga:		
Alternatyvūs scenarijai		
Sistemos vartotojas gali pasirinktinai pasirinkti alternatyvaus scenarijaus žingsnius priklausomai nuo ką nori redaguoti.		
1. Vartotojos nori redaguoti duomenų bazės stulpelį.		Sistema vykdo PA „Sutvarkyti duomenų bazės stulpelius“. Ir pereina prie pagrindinio scenarijaus 2 žingsnio.
2. Vartotojas nori redaguoti duomenų bazės lentelę		Sistema vykdo PA „Sutvarkyti duomenų bazės lenteles“. Ir pereina prie pagrindinio scenarijaus 2 žingsnio.
3. Vartotojas nori redaguoti duomenų bazės ryšį.		Sistema vykdo PA „Sutvarkyti duomenų bazės ryšius“ ir pereina prie pagrindinio scenarijaus 2 žingsnio.

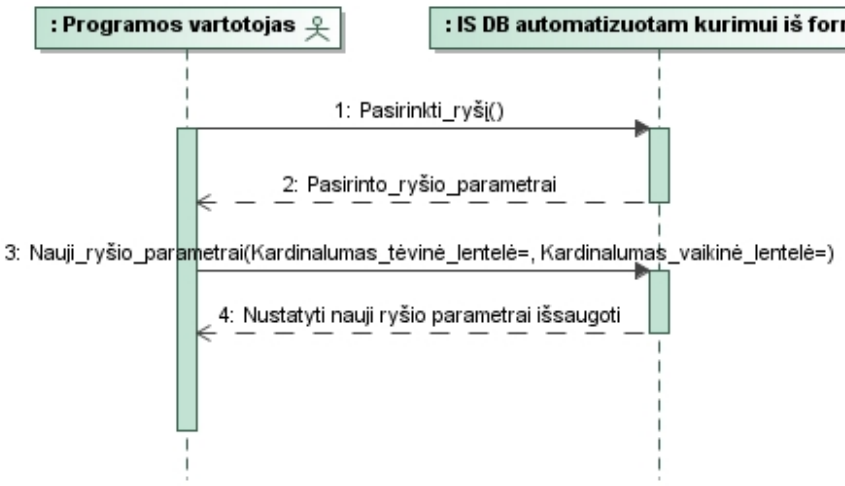


9.7 paveikslas. PA „Sutvarkyti duomenų bazės lentelę“ sekų diagrama

9.7 *lentelė. PA „Sutvarkyti duomenų bazės lentelę“ specifikacija*

PA „Sutvarkyti duomenų bazės lentelę“		
Tikslas. Sutvarkyti sugeneruota duomenų bazės lenteles.		
Aprašymas. Sugeneruotos duomenų bazės redagavimui skirtas PA. (Jei reikalinga redaguoti)		
Prieš sąlyga		1. Turi būti sukurta arba pateikta forma. 2. Sugeneruota duomenų bazė.
Aktorius		Programos vartotojas
Sužadinimo sąlyga		
Susiję	**Išplečia PA**	

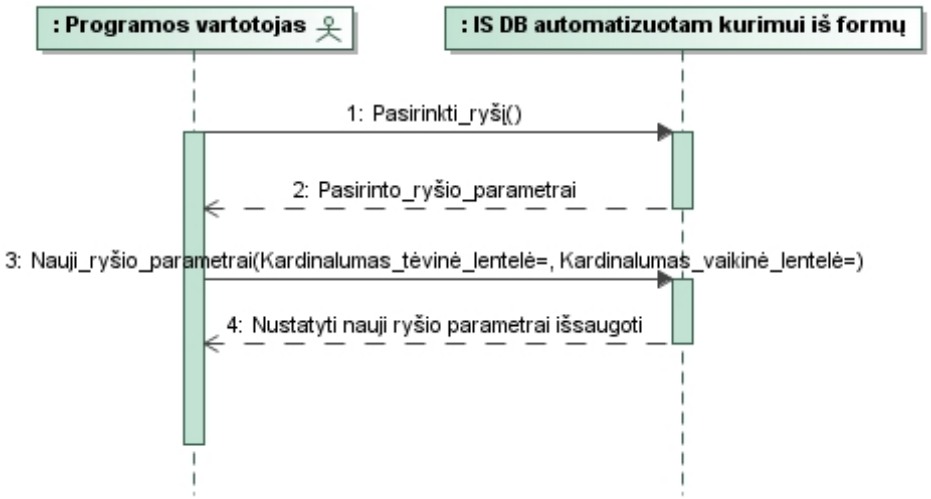
panaudojimo atvejai	**Apima PA**	
	Specializuoja PA	„Redaguoti duomenų bazę“
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Vartotojas pasirenka lentelę.		Sistema pateikia pasirinkta lentelę.
2. Vartotojas pateikia naujus lentelės parametrus.		Sistema išsaugo pateiktus lentelės parametrus.
3. Sistema baigia PA.		
Po sąlyga:		
Alternatyvūs scenarijai		



9.8 paveikslas. PA „Sutvarkyti duomenų bazės ryšius“ sekų diagrama

9.8 *lentelė. PA „Sutvarkyti duomenų bazės ryšius“ specifikacija*

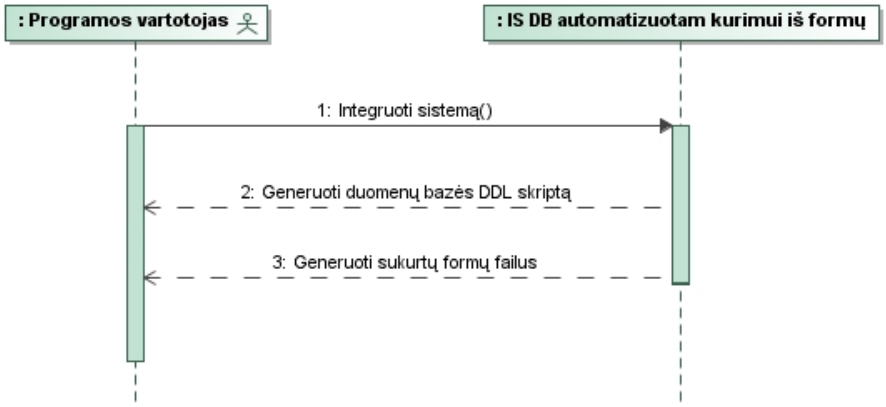
PA „Sutvarkyti duomenų bazės ryšį**“**		
Tikslas. Sutvarkyti sugeneruota duomenų bazės ryšius, jų tipus.		
Aprašymas. Sugeneruotos duomenų bazės redagavimui skirtas PA. (Jei reikalinga redaguoti)		
Prieš sąlyga		1. Turi būti sukurta arba pateikta forma. 2. Sugeneruota duomenų bazė.
Aktorius		Programos vartotojas
Sužadinimo sąlyga		
Susiję panaudojimo atvejai	**Išplečia PA**	
	Apima PA	
	Specializuoja PA	„Redaguoti duomenų bazę“
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Vartotojas pasirenka ryšį.		Sistema pateikia pasirinkta ryšį.
2. Vartotojas pateikia naujus lentelės parametrus.		Sistema išsaugo pateiktus ryšio parametrus.
3. Sistema baigia PA.		
Po sąlyga:		
Alternatyvūs scenarijai		



9.9 paveikslas. PA „Sutvarkyti duomenų bazės laukus“ sekų diagrama

9.9 *lentelė. PA „Sutvarkyti duomenų bazės laukus“ specifikacija*

PA „Sutvarkyti duomenų bazės laukus**“**		
Tikslas. Sutvarkyti sugeneruota duomenų laukus jų tipus bei parametrus.		
Aprašymas. Sugeneruotos duomenų bazės redagavimui skirtas PA. (Jei reikalinga redaguoti)		
Prieš sąlyga		1. Turi būti sukurta arba pateikta forma. 2. Sugeneruota duomenų bazė.
Aktorius		Programos vartotojas
Sužadinimo sąlyga		
Susiję panaudojimo atvejai	**Išplečia PA**	
	Apima PA	
	Specializuoja PA	„Redaguoti duomenų bazę“
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Vartotojas pasirenka lauką.		Sistema pateikia pasirinkta lauką.
2. Vartotojas pateikia naujus lauko parametrus.		Sistema išsaugo pateiktus lauko parametrus.
3. Sistema baigia PA.		
Po sąlyga:		
Alternatyvūs scenarijai		

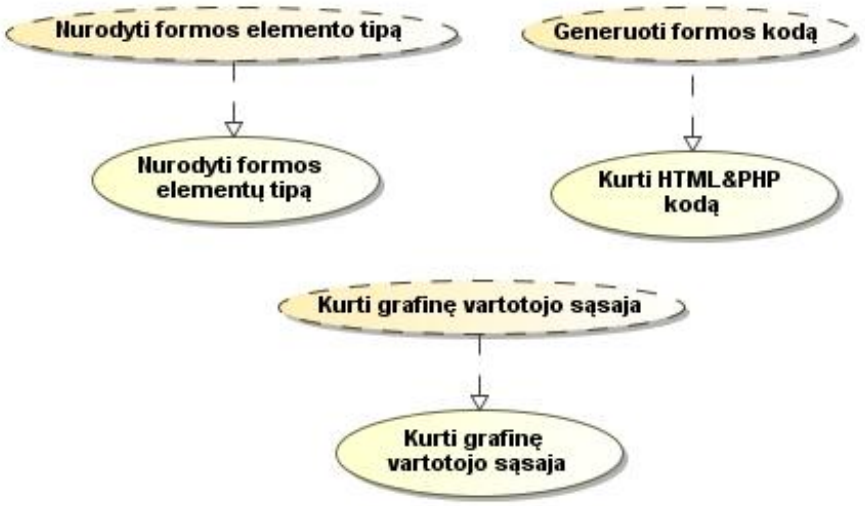


9.10 paveikslas. PA „Integruoti į vieną sistemą“ sekų diagrama

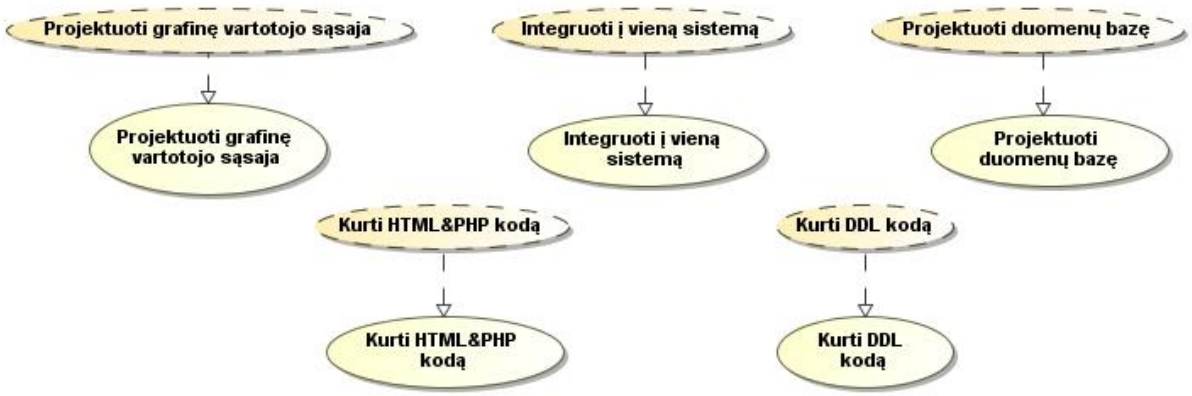
9.10 *lentelė. PA „Integruoti į vieną sistemą“ specifikacija*

PA „Integruoti į vieną sistemą**“**		
Tikslas. Pateikti sistemą vartotojui.		
Aprašymas. Sugeneruotos duomenų bazės ir sukurtos formos pateikimas galutiniam vartotojui.		
Prieš sąlyga		1. Turi būti sukurta arba pateikta forma. 2. Sugeneruota duomenų bazė.
Aktorius		Programos vartotojas
Sužadinimo sąlyga		
Susiję panaudojimo atvejai	**Išplečia PA**	
	Apima PA	
	Specializuoja PA	
Pagrindinis įvykių srautas		Sistemos reakcija ir sprendimai
1. Vartotojas nori gauti sistemos programinį kodą.		Sistema generuoja DDL, HTML ir PHP kodus.
2. Sistema baigia PA.		
Po sąlyga:		
Alternatyvūs scenarijai		

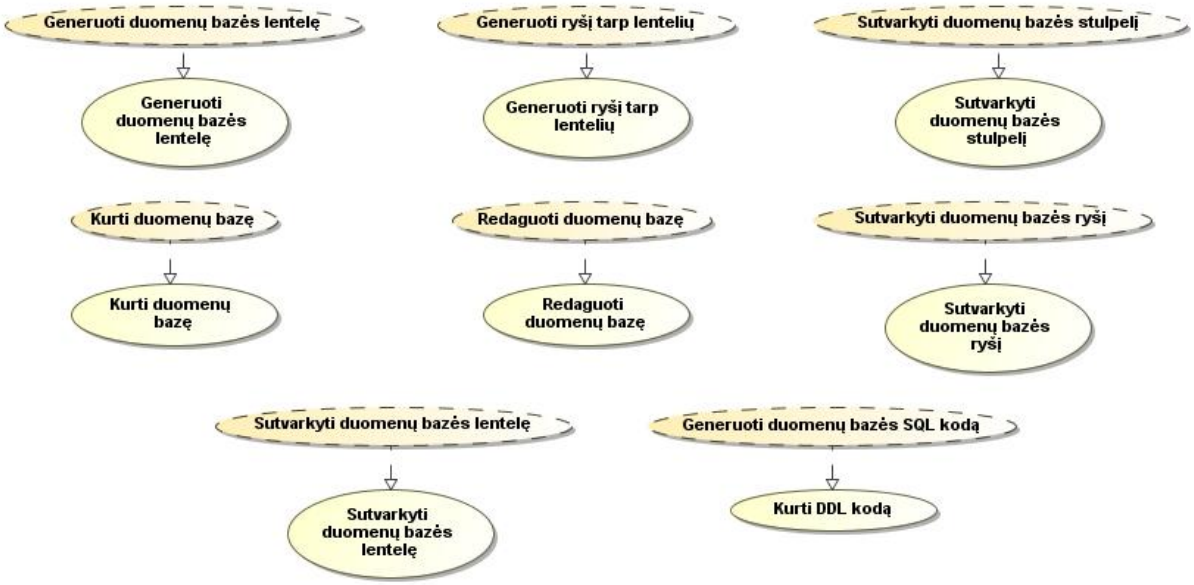
9.2 Priedas. Panaudojimo atvejų realizacijų daigramos



9.11 paveikslas. Formos kūrimo posistemio panaudojimo atvejų realizacijos



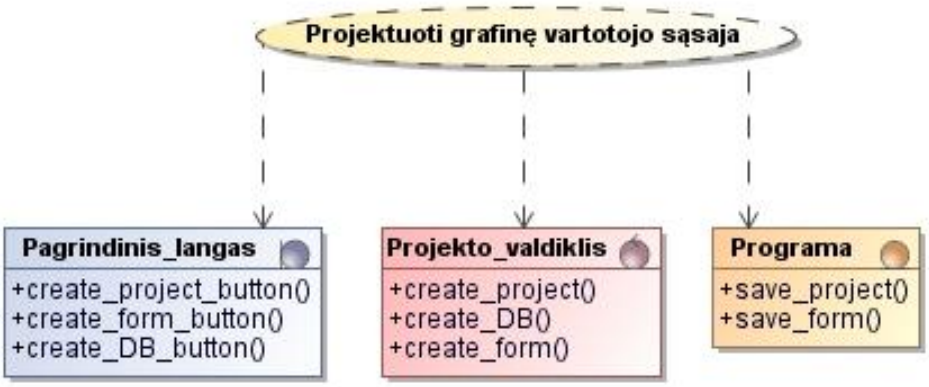
9.12 paveikslas. Pagrindinio posistemio panaudojimo atvejų realizacijos



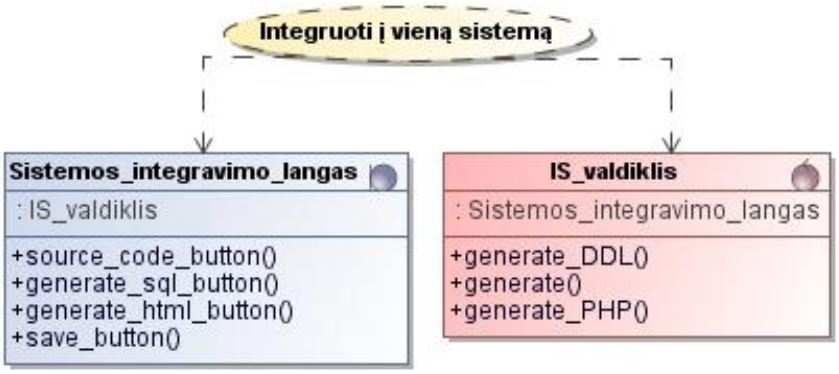
9.13 paveikslas. Duomenų bazės generavimo posistemio panaudojimo atvejų realizacijos

9.3 Priedas. Realizacijos klasių diagramos

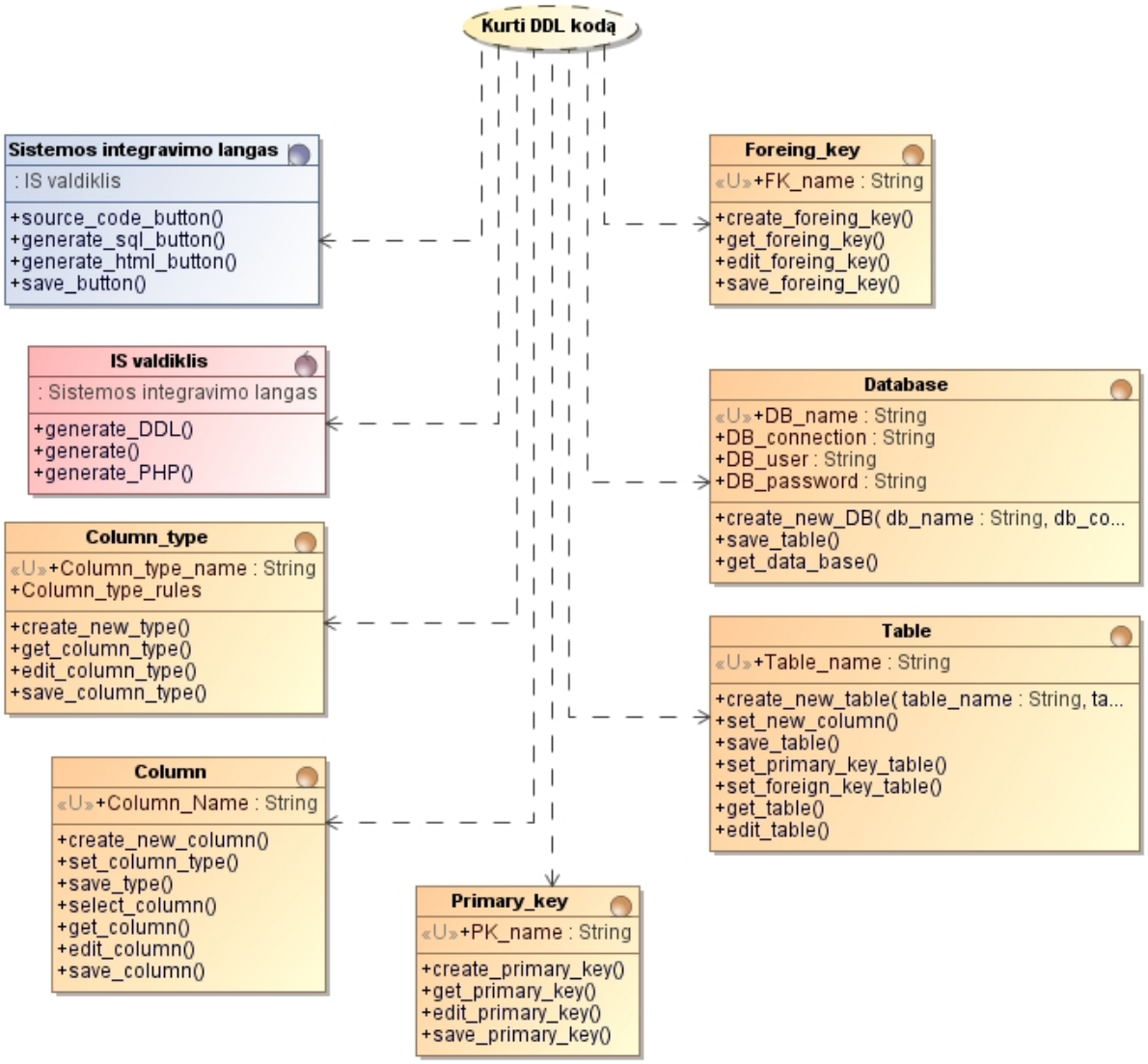
9.14 paveikslas. Panaudojimo atvejo „Projektuoti duomenų bazę“ realizacijos klasių diagrama



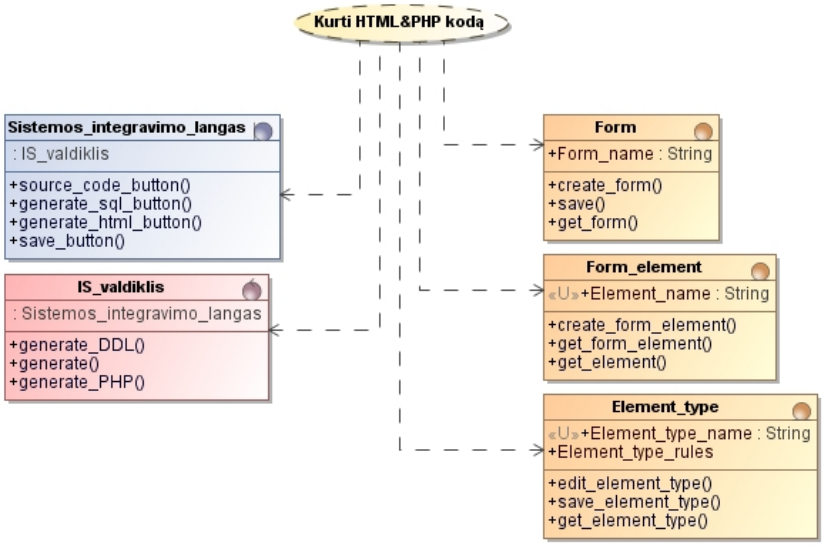
9.15 paveikslas. Panaudojimo atvejo „Projektuoti grafinę vartotojo sąsają“ realizacijos klasių diagrama



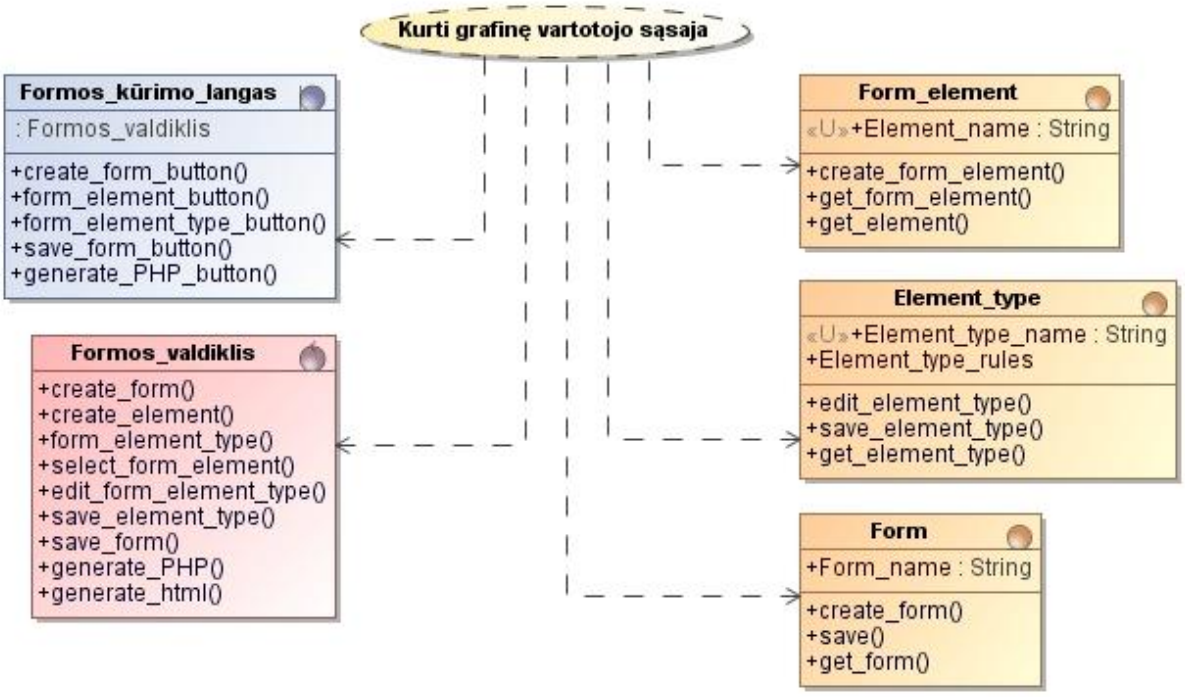
9.16 paveikslas. Panaudojimo atvejo „Integruoti į vieną sistemą“ realizacijos klasių diagrama



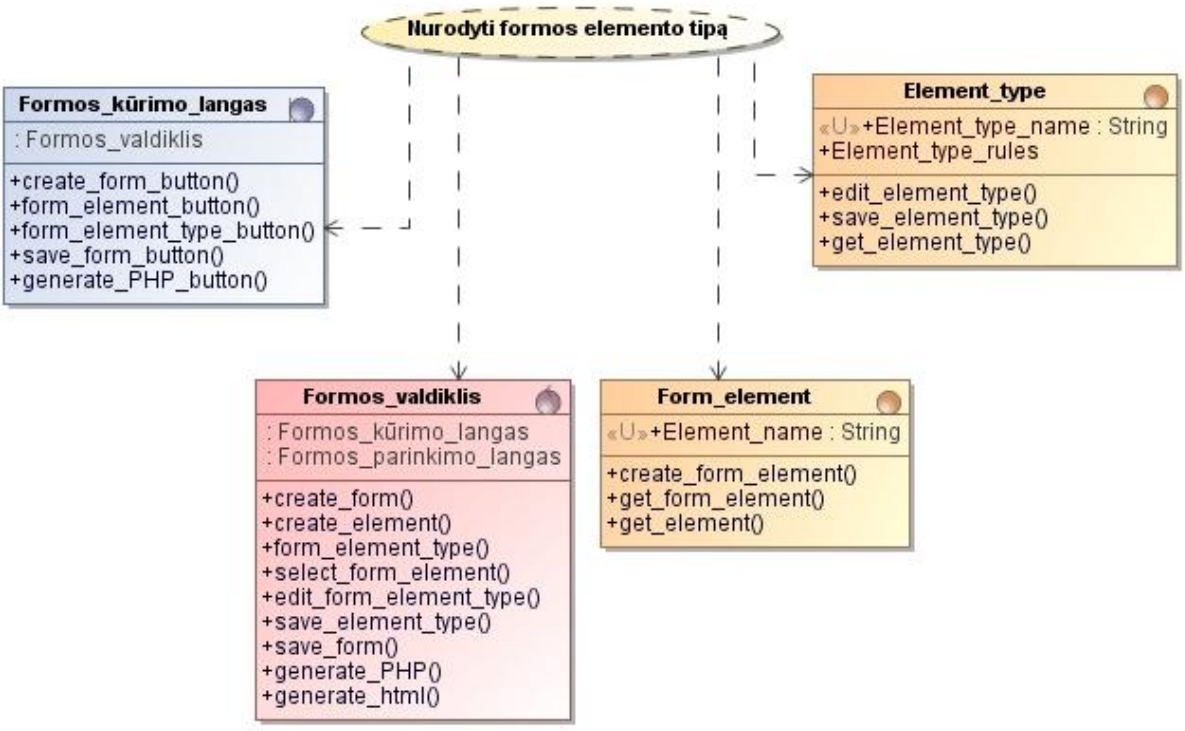
9.17 paveikslas. Panaudojimo atvejo „Kurti DDL kodą“ realizacijos klasių diagrama



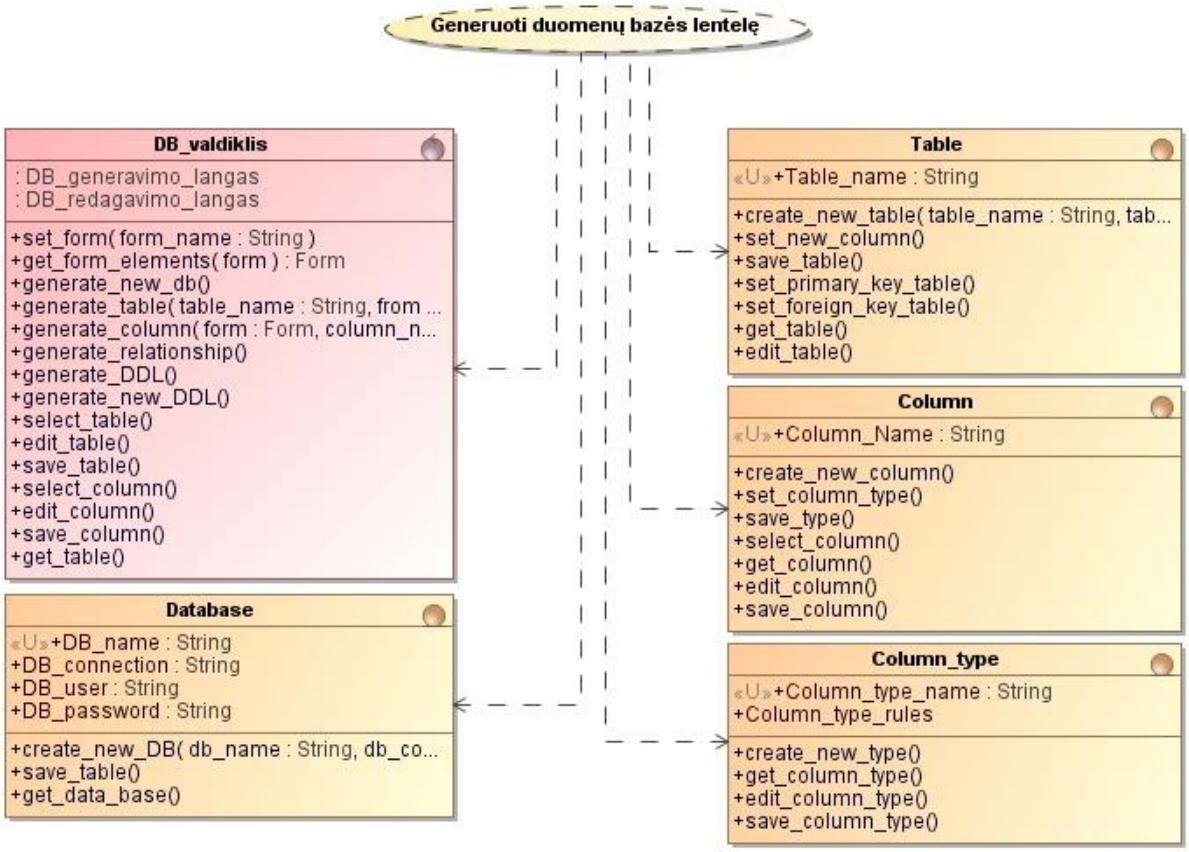
9.18 paveikslas. Panaudojimo atvejo „Kurti HTML&PHP kodą“ realizacijos klasių diagrama



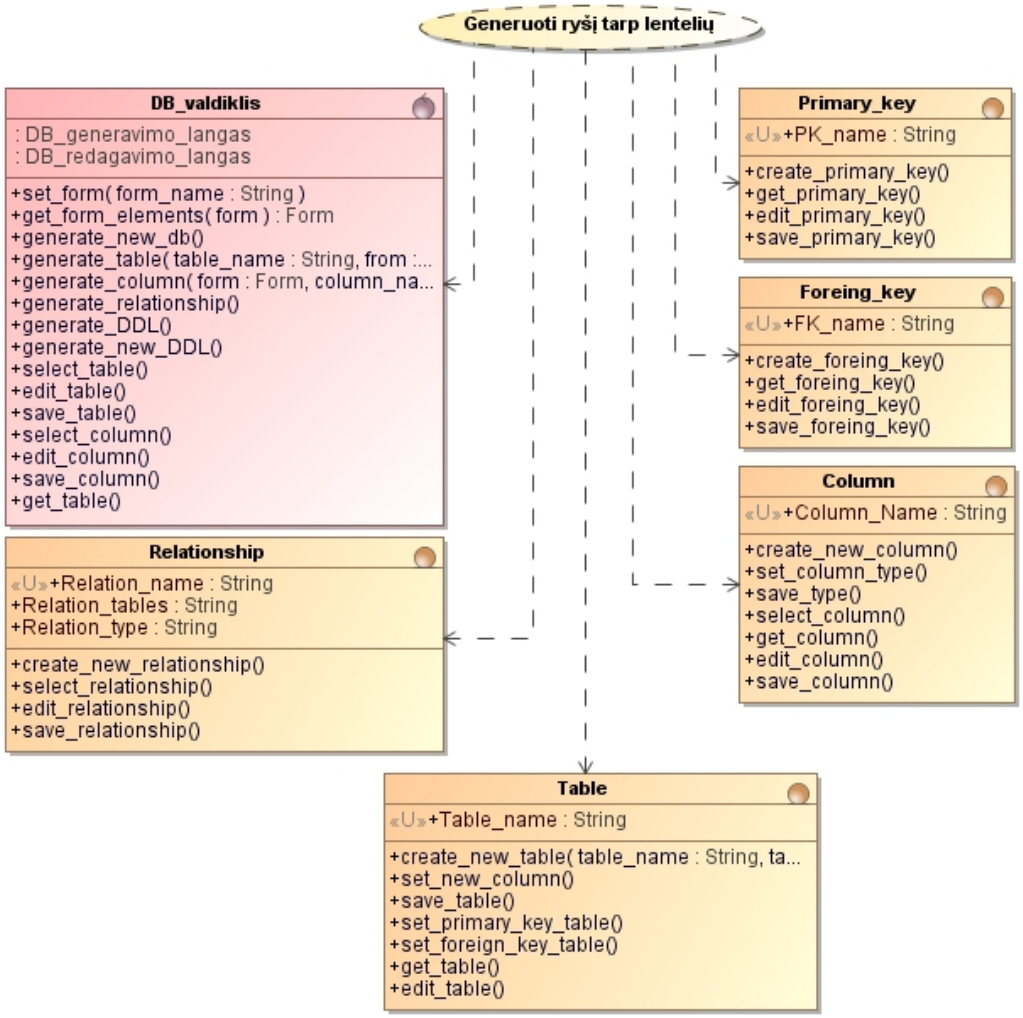
9.19 paveikslas. Panaudojimo atvejo „Kurti grafinę vartotojo sąsają“ realizacijos klasių diagrama



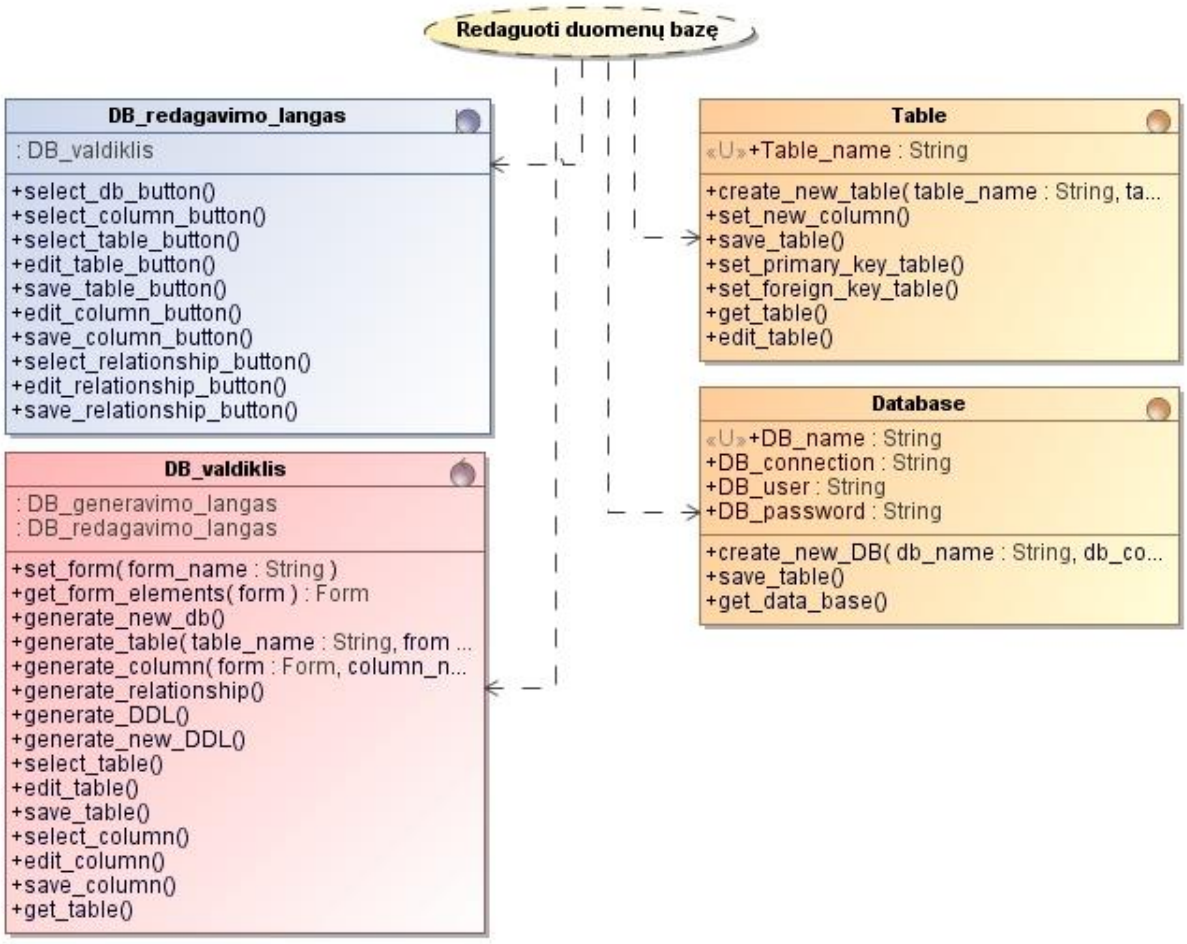
9.20 paveikslas. Panaudojimo atvejo „Nurodyti formos elemento tipą“ realizacijos klasių diagrama



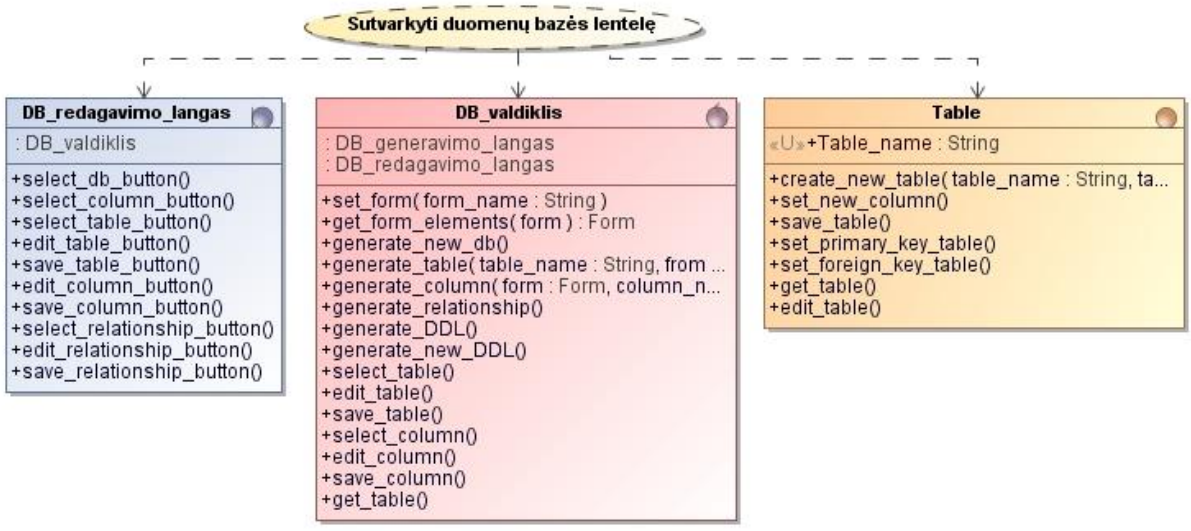
9.21 paveikslas. Panaudojimo atvejo „Generuoti duomenų bazės lentelę“ realizacijos diagrama



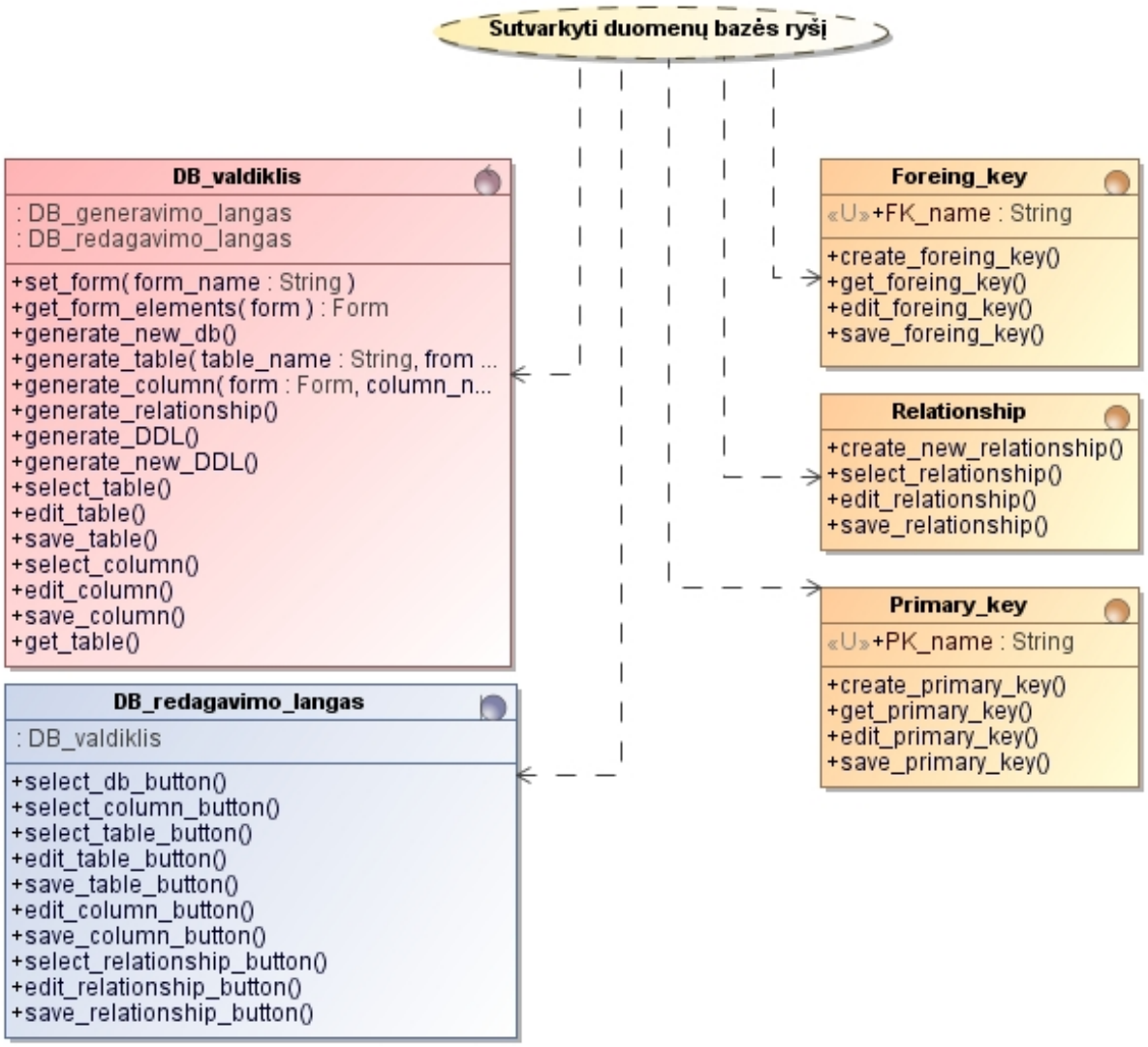
9.22 paveikslas. Panaudojimo atvejo „Generuoti ryšį tarp lentelių“ realizacijos diagrama



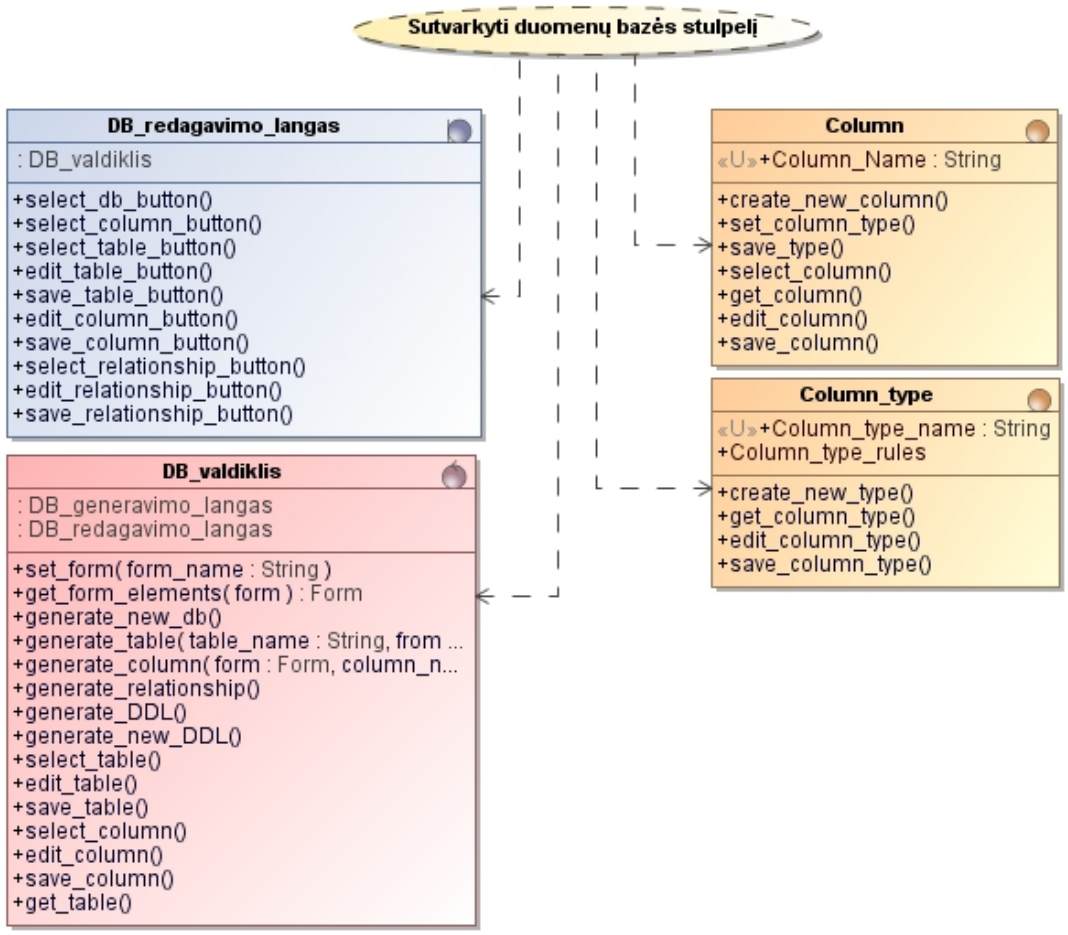
9.23 paveikslas. Panaudojimo atvejo „Redaguoti duomenų bazę“ realizacijos klasių diagrama



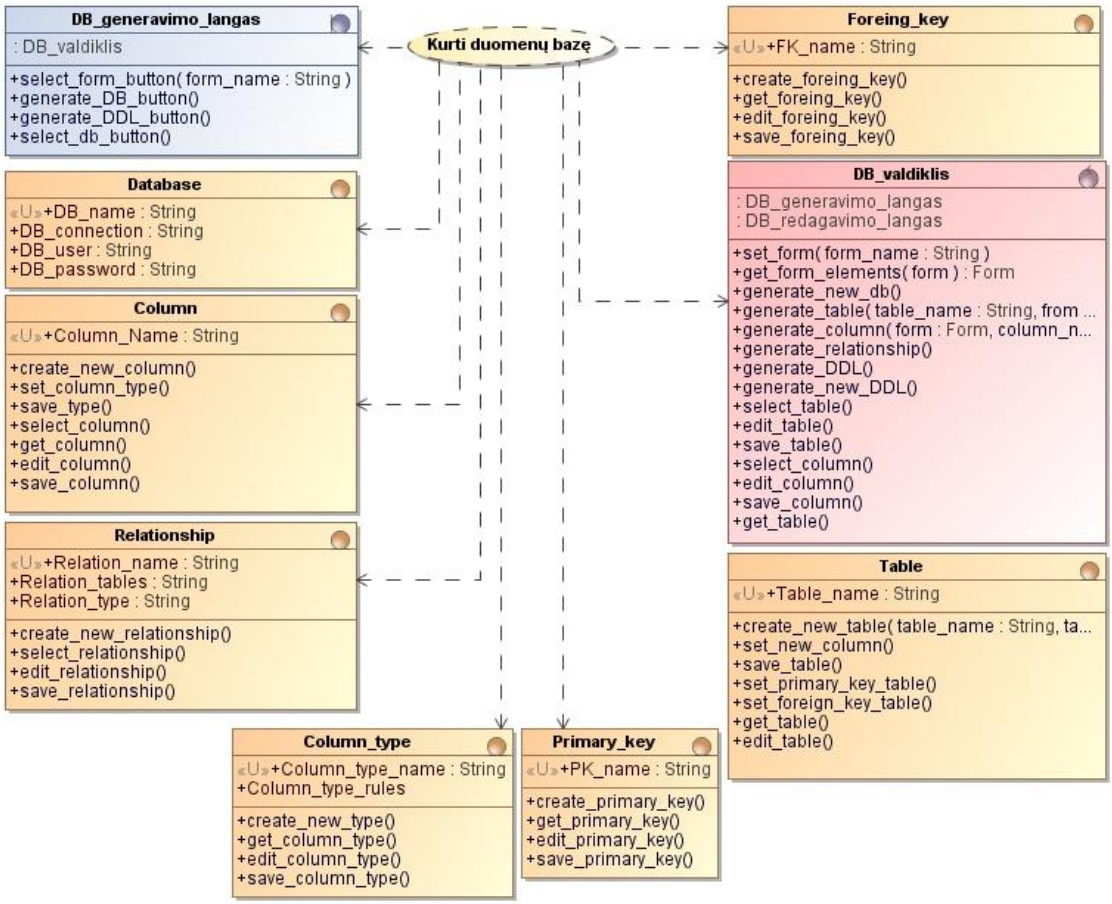
9.24 paveikslas. Panaudojimo atvejo „Sutvarkyti duomenų bazės lentelę“ realizacijos klasių diagrama



9.25 paveikslas. Panaudojimo atvejo „Sutvarkyti duomenų bazės ryšius“ realizacijos klasių diagrama

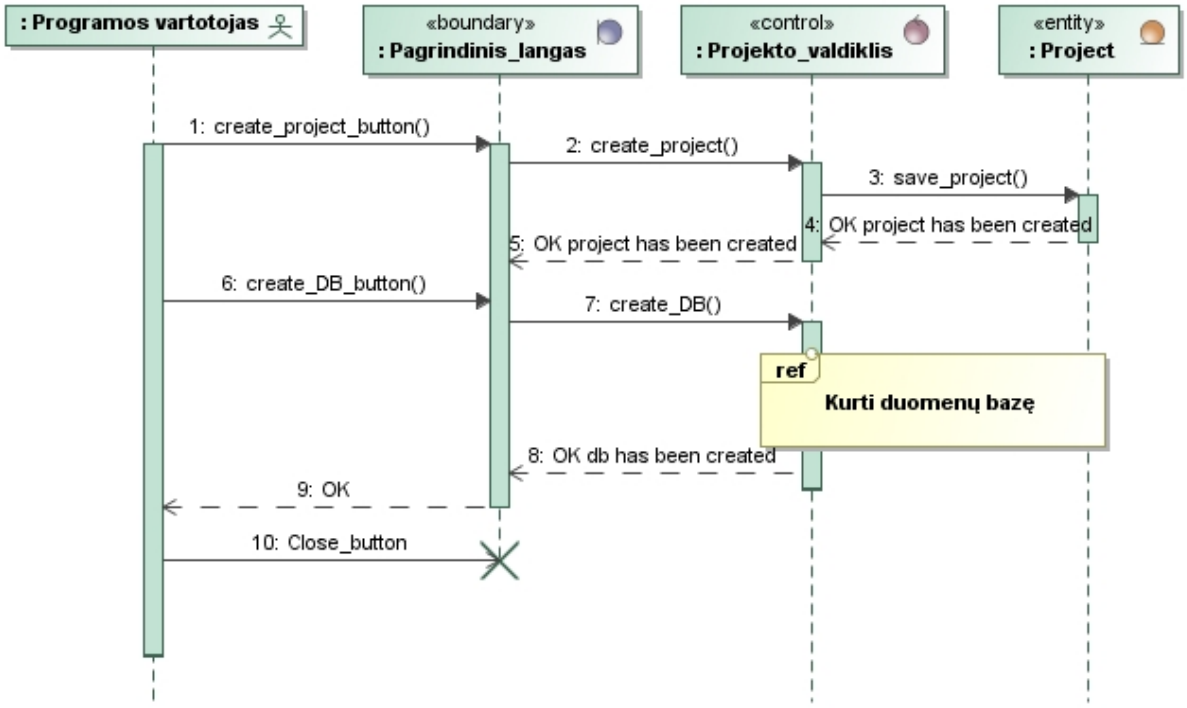


9.26 paveikslas. Panaudojimo atvejo „Sutvarkyti duomenų bazės stulpelį“ realizacijos klasių diagrama

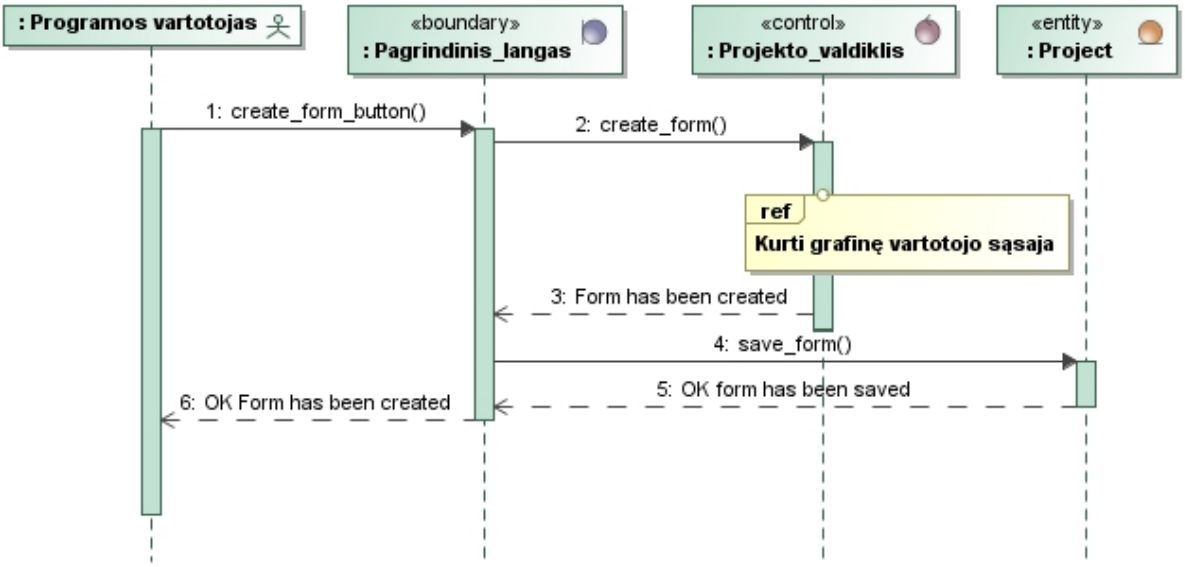


9.27 paveikslas. Panaudojimo atvejo „Kurti duomenų bazę“ realizacijos diagrama

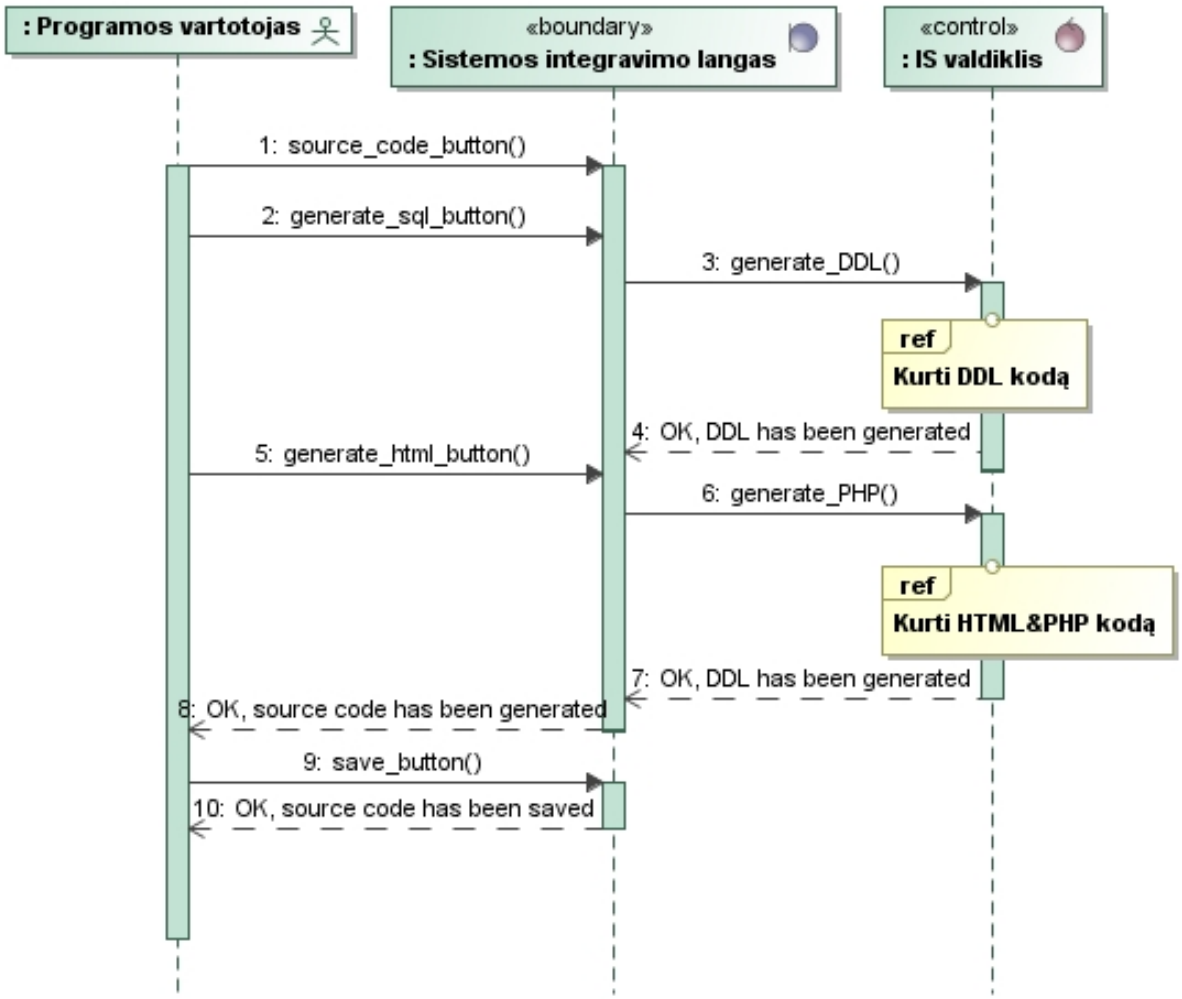
9.4 Priedas. Sekų diagramos



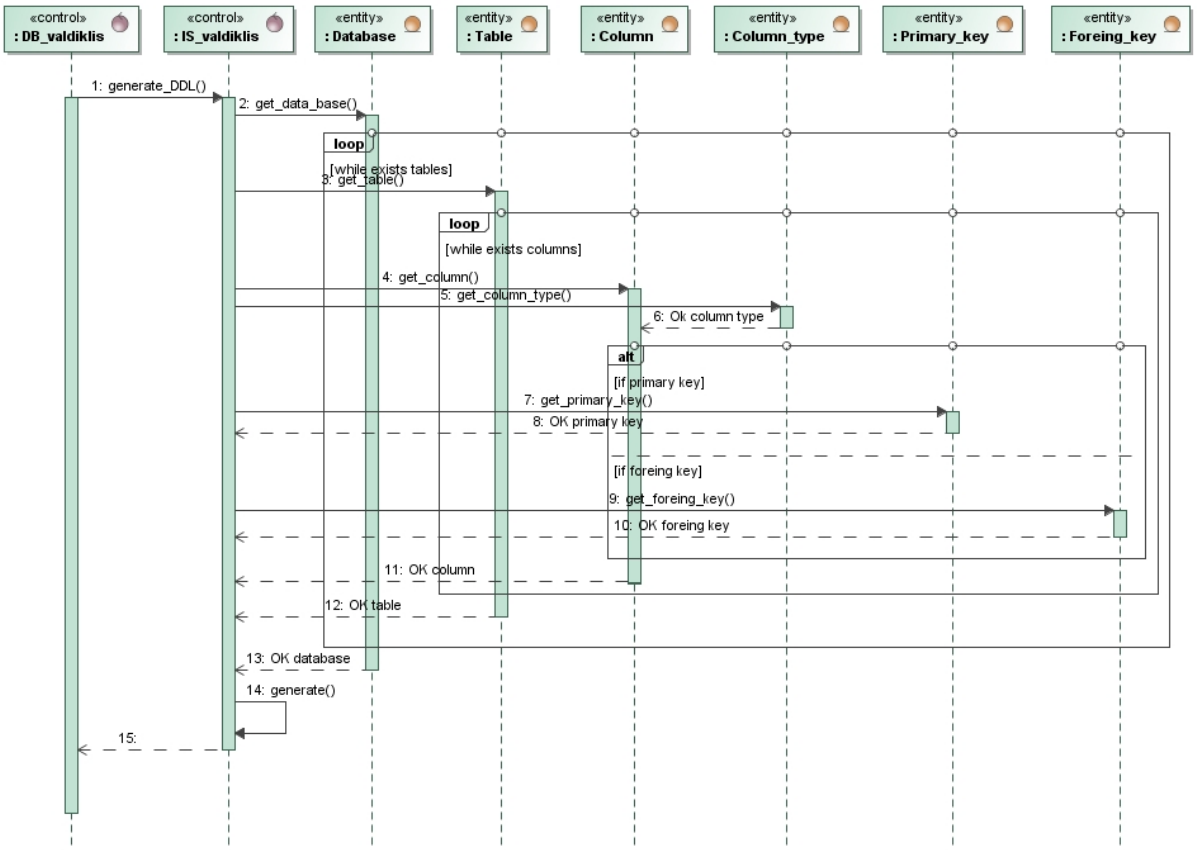
9.28 paveikslas. PA „Projektuoti duomenų bazę“ sekų diagrama



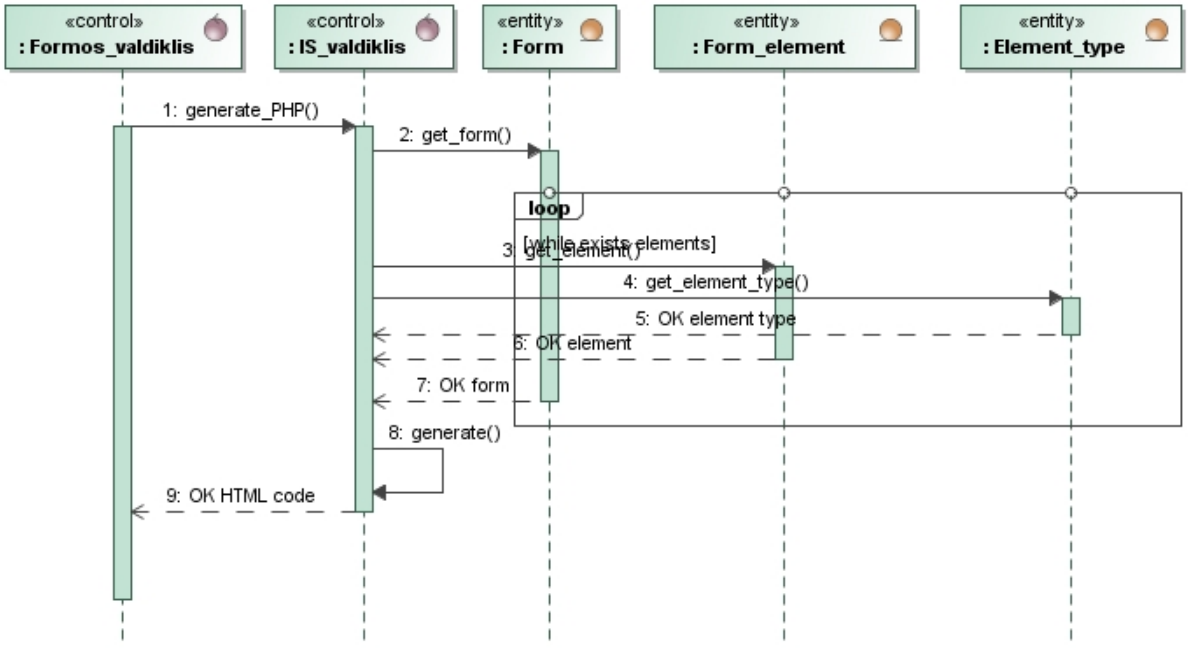
9.29 paveikslas. PA „Projektuoti grafinę vartotojo sąsaja“ sekų diagrama



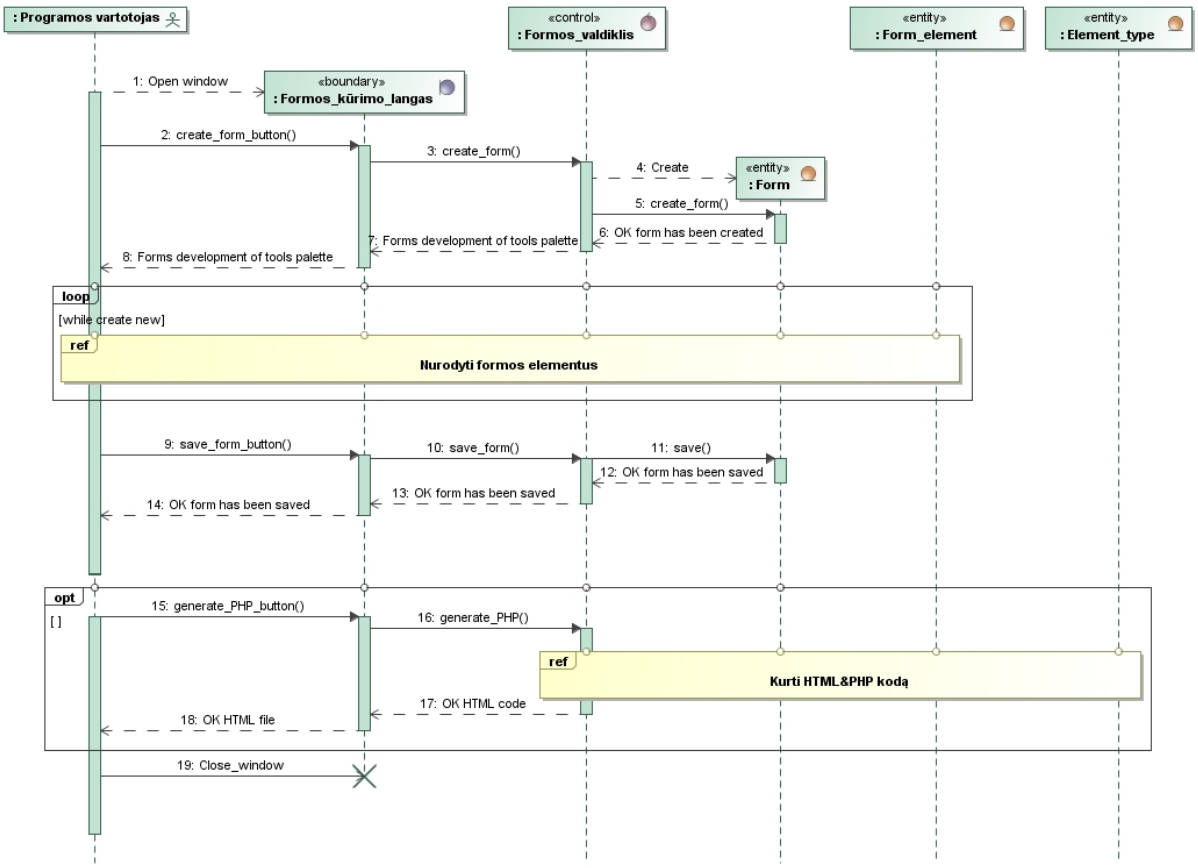
9.30 paveikslas. PA „Integruoti į vieną sistemą“ sekų diagrama



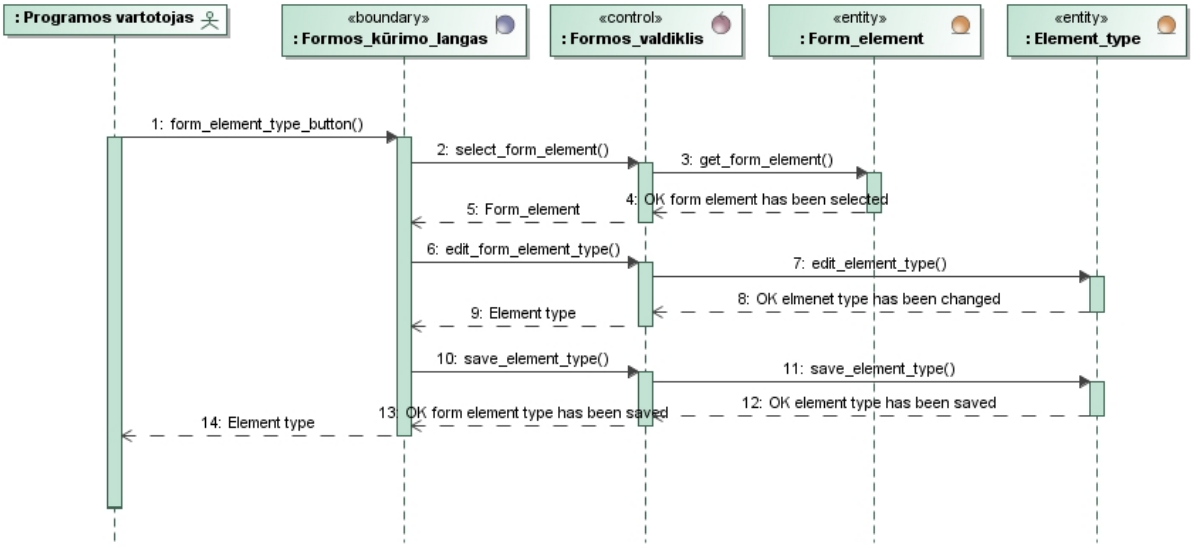
9.31 paveikslas. PA „Kurti DDL kodą“ sekų diagrama



9.32 paveikslas. PA „Kurti HTML&PHP kodą“ sekų diagrama



9.33 paveikslas. PA „Kurti grafinę vartotojo sąsają“ sekų diagrama



9.34 paveikslas. PA „Nustatyti formos elementų tipus“ sekų diagrama

9.5 Priedas. Paprasta registracijos forma

9.11 *lentelė. Paprastos formos programinis kodas*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Untitled Document</title>
<style type="text/css">
body {
       background-color: #CCC;
}
</style>
</head>

<body>
<form action="" method="post" enctype="multipart/form-data" name="form1" id="form1">
  <p>
    <label for="vardas">Vardas:</label>
    <input type="text" name="vardas" id="vardas" />
  </p>
  <p>
    <label for="pavarde">Pavardė:</label>
    <input type="text" name="pavarde" id="pavarde" />
  </p>
  <p>
    <label>
      <input name="tautybe" type="radio" id="r_lietuvis" value="lt" checked="checked" />
      Lietuvis/ė</label>
    <br />
    <label>
      <input type="radio" name="tautybe" value="uzs" id="r_uzsenietis" />
      Užsenietis</label>
  </p>
  <fieldset>
   <legend>Socialinė padėtis</legend>
    <label>
      <input name="statusas" type="radio" id="statusas_0" value="neved" checked="checked" />
      Nevedęs/Netekėjus</label>
    <br />
    <label>
      <input type="radio" name="statusas" value="ved" id="statusas_1" />
      Vedęs/Ištekėjus</label>
    <br />
    <label>
      <input type="radio" name="statusas" value="radio" id="statusas_2" />
      Našlys/Našlė</label>

  </fieldset>

</form>
</body>
</html>
```

9.6 Priedas. Forma „vairuotojų registracija“

9.12 *lentelė. Formos „vairuotojų registracija“ programinis kodas*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Vairuotoji registracija</title>
<style type="text/css">
body {
	background-color: #999;
}
</style>
</head>

<body>
Vairuotojo registracija:
<div style="width:450px">
<form id="vairuotojas_form" name="vairuotojas" method="post" action="">
	<p>
	<label>Vardas:   <input name="vardas" type="text" required="required"
/></label><br />
	<label>Pavardė:  <input name="pavarde" type="text" /></label>
    </p>
	<fieldset>
		<legend>Lytis:</legend>
			<label><input name="lytis" type="radio" value="vyras" />Vyras </label><br />
			<label><input name="lytis" type="radio" value="moteris" />Moteris</label>
		</fieldset>
	<fieldset id='stazas'>
		<legend>Stažas:</legend>
		<label><input type="checkbox" name="stazas" value="iki2metu" id="Stazas_0" />Iki 2
metų</label><br />
	<label><input type="checkbox" name="stazas" value="daugiau2metu" id="Stazas_1" />Daugiau negu
2 metus</label>
	</fieldset>
	<fieldset>
	<legend>Kategorijos:</legend>
		<label><input type="checkbox" name="kategorija" value="A1" id="Kategorija_0" />
A1</label>
		<label><input type="checkbox" name="kategorija" value="A" id="Kategorija_1" />A</label>
                                                <label><input type="checkbox"
name="kategorija" value="B1" id="Kategorija_2" />B1</label>
		<label><input type="checkbox" name="kategorija" value="B" id="Kategorija_3" />B</label>
		<label><input type="checkbox" name="kategorija" value="C1" id="Kategorija_4"
/>C1</label>
		<label><input type="checkbox" name="kategorija" value="C" id="Kategorija_5" />C</label>
		<label><input type="checkbox" name="kategorija" value="D1" id="Kategorija_6"
/>D1</label>
		<label><input type="checkbox" name="kategorija" value="D" id="Kategorija_7" />D</label>
		<label><input type="checkbox" name="kategorija" value="BE" id="Kategorija_8"
/>BE</label>
		<label><input type="checkbox" name="kategorija" value="C1E" id="Kategorija_9"
/>C1E</label>
		<label><input type="checkbox" name="kategorija" value="CE" id="Kategorija_10"
/>CE</label>
		<label><input type="checkbox" name="kategorija" value="D1E" id="Kategorija_11"
/>D1E</label>
		<label><input type="checkbox" name="kategorija" value="DE" id="Kategorija_12"
/>DE</label>
		</fieldset>
<p>
		<label>Vairuojamas automobilis:
		<select name="automobilis">
			<optgroup label="Japoniški automobiliai:">
				<option value="subaru">Subaru</option>
				<option value="honda">Honda</option>
				<option value="mitsubishi">Mitsubishi</option>
				<option value="mazda">Mazda</option>
				<option value="nissan">Nissan</option>
			</optgroup>
		<optgroup label="JAV automobiliai:">
			<option value="chrysler">Chrysler</option>
			<option value="ford">Ford</option>
			<option value="jeep">Jeep</option>
		</optgroup>
		<optgroup label="Itališki automobiliai:">
			<option value="alfaromeo">Alfa Romeo</option>
			<option value="ferrari">Ferrari</option>
```

```
                        <option value="fiat">Fiat</option>
                        <option value="lamborghini">Lamborghini</option>
                </optgroup>
                <optgroup label="Ispaniški automobiliai:">
                        <option value="seat">Seat</option>
                </optgroup>
                <optgroup label="Prancuziški automobiliai":>
                        <option value="peugeot">Peugeot</option>
                        <option value="renault">Renault</option>
                </optgroup>
                <optgroup label="Vokiški automobiliai:">
                        <option value="audi">Audi</option>
                        <option value="bmw">BMW</option>
                        <option value="mercedesbenz">Mercedes-Benz</option>
                        <option value="volkswagen">Volkswagen</option>
                </optgroup>
        </select></label>
</p>
<p>
        <label>Išnuomojamas automobilis:<br />
    <select name="nuom-auto" size="5" id="nuomauto">
                <option value="audia3">Audi A3</option>
                <option value="peugeot206">Peugeot 206</option>
                <option value="renaultmegane">Renault Megane</option>
                <option value="seatcordoba">SEAT CORDOBA</option>
                <option value="vwgolf">Volkswagen Golf</option>
        </select></label>
</p>
<p>
        <label>Trumpas aprašymas:<br /><textarea name="aprasymas" cols="50" rows="8"
></textarea></label>
</p>
    <input name="saugoti" type="button" value="saugoti" />
</form>
</div>
</body>
</html>
```

9.7 Priedas. Forma „vairuotojų registracija ir krovinio registracija“

9.13 *lentelė. Formos „vairuotojų registracija ir krovinio registracija“ programinis kodas*

```
         <!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Sudėtinga forma</title>
</head>

<body bgcolor="#CCCCCC">
<p>
Vairuotojas:
</p>
<form action="index.php" method="post" enctype="multipart/form-data" name="driver">
  <p>
    <label for="name">Vardas:</label>
    <input type="text" name="vardas" id="vardas">
  </p>
  <p>
    <label for="surname">Pavardė:</label>
    <input type="text" name="pavarde" id="pavarde">
  </p>
  <p>
    <label for="tabel_id">Tabelio numeris:</label>
    <input type="text" name="tabel_id" id="tabel_id">
  </p>
  <p>
    <label>
      <input name="cartype" type="radio" id="car_0" value="car" checked>
      Lengvoji</label>
    <br>
    <label>
      <input type="radio" name="cartype" value="bus" id="car_1">
      Mikroautobusas</label>
    <br>
    <label>
      <input type="radio" name="cartype" value="truck" id="car_2">
      Sunkvežimis</label>
```

```
  </p>
  <p>
  <label>Vairuojamas automobilis:
              <select name="car">
            <option value="subaru">Subaru</option>
            <option value="honda">Honda</option>
            <option value="mitsubishi">Mitsubishi</option>
            <option value="mazda">Mazda</option>
            <option value="nissan">Nissan</option>
                      <option value="chrysler">Chrysler</option>
                      <option value="ford">Ford</option>
                      <option value="jeep">Jeep</option>
                      <option value="alfaromeo">Alfa Romeo</option>
                      <option value="ferrari">Ferrari</option>
                      <option value="fiat">Fiat</option>
                      <option value="lamborghini">Lamborghini</option>
                      <option value="seat">Seat</option>
                      <option value="peugeot">Peugeot</option>
                      <option value="renault">Renault</option>
                      <option value="audi">Audi</option>
                      <option value="bmw">BMW</option>
                      <option value="mercedesbenz">Mercedes-Benz</option>
                      <option value="volkswagen">Volkswagen</option>
        </select></label>
  </p>

</form>
<p>
Krovinys:
</p>
<form action="index.php" method="post" enctype="multipart/form-data" name="uzsakymas">
  <p>
    <label for="order_nr">Užsakymo nr:</label>
    <input type="text" name="uzsakymo_nr" id="uzsakymo_nr">
  </p>
  <p>
    <label for="wight">Krovinio svoris:</label>
    <input type="text" name="svoris" id="svoris">
  kg.</p>
  <p>
    <label>
      <input type="checkbox" name="priority" value="skubus" id="priority_0">
      Skubus</label>
  </p>
  <fieldset>
    <legend>Adresas    </legend>

    <p>
      <label for="custumer">Gavėjas:</label>
      <input type="text" name="custumer" id="custumer">
    </p>
    <p>
      <label for="address">Adresas:</label>
      <input type="text" name="adresas" id="adresas">
    </p>
    <p>
      <label for="city">Miestas:</label>
      <select name="city" id="city">
        <option value="alytus">Alytus</option>
        <option value="kaunas">Kaunas</option>
        <option value="klaipeda">Klaipėda</option>
        <option value="panevezys">Panevėžys</option>
        <option value="siauliai">Šiauliai</option>
        <option value="vilnius">Vilnius</option>
      </select>
      <label for="postcode">Pašto kodas:</label>
      <input type="text" name="postcode" id="postcode">
    </p>
</fieldset>
</form>
<p> </p>
</body>
</html>
```

9.8 Priedas. Forma „Svečias“ ir forma „Kambarys“

9.14 *lentelė. Formos „Sveičias“ programinis kodas*

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Svečio forma</title>
</head>

<body bgcolor="#FFFFCC">
<form action="index.html" method="post" enctype="multipart/form-data" name="svecias">
  <p>
    <label for="sve_vardas">Svečio Vardas:</label>
    <input type="text" name="sve_vardas" id="sve_vardas">

  </p>
  <p>
    <label for="sve_pavarde">Svečio Pavardė</label>
    <input type="text" name="sve_pavarde" id="sve_pavarde">
  </p>
  <p>
    <label>
      <input type="radio" name="Lytis" value="vyras" id="sve_l_vyras">
      Vyras</label>
    <br>
    <label>
      <input type="radio" name="Lytis" value="moteris" id="sve_l_mot">
      Moteris</label>
  </p>
  <p><br>
    <label for="sve_salis">Svečio gimtoji šalis:</label>
    <input type="text" name="sve_salis" id="sve_salis">
  </p>
  <p>
    <label for="sve_doknr">Dokumento numeris:</label>
    <input type="text" name="sve_doknr" id="sve_doknr">
  </p>
  <p>
    <label for="sve_kambnr">Svečiui skirtas kambarys:</label>
    <input type="text" name="sve_kambnr" id="sve_kambnr">
  </p>
  <p>
    <label for="sve_pageid">Specialūs svečio pageidavimai:</label><br />
    <textarea name="sve_pageid" id="sve_pageid" cols="45" rows="5"></textarea>
  </p>
</form>
</body>
</html>
```

9.15 *lentelė. Formos „Kambarys“ programinis kodas*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Untitled Document</title>
</head>

<body bgcolor="#FFFFCC">
<form action="svecias" method="post" enctype="multipart/form-data" name="kambarys">
  <p>
    <label for="kamb_nr">Kambario numeris:</label>
    <input type="text" name="kamb_nr" id="kamb_nr" />
  </p>
  <p>
    <label>
      <input type="checkbox" name="Maitinimas" value="psrt" id="Maitinimas_0" />
      Pusrytčiai</label>
    <br />
    <label>
      <input type="checkbox" name="Maitinimas" value="piet" id="Maitinimas_1" />
      Pietūs</label>
    <br />
    <label>
      <input type="checkbox" name="Maitinimas" value="vakr" id="Maitinimas_2" />
      Vakarienė</label>
  </p>
  <p>
    <label for="lova">Lovos tipas:</label>
    <select name="lova" id="lova">
      <option value="single">Viengulė</option>
      <option value="double">Dvigulė</option>
      <option value="2single">2 viengulės</option>
    </select>
  </p>
  <p>
    <label for="kamb_tipas">Kambario tipas:</label>
    <select name="kamb_tipas" id="kamb_tipas">
      <option value="simple">Paprastas</option>
      <option value="liux">Liuksas</option>
    </select>
    <br />
  </p>
</form>
</body>
</html>
```