KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
KOMPIUTERIŲ KATEDRA

Andrius Galinauskis

Verslo programų saugos architektūros sukūrimas ir tyrimas

Magistro darbas

Darbo vadovas
doc. dr. Nerijus Morkevičius

Kaunas, 2012

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
KOMPIUTERIŲ KATEDRA

Andrius Galinauskis

Verslo programų saugos architektūros sukūrimas ir tyrimas

Magistro darbas

Recenzentas

doc. dr. Tomas Adomkus
2012-05-28

Vadovas

doc. dr. Nerijus Morkevičius
2012-05-28

Atliko

IFN-0/3 gr. stud.
Andrius Galinauskis

2012-05-28

Kaunas, 2012

TURINYS

ĮVADAS 6
1. PRIEIGOS TEISIŲ VALDYMO METODŲ IR PRIEMONIŲ ANALIZĖ 8
1.1 Verslo valdymo programos apžvalga 8
1.2 Saugos politikos formavimo apžvalga 11
1.3 Prieigos valdymo apžvalga 12
1.4 Prieigos saugos modelių analizė 13
1.4.1 DAC prieigos valdymo modelis 13
1.4.2 MAC prieigos valdymo modelis 14
1.4. 3 RBAC prieigos saugos modelis 17
1.4.4 Apibendrinimas 22
1.5 Kliento serverio architektūros analizė 23
1.5.1 Apibendrinimas 26
1.6 Verslo programos saugos architektūros analizė 27
1.7 „Firebird“ duomenų bazės apžvalga 30
2. REIKALAVIMŲ SPECIFIKACIJA 32
2.1 Funkciniai reikalavimai 32
2.1 Nefunkciniai reikalavimai 32
3. APIBENDRINTO VERSLO VALDYMO PROGRAMOS SAUGOS METODAS 34
3.1 Metodo panaudojimo atvejų modelis 34
3.2 Metodo duomenų bazės struktūra 35
3.3 Rolių ir leidimų valdymas jungiantis per SQL serverį 36
3.4 Prieigos valdymo metodo įgyvendinimas 37
3.4.1 Leidimų veikimo principas (admistratorius) 37
3.4.2 Laukų surinkimo formoje principas 39
3.4.3 Leidimų veikimo principas (vartotojas) 39
3.4.4 Teisių valdymas programos vartotojams 40
3.4.5 Duomenų nuskaitymas iš duomenų bazės 41
3.4.6 Duomenų išsaugojimas į duomenų bazę 41
3.4.7 Naujų duomenų pridėjimas į duomenų bazę 41
3.4.8 Auditas 42
4. EKSPERIMENTINĖ REALIZACIJA 43
4.1 Administravimas 43
4.2 Klientų langas 44
4.3 Užsakymo langas 45
4.4 Prekių langas 45
4.5 Audito langas 46
5. EKSPERIMENTO REZULTATAI 47
5.1 Eksperimento metu gauti rezultatai 47
5.2 Eksperimento išvados 50
6. IŠVADOS 51
LITERATŪRA 52
TERMINŲ IR SANTRUMPŲ ŽODYNAS 54

Summary

Enterprise Applications Security Framework Development and Analysis
Written by Andrius Galinauskis

This thesis investigates specific methods of access control of enterprise applications, for small and medium-sized business management in information systems. Most security problems are caused by staff rather than those outside the organization, so this must be addressed first. Enterprise Resource Planning (ERP) system is a multi-user, multi-role software solution. This work discusses the need for role based access control to be an integral part of an enterprise's user management facilities and to be easily managed too.

The main objective of the research is to review and analyze the problems of business access control and their solutions. This work consists of created a method based on results, which will allow administrators of ERP to manage resources of information smoothly and to assign it to the users.

C++ programming language with “Code Gear“ platform and “Firebird” Server DBMS was selected to perform work. Work consists of analysis of access control methods and implementation, requirements specification, summarized ERP security method formation, experiment realization, results and conclusions.

Santrauka

Šiame darbe yra nagrinėjami verslo valdymo programų prieigos teisių modeliai, kurie yra integruojami į verslo programas, kad apriboti vartotojų priėjimą prie informacijos resursų. Daugiau saugos pažeidimų sukelia patys įmonės darbuotojai, negu, kad išorėje esantys žmones, į tai reikia pirmiausia ir atkreipti dėmesį. Įmonėje viena verslo valdymo sistema naudojasi daug vartotojų, kurie turi skirtingas pareigas, roles. Šiuo darbu norima pasakyti, kad patogus prieigos teisių valdymo modulis turi būti neatsiejamas nuo įmonės vartotojų administravimo.

Pagrindinis darbo tikslas sukurti metodą gebantį patogiai valdyti DB ir ERP programos prieigos teises iš vienos vietos naudojant vieninga sąsają, nesigilinant DB ir programos prieigos teisių valdymo skirtumus.

Šiame darbe buvo sukurtas universalus verslo valdymo programų prieigos teisių ir audito informacijos valdymo metodas, užtikrinantis patogų vartotojo teisių valdymą bei šios informacijos perdavimą į duomenų bazių valdymo sistemą.

Darbui pasirinkta C++ programavimo kalbos įrankis „Code Gear“ bei „Firebird“ duomenų bazių valdymo sistema. Darbą sudaro prieigos teisių valdymo metodų ir priemonių analizė, reikalavimų specifikacija, apibendrinto verslo valdymo programos saugos metodo sudarymas, eksperimentinė realizacija su rezultatais bei išvados.

ĮVADAS

Informacijos apsauga – tai informacinių vertybių (materialių ir nematerialių) apsauga nuo nesankcionuoto sunaikinimo, pakeitimo ir prieigos. Informacijos apsauga neapsiriboja vien tik kompiuterių saugumu: daugelį problemų lemia netinkama fizinė apsauga, pavyzdžiui, neužrakintos durys arba pavogtas kompiuteris. Be to, problemų kyla ne vien tik dėl tyčinių įsibrovėlių veiksmų: neatsargaus darbuotojo sukeltas gaisras arba įdiegta kenksminga programa gali sutrikdyti prieigą prie informacijos, o svarbūs įstaigos duomenys gali būti prarasti visam laikui.

Prieigos valdymas skirtas apsaugoti įmonės duomenis nuo tyčinių ir netyčinių pavojų. Netyčiniai pavojai, kuriuos sukelia lojalių darbuotojų klaidingi veiksmai, yra jų menkos kvalifikacijos arba neatsakingumo padariniai. Tyčiniai pavojai gali apsiriboti pasyviu duomenų skaitymu arba sistemos stebėjimu, arba apimti aktyvius veiksmus, pavyzdžiui, vientisumo ir informacijos prieinamumo pažeidimus. Tyčiniai pavojai kyla dėl įsilaužėlių arba vidaus darbuotojų veiklos siekiant pasipelnyti ir yra nukreipti į žalos įmonei padarymą.

Legalaus vartotojo nelegalūs veiksmai – pavojai, kylantys dėl legalių programos vartotojų, kurie naudodamiesi savo įgaliojimais bando atlikti jų pareigų ribas viršijančius veiksmus. Pavyzdžiui, tinklo administratorius turi beveik neribotas prieigos prie visų tinklo išteklių teises. Vis dėlto įmonė gali turėti informaciją, prie kurios prieiti tinklo administratoriui uždrausta. Pamėginti imtis nelegalių veiksmų gali ir paprastas kompiuterių vartotojas. Statistika rodo, kad vos ne pusę visų mėginimų pažeisti sistemos saugumą padaro įmonės darbuotojai, kurie kaip tik ir yra legalūs kompiuterių vartotojai.

Prieigos autorizacijos priemonės kontroliuoja legalių vartotojų prieigą prie sistemos išteklių, suteikdamos kiekvienam iš jų būtent tas teises, kurios jam buvo nustatytos administratoriaus. Be vartotojų prieigos prie katalogų, failų ir spausdintuvų teisių suteikimo, autorizacijos sistema gali kontroliuoti vartotojų galimybę atlikti įvairias sistemines funkcijas, tokias kaip lokali prieiga prie serverio, sisteminio laiko nustatymas, duomenų rezervinių kopijų sukūrimas, serverio išjungimas ir panašiai.

Prieš pradedant gilinimąsi į magistrinį darbą apžvelgsime kai kurias pagrindines sąvokas. Žmonės, kurie sąveikauja su kompiuterio sistema yra vadinami vartotojais, o pats bendravimas su sistema yra vadinama sesija. Kompiuterio procesas, vykstantis dėl vartotojo veiksmų yra

vadinamas subjektu (naudojamos programos), o objektai yra sistemoje saugomi resursai (informacija), tai elementai, kurių prieigai reikalingos rolėm apibrėžtos teisės. Skaitymo leidimas užtikrina vartotojui tik peržiūrėti tą informacijos dalį, o rašymas apima skaitymą ir pridedama informacijos koregavimo galimybė. Žodis „įrašas“ šiame darbe atspindi vieną eilutę duomenų bazėje (pvz. eilutė susidedanti iš konkretaus vartotojo vardo, pavardės, adreso, telefono numerio).

Darbo tikslas yra sukurti universalų, mobilumu pasižymintį verslo programų prieigos teisių ir audito informacijos valdymo metodą, užtikrinantį patogų vartotojų teisių valdymą bei informacijos perdavimą į duomenų bazių valdymo sistemą.

Magistrinio darbo uždaviniai:

- išanalizuoti esamus verslo programų prieigos teisių informacijos valdymo metodus;
- nustatyti kuriamo metodo funkcinius ir nefunkcinius reikalavimus;
- sukurti verslo programų prieigos teisių ir audito informacijos valdymo metodą ir jį realizuoti;
- nustatyti sukurto metodo įtaką verslo programos stabilumui ir greitaveikai.

Sprendžiama verslo programų, naudojamų daugiavartotojiškoje aplinkoje, saugos ir prieigos teisių problema. Problema iškyla tada, kai viena programa naudojasi keletas vartotojų turinčių skirtingas teises ir jiems būtina užtikrinti skirtingą priėjimą prie programos formų ir/ar jų elementų.

Sukurtas universalus, mobilumu pasižymintis verslo programų prieigos teisių ir audito informacijos valdymo metodas, užtikrinantis patogų vartotojų teisių valdymą bei informacijos perdavimą į duomenų bazių valdymo sistemą. Metodas, sudarytas iš objektų, integruojamas į kiekvieną vartotojo sąsajos formą. Objektai kreipiasi į duomenų bazę ir, pagal prisijungusio vartotojo teises, pakeičia formos elementų būsenas (tik skaitymui ir/ar redagavimui). Taip pat administravimui sukurta vartotojų teisių valdymo sąsaja.

1. PRIEIGOS TEISIŲ VALDYMO METODŲ IR PRIEMONIŲ ANALIZĖ

1.1 Verslo valdymo programos apžvalga

Verslo valdymo programa – tai programinė įranga, skirta kompiuterizuoti įmonės valdymą, galinti apimti ir integruotis į visus įmonės verslo procesus, naudojama apskaitos vedimo palengvinimui, efektyviam visų resursų išnaudojimui, kontaktų valdymui, efektyviam tiekimo grandinės veikimui užtikrinimui, analitinės įmonės veiklos ataskaitų sudarymui [22].

Verslo valdymo programa leidžia darbuotojams atlikti greitesnius sprendimus, leisdama be kliūčių pasiekti tam tikrą įmonės informaciją. Tai yra pusiau baigtas produktas. Įmonės gali prisitaikyti jas sau pagal savo poreikius nustatant reikiamus parametrus, įjungiant ar išjungiant tam tikrus verslo procesus.

Įmonės, kuriose yra įdiegta verslo valdymo programa, supranta, kad apjungti verslo procesai duoda didelę vertę, padeda geriau vykdyti įmonės strategiją. Mažos bei vidutinio dydžio įmonės naudoja verslo valdymo programas, kad įgautų konkurencinį pranašumą.

Verslo valdymo sistemą sudaro trys pagrindinės technologijų dalys: duomenų surinkimas ir parengimas, duomenų saugykla ir galutinio naudotojo sąsaja [23]. Tai leidžia įmonei automatizuoti ir sujungti dauguma verslo procesų, dalintis bendrais duomenimis ir užduotimis, kurti ir prieiti prie informacijos realiu laiku.

Verslo valdymo sistema yra paketinė verslo programa susidedanti iš atskirų modulių, kurie pasirenkami pagal konkrečius įmonės poreikius. Į vieną sistemą galima integruoti ir skirtingų gamintojų modulius.

Dažniausiai verslo valdymo programa apima standartinius, pagrindinius įmonės veiksmus. Įmonės gali tas programas pasitobulinti ir pridėti nestandartines galimybes, bet tam reikia keisti programinės įrangos kodą. Galima atlikti mažus pakeitimus, tokius kaip patobulinti ataskaitas, ar didesnius pakeitimus, kai yra keičiamas programos pirminis kodas [22].

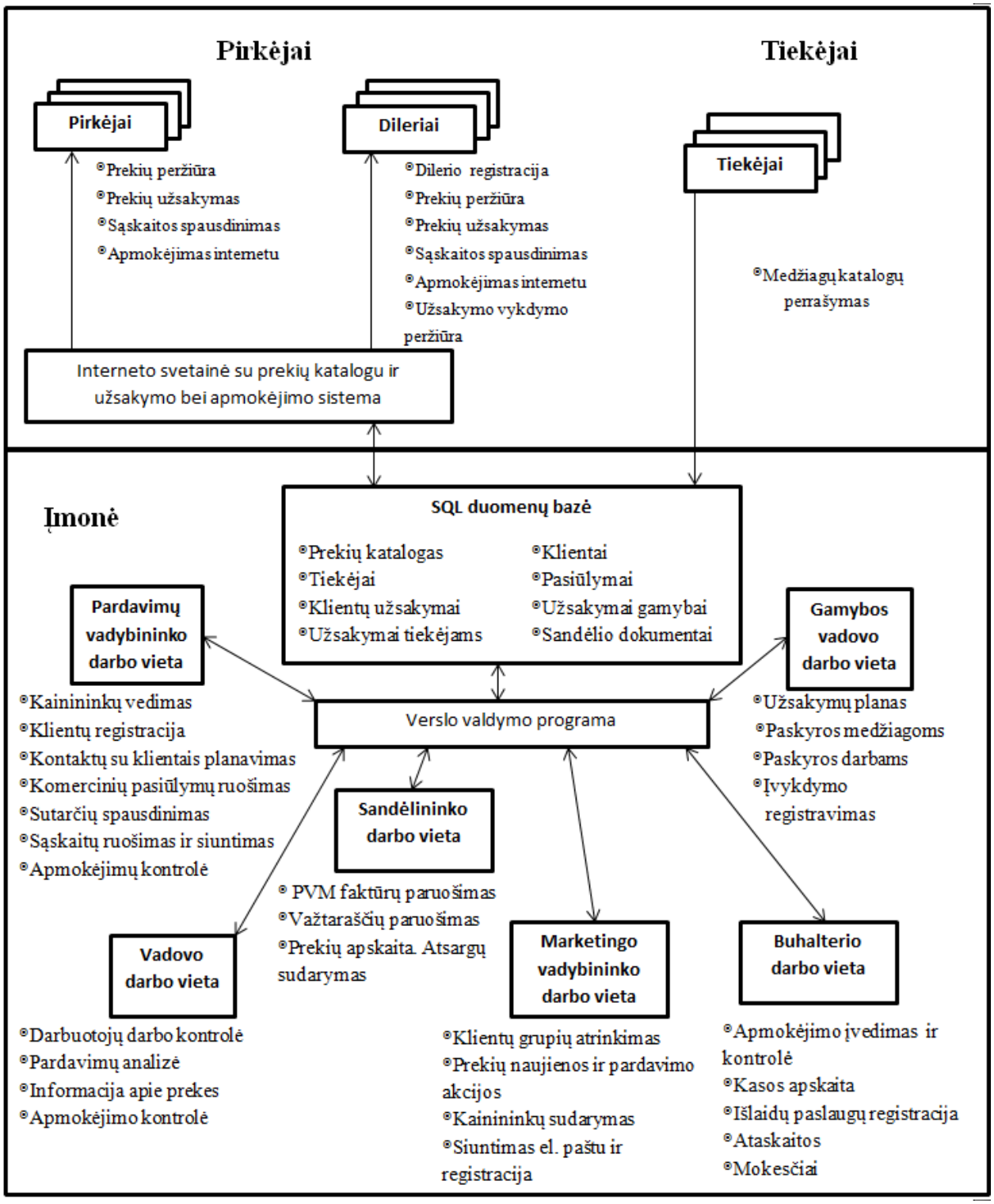
Programos modulių komplektas – tai verslo valdymo sistemos dalys (rinkinys), kuriose yra keletas verslo valdymo paketų, galinančių tarpusavyje apdoroti duomenis. Tokio programų komplekto aplinkoje galima atlikti įvairius dažnai daromus biuro darbus, pavyzdžiui, rengti ataskaitas, valdyti duomenis duomenų bazėje ir pan.

Programų modulių komplektą sudaro įvairūs programų paketų rinkiniai. Tokio rinkinio struktūra priklauso nuo naudotojų poreikių, t. y. kokiam naudotojui: stambiai, vidutinei ar smulkiai įstaigai bus skirtas komplektas [24]. Komplektas gali turėti pagrindinius ir pagalbinius komponentus, dažnai programų komplektą galima papildyti reikalingais ar naujais komponentais.

Dažniausiai pasitaikančios verslo valdymo sistemų dalys (jų gali būti daugiau ar mažiau, vienokių ar kitokių, priklausomai nuo konkrečių įmonės poreikių):

- Klientų-kontaktų valdymas
- Partnerių valdymas
- Darbuotojų valdymas
- Dokumentų valdymas
- Projektų / užduočių valdymas
- Prekių / paslaugų valdymas
- Kainodaros valdymas
- Sandėlio valdymas
- Gamybos valdymas
- Sąmatų valdymas
- Sąskaitų valdymas
- Komercinių pasiūlymų valdymas
- Tiekimo valdymas
- Aptarnavimo valdymas
- Turto valdymas
- Skolų valdymas
- Kalendoriaus valdymas
- Ataskaitų valdymas
- Istorijos valdymas
- Klientų sritis

Verslo programos pavyzdyje (žr. 1 pav.) pateikta darbuotojų užimamos pareigos su numatytomis veiklomis už ką būtent jie yra atsakingi. Darbuotojai naudojasi viena verslo programa, kuri pateikia duomenis iš duomenų bazės arba įrašo duomenis į ją.



1 pav. Įmonės vartotojų išsidėstymas.

1.2 Saugos politikos formavimo apžvalga

Saugumo strategija – tai veiklų rinkinys, nusakantis, kaip organizacija valdo, apsaugo ir skiria resursus siekdama užtikrinti organizacijos išsikeltus saugumo tikslus.

Įvairūs įstatymai, nutarimai, tvarkos ir instrukcijos reglamentuoja informacijos apsaugą valstybinėse įstaigose, privačiame sektoriuje ir visuomenėje. Šiuose teisiniuose ir norminiuose aktuose dažniausiai yra įtrauktas vienas arba keli informacijos konfidencialumo, vientisumo ar pasiekiamumo reikalavimai.

Svarbu yra suprasti visus galimus pavojus konkrečiai sistemai/programai ir apgalvoti apsisaugojimo taktiką nuo šių pavojų. Čia galima ir reikia panaudoti įvairiaplanius veiksmus ir priemones: moralines-etines ir įstatymų leidybos, administracines ir psichologines, programinės ir aparatinės kompiuterių įrangos apsaugos galimybes.

Saugumo užtikrinimo problemų svarba ir sudėtingumas reikalauja parengti informacinio saugumo politiką, kuri atsako į tokius klausimus:

•Kokią informaciją saugoti?

•Kokią žalą patirs įmonė, praradusi arba atskleidusi tuos ar kitus duomenis?

•Kokie yra galimi pavojaus šaltiniai, kokios rūšies atakų gali būti imtasi prieš sistemos saugumą?

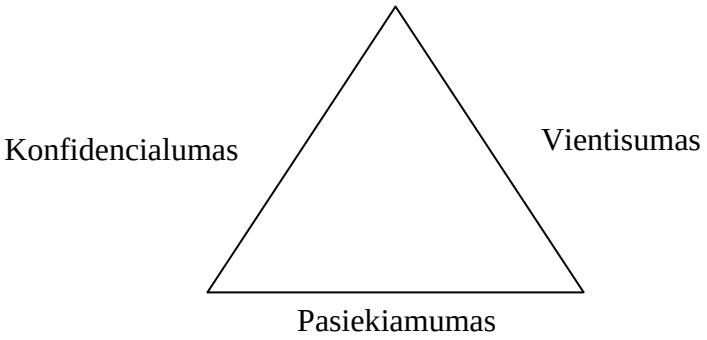
• Kokias priemones naudoti kiekvienos rūšies informacijos apsaugai?

Specialistai, atsakingi už sistemos saugumą, formuodami saugumo politiką, turi prisiminti keletą pagrindinių principų. Vienas iš tokių principų yra kiekvienam įmonės darbuotojui suteikti tik tą minimalų prieigos prie duomenų lygį, kuris būtinas jo pareigoms atlikti. Atsižvelgiant į tai, kad didelė dalis įmonės saugumo pažeidimų padaroma savų darbuotojų, svarbu įvesti aiškius apribojimus visiems kompiuterių vartotojams, nesuteikiant jiems bereikalingų galimybių ir teisių.

Dažniausiai saugumo strategiją apsprendžia prieigos valdymo modelis, kuris aprašo metodus, kaip įgyvendinti saugumo strategiją. Dauguma atveju tai yra matematinis modelis, kuris yra skaitomas formaliu modeliu tik tada, kai jis yra patikrinamas matematiškai.

1.3 Prieigos valdymo apžvalga

Prieigos valdymas - būdas valdyti prieigą prie įmonės resursų [1]. Šis apibrėžimas reiškia, kad prieigos valdymas yra mechanizmas, kuris valdo informacijos srautą tarp subjektų ir objektų. Tai yra trijų žingsnių procesas – KVP (Konfidencialumas, Vientisumas ir Pasiekiamumas) modelis (žr. 2 pav).



2 pav. KVP modelis.

KVP modelis vaizduojamas trikampiu, kurio briaunos atspindi pagrindines informacijos saugos charakteristikas:

- konfidencialumą – principą, reiškiantį, kad objektai (pvz., bylos, jose saugomi slapti duomenys) nėra atskleisti neautorizuotiems subjektams (vartotojams, programoms, procesams). Konfidencialumas garantuoja, kad duomenys nebuvo sukompromituoti. Konfidencialumo išlaikymas reiškia, kad skaityti ar suprasti slaptą informaciją gali tik patikimi subjektai;
- vientisumą – principą, reiškiantį, kad objektai išlieka teisingi (neiškraipyti), o juos pakeisti gali tik autorizuoti subjektai. Siekiama uždrausti neautorizuotas modifikacijas ar sunaikinti informaciją. Šios savybės išlaikymas užtikrina, kad subjektas A ir subjektas B iš tiesų yra tie, kuo prisistato, o subjekto A pateikti duomenys subjektą B pasiekia nepakitę;
- pasiekiamumą – principą, reiškiantį, jog autorizuotiems subjektams laiku suteikiama patikima prieiga prie objektų, o sąveikos sparta yra pakankama. Šios savybės išlaikymas vartotojams užtikrina kompiuterių tinklo veikimą ir galimybę pasiekti įmonės saugomus resursus reikiamu momentu.

1.4 Prieigos saugos modelių analizė

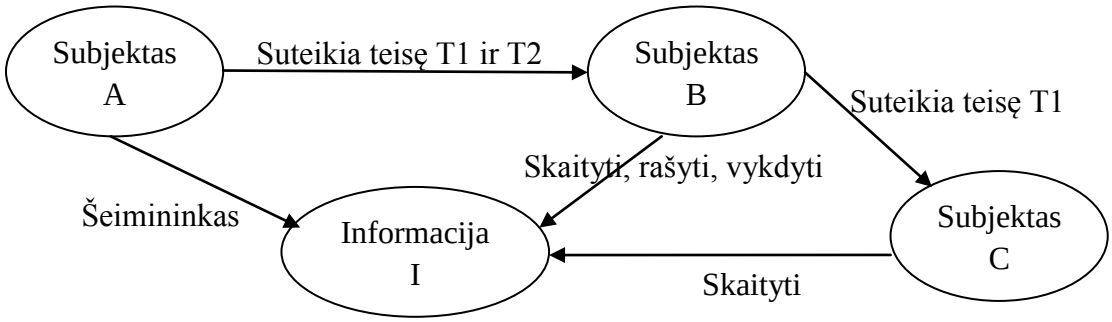
Prieigos valdymui yra naudojami trys skirtingi metodai, kurie įgyvendinami nepriklausomai: DAC, MAC ir RBAC. Kiekvienas iš jų turi savo privalumus ir trūkumus, kurie priklauso nuo to, kokioje aplinkoje jie yra realizuojami. Šiame darbe mes juos panagrinėsim, apžvelgsime privalumus ir trūkumus. Šiais metodais grindžiami formalieji saugos modeliai.

1.4.1 DAC prieigos valdymo modelis

DAC (angl. Discretionary Access Control) prieigos valdymo modelis – diskretinis prieigos valdymas, kuriame subjektas turi visišką priėjimo teisę prie informacijos, kurią jis pats kuria ir ji jam priklauso, bei pats nustato kokie dar subjektai ar jų grupė gali turėti priėjimą prie tos informacijos [6].

Privalumai yra tokie, kad galima suteikti žemiausias (pvz. tik skaitymui) reikiamas teises vartotojams, kad būtų atlikta tam tikra veikla. Kiekvienas objektas gali turėti tokį prieigos valdymą, kad apribotų kiekvieno subjekto prieiga su kiek galima minimalesnėmis, bet reikalingomis teisėmis [3].

DAC modelis yra lankstus, bet kartu ir sudėtingas (kai kada atsiranda paradoksinės situacijos). Pvz. Vartotojas A (žr. 3 pav.) yra informacijos I savininkas ir jis suteikė prieigos teises T1 ir T2 (skaityti ir rašyti, vykdyti) vartotojui B, pastarasis suteikia teisę T1 vartotojui C ir čia atsiranda problema, jei vartotojas A nuspręs nebeleisti prieigos vartotojui B, tai kas atsitiks vartotojo C prieigai prie tos informacijos [2]?



3 pav. *DAC modelis.*

Sistemos priežiūra ir saugumo tikrinimas yra labai sudėtingas DAC sistemose, nes vartotojai valdo savo objektų prieigos teises (be administracijos įsikišimo), nurodo kas gali prieiti prie tos informacijos. Kopijuojant informaciją iš vieno failo į kitą yra pastebimas apribojimų trūkumas.

1.4.1.1 Sąrašai

DAC dažniausiai įgyvendinamas naudojant ACL (angl. Access Control List) – prieigos valdymo sąrašą ir C-list (angl. Capabilities List) – C-sąrašą. Prieiga ribojama atsižvelgiant į vartotojų teises, gautas autorizacijos metu [4]. Abu autorizacijos būdai yra pagrįsti prieigos matrica (žr. 1 lentelę), kurios eilutėse nurodomi subjektai, o stulpeliuose – objektai (pvz., informacija). Kiekvieno subjekto ir objekto susikirtimo langelyje nurodytos subjekto prieigos prie objekto teisės (r-read (skaityti), w-write (rašyti), x-execute (vykdyti)).

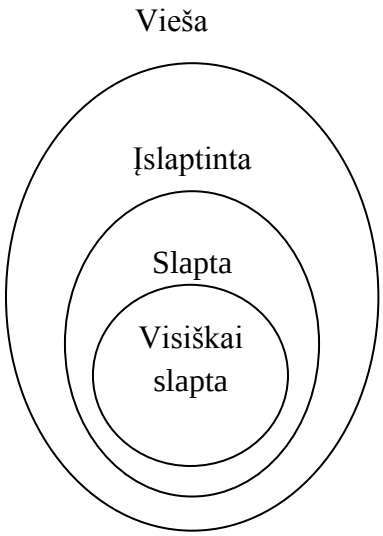
1 lentelė

	Failas.exe	**Failas.doc**	**Failas.com**
Vartotojas 1	-	Skaityti	Vykdyti, skaityti
Vartotojas 2	Vykdyti	Skaityti, rašyti	Vykdyti, skaityti, rašyti

1.4.2 MAC prieigos valdymo modelis

MAC (angl. Mandatory Access Control) prieigos valdymo modelis – privalomasis prieigos valdymo modelis, kuriame dokumento savininkas turi ribotą laisvę valdyti prieigos teises. Šis modelis yra priešingas DAC modeliui, yra saugesnis bei sunkiau konfigūruojamas [6]. Informacija yra suskirstyta pagal kategorijas ir kiekviena kategorija yra susieta su skirtingu slaptumo lygiu (žr. 4 pav.). Objektų lygius gali keisti tik administratorius, bet ne vartotojas, kuriam priklauso objektas. Tam tikras teises turintis subjektas gali prieiti tik prie tam tikro įslaptinto lygmens objektų pagal principą, vadinamą „reikia žinoti“. Šis modelis neleidžia įrašyti į objektą aukštesnio slaptumo lygmens informacijos negu to objekto įslaptinimo lygmuo. Pvz. informacija I yra labai svarbus įmonės resursas, kuris priskirtas „Visiškai slapta“ slaptumo

lygiui, ir kai subjektas, kuris turi aukščiausią prieigą tik prie „Slapta“ pažymėtų resursų slaptumo lygio, pateikia užklausą informacijai I pasiekti – jam yra išmetamas klaidos pranešimas. Šis modelis yra tinkamas, kai saugomai informacijai yra reikšmingas labai didelis jos konfidencialumas (pvz. karinių operacijų sistemoms) [3].



4 pav. MAC modelio slaptumo lygmenys.

MAC modelio atmaina BLP modelis su susieta daugiasluoksne apsauga yra pagrindinis modelis naudojamas prieigos valdymui karinėse ir žvalgybos agentūrose, kur yra labai griežta saugos politika. Taip pat šis modelis kartu su patikimais komponentais suteikia kelių sluoksnių apsaugą sistemai imunitetą nuo Trojos arklių, nes vartotojai neturi galimybės atslaptini objektų. [6].

Taip pat MAC modelis yra nesudėtingas ir gana tinkamas komercinėms sistemoms, kurios veikia „nedraugiškose“ aplinkose (web serveriams, finansinėms institucijoms), kur atakos rizika yra labai didelė, konfidencialumas yra primityvus ar kur apsaugoti objektus yra labai brangu [6].

Patikimus komponentus sudaro procesai ir bibliotekos, tokios kaip atslaptinantys kriptografijos procesai, kurie pažeidžia MAC principus ir kurie nelabai gali būti šiame modelyje. Tam kad palaikyti saugumo politiką ir neleisti neautorizuotam priėjimui prie informacijos šių komponentų kodas yra tik fiktyvus. Kaip bebūtų, praktika rodo, kad virtualiai neįmanoma

įgyvendinti kelių sluoksnių apsaugą naudojant MAC saugos modelį be operacinės sistemos ir daugumos susijusių veiksmų iškėlimo iš MAC modelio į patikimų komponentų aplinką.

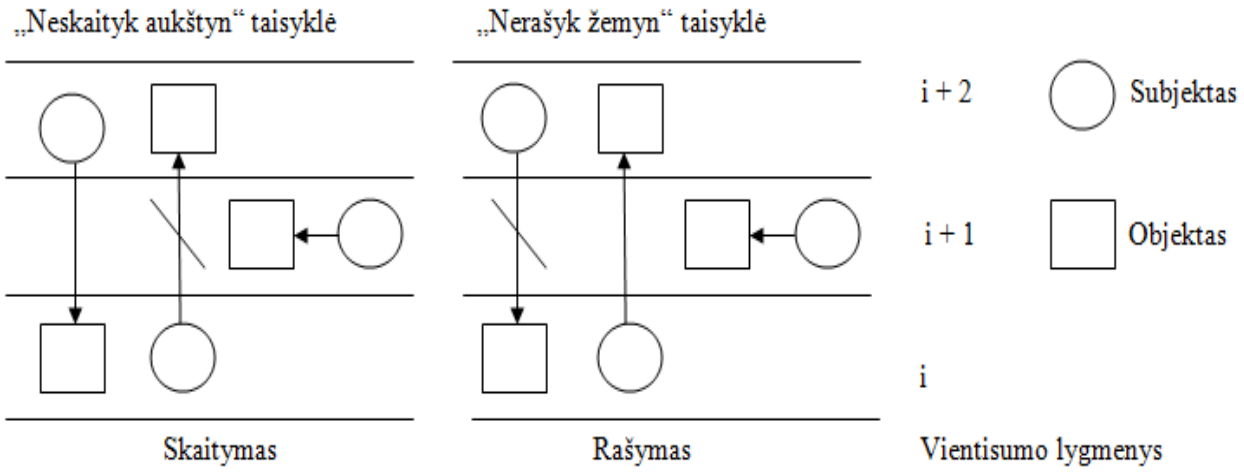
1.4.2.1 BLP prieigos valdymo modelis

BLP (angl. Bell-LaPadula) saugos modelis yra visiškai matematiškai formalizuotas. Jame numatytas privalomas kreipties valdymas (MAC), daugiausia dėmesio skiriama konfidencialumui [5]. Šis metodas pagrįstas subjekto ir objekto saugumo žymių palyginimui.

BLP modelis sukurtas remiantis daugiasluoksnės saugos (Multi-Layer Secure (MLS)) modeliu, kuris palengvina įslaptintos informacijos apsaugojimą. Šis modelis sudarytas pagal būsenų automato principą, kur būsenų automatas yra sudarytas iš tam tikro skaičiaus būsenų ir tarp dviejų bet kurių jų yra aiškiai nustatyti perėjimas.

BLP modelis naudoja dvi saugos savybes (žr. 5 pav.):

- Paprasta saugos savybė (Simple Secuirty Property), kuri suteikia teisę subjektui prieigą prie informacijos tik jei ji yra tame pačiame slaptumo lygyje arba žemesniame. Žemesniojo lygio subjektas negali skaityti aukštesniojo lygio objekto.
- * (žvaigždutės) saugos savybė (*-property), kuri suteikia teisę subjektui rašyti tame pačiame slaptumo lygyje arba aukštesniame. Aukštesniojo lygio subjektas negali rašyti į žemesniojo lygio objektą.

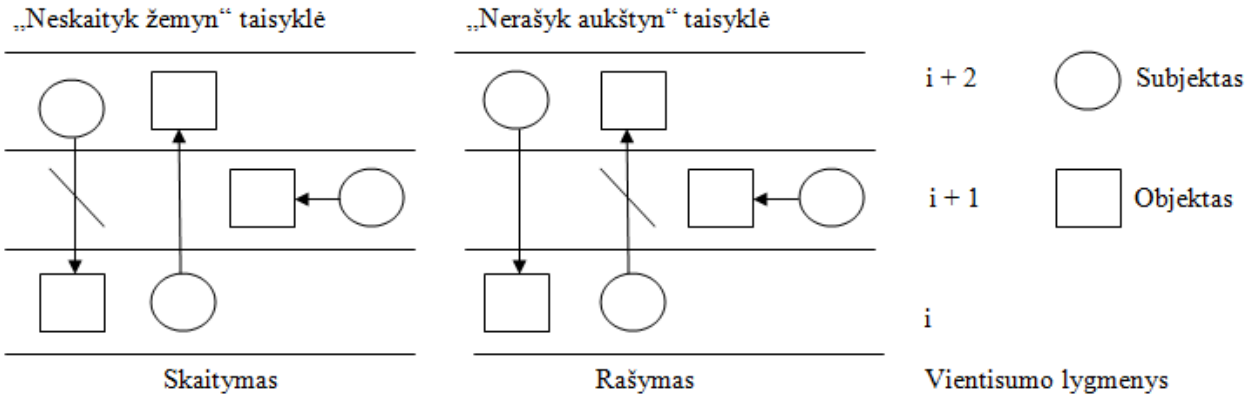


5 pav. BLP modelio lygmenys.

1.4.2.2 Biba prieigos valdymo modelis

Biba modelis yra sąlyginai panašus į BLP modelį, tik pagrindinis tikslas yra ne informacijos konfidencialumas, o jos vientisumas [5]. Vientisumas yra palaikomas griežtais skaitymo ir rašymo principais ir pastarieji yra visiškai priešingi BLP principams, t.y. subjektas gali skaityti objektus, esančius tik savo slaptumo lygyje arba aukštesniame, o rašyti tik į savo slaptumo lygyje esančius objektus arba žemesnius (žr. 6 pav.).

MAC metodas yra skirtas primityviems tikslams, kur konfidencialumas yra daug svarbesnis negu vientisumas, tai Biba modelis daro didelę įtaka tolesnių MAC metodų plėtrai.



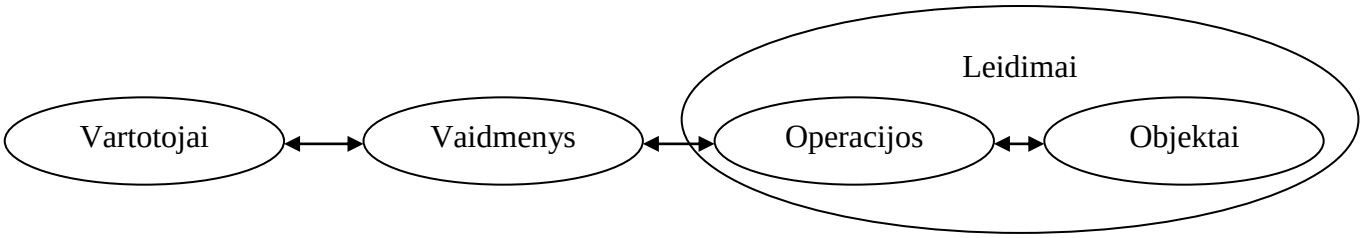
6 pav. Biba modelio lygmenys.

Bibos modelis – tai padvigubintas Bello ir LaPadulos modelis. Jo taisyklės yra tokios (O – objektų aibė, S – subjektų aibė, I – vientisumo lygių aibė):

1. $s \in S$ gali skaityti, $o \in O$ tada ir tik tada, jeigu $i(s) \leq i(o)$.
2. $s \in S$ gali rašyti, $o \in O$ tada ir tik tada, jeigu $i(o) \leq i(s)$.
3. $s1 \in S$ gali vykdyti $s2 \in S$ tada ir tik tada, jeigu $i(s2) \leq i(s1)$.

1.4. 3 RBAC prieigos saugos modelis

RBAC (angl. Role Based Access Control) modelis yra vaidmenimis (rolėmis) paremtas autorizuotų vartotojų prieigos valdymo modelis, kuriame sprendimai priimami atsižvelgiant į subjektų roles [8] (žr. 7 pav.), ir kuris reguliuoja priėjimą prie tinklo resursų priklausomai nuo vartotojų rolių įmonės viduje.



7 pav. RBAC modelis.

Rolės yra skirstomos atsižvelgiant į vartotojo pareigas, kompetenciją, autorizaciją ir atsakomybę [16]. Atitinkamai yra nustatomi ir leidimai. RBAC leidžia subjektams prieiti prie tam tikrų objektų, dinamiškai reguliuojant jų veiksmus atsižvelgiant į lankstumą, santykius ir apribojimus [11]. Rolės vartotojams lengvai kuriamos, keičiamos ar atšaukiamos grupėmis, nereikia atnaujinti privilegijas kiekvienam atskirai. RBAC modelis yra MAC ir DAC modelių alternatyva. Saugumo politika yra paremta daugiau atsižvelgiant į roles negu, kad į asmenis. Teisių suteikimas ir saugos politikos vykdymas yra apsaugos administratoriaus rankose ir vartotojams yra neleidžiama suteikti priskirtą leidimą rolei kitiems vartotojams.

NITS (National Institute of Standards and Technology) šį modelį suskirstė į keturis tipus, kurie pateikti pirmoje lentelėje (žr. 2 lent.). Modelis yra sudarytas iš keturių lygių, kurie didėjant paveldi ir prieš tai buvusio modelio funkcines galimybes [9, 10]. Kiekvienas lygis prideda po vieną naują reikalavimą.

2 lentelė

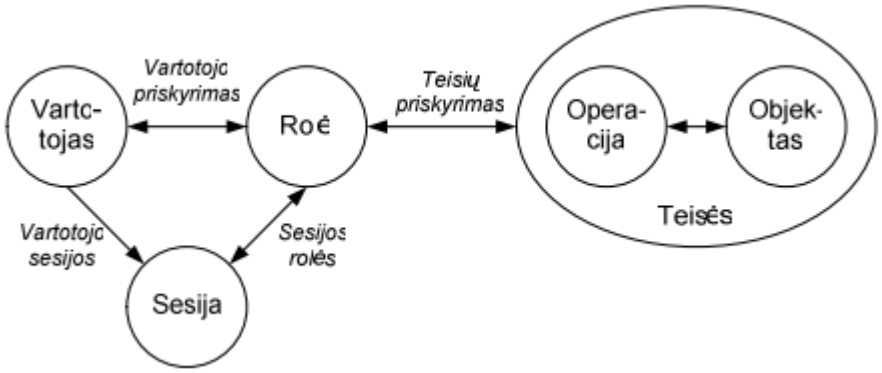
Lygis	Pavadinimas	Savybės
1	Bazinis RBAC modelis	• vartotojams suteikiamas leidimas per roles • palaiko daug-su-daug vartotojų - rolių priskirimą • palaiko daug-su-daug leidimų - rolių priskirimą • palaiko vartotojų - rolių priskirimo apžvalgą • vartotojai gali naudotis skirtingų vaidmenų leidimais vienu metu
2	Hierarchinis RBAC modelis	• palaiko Bazinį RBAC modelį • palaiko rolių hierarchiją (paveldimumas)
3	Ribojantis RBAC modelis	• palaiko Hierarchinį RBAC modelį • palaiko pareigų atskyrimo sąryšį
		• palaiko Ribojantį RBAC modelį

4	Simetrinis RBAC modelis	• palaiko leidimų – rolių apžvalgos palyginimą su vartotojų – rolių apžvalgą

Tranzakcijomis pagrįstos teisės padeda užtikrinti sistemos vientisumą ir galimybę kontroliuoti netik kokie resursai gali būti pasiekiami, bet ir kaip prieiga gali įvykti. Didelėse įmonėse grupės vartotojų prieigos valdymo sujungimas į vieną atskirą vaidmenį suteikia galimybę lengvesniam visos sistemos administravimui. Dar vienas modelio pranašumas yra tai, kad palaikomas žemiausios teisės principas (žemiausio leidimo suteikimas vartotojui atsižvelgiant į jo pareigas ir funkcijas), pareigų atskirimas ir centralizuotas prieigos valdymo bei vaidmenų administravimas [13, 14]. Pareigų atskirimas nėra palaikomas MAC modelyje, kuriame ir centralizuotas administravimas nėra visiškai palaikomas ir išvis yra neįmanoma DAC modelyje dėl saugumo principų pažeidimo [12].

1.4.3.1 Bazinis RBAC modelis

Šis modulis yra sudarytas iš minimalių elementų ir ryšių rinkinių, galintis visiškai išreikšti rolėmis paremtą prieigos valdymą sistemoje (žr. 8 pav.).



8 pav. Bazinis RBAC modelis.

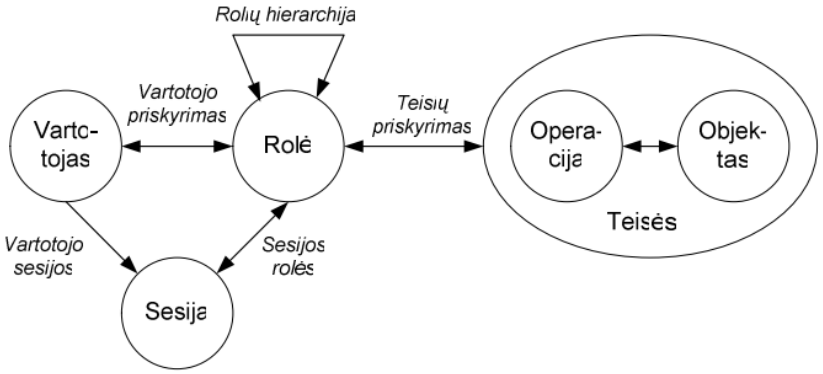
Sesija jungia vartotoją su aktyvuotų rolių rinkiniu. Sesijos vaidmens ryšys parodo sesijos metu aktyvuotas roles, o vartotojo sesijų ryšys parodo visas su vartotoju susietas sesijas. Vartotojui taikomos tos teisės, kurios priskirtos vartotojo sesijos metu aktyvuotam rolei.

RBAC valdomų operacijų ir objektų tipai priklauso nuo aplinkos. Pvz., failų sistemoje operacijos gali būti skaitymas, rašymas ir vykdymas, o duomenų bazių valdymo sistemoje galimos įterpimo, šalinimo, atnaujinimo ir kitos operacijos. Objektais laikomi elementai, kurių prieigai reikalingos role apibrėžtos teisės.

Vartotojams priskiriami rolės, o pastariesiem priskiriamos teisės. Tam naudojami vartotojo priskyrimo ir teisių priskyrimo ryšiai, kurių kardinalumas „ daug su daug“ . Tokiu būdu rolė leidžia susieti vartotojo ir teisės esybes vieną su kita ryšiu, kurio kardinalumas yra „daug su daug“. Toks sprendimas suteikia lankstumo ir skaidymo į gylį galimybę atliekant priskyrimus. Pavojus gali kilti dėl ribotos ryšių tarp vartotojui priskirtų rolių kontrolės. Administratorius turi nuspręsti, kokioms rolėms priskirti vartotoją, kad jam būtų suteiktos tik reikalingos teisės (mažiausių teisių principas).

1.4.3.2 Hierarchinis RBAC modelis

Hierarchinis RBAC modelio pavidalas gaunamas įtraukus rolių hierarchijos principą (žr. 9 pav.). Rolių hierarchija leidžia sudaryti organizacijos pavaldumo hierarchiją atitinkančias rolių struktūras. Galimi medžio, apversto medžio ir tinklelio vaidmenų hierarchijos tipai. Hierarchijos vaizdavimui modelyje naudojamas paveldėjimą reiškiantis ryšys. Paveldimumas apibūdinamas taip: rolė V1 paveldi rolę V2, jei visos V2 teisės yra ir V1 teisės. Įvairūs autoriai siūlo skirtingus paveldėjimo apibrėžimus ir interpretacijas.

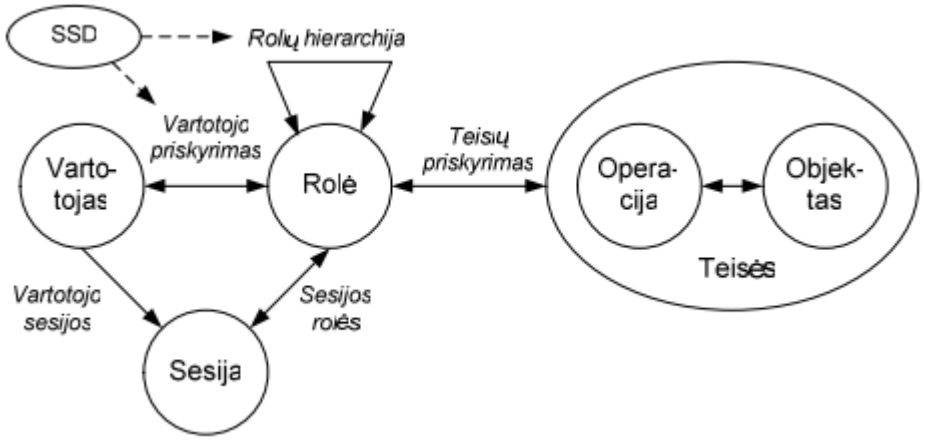


9 pav. Hierarchinis RBAC modelis.

1.4.3.3 Ribojantis RBAC modelis

Autorizacijos metu vartotojui gavus konfliktinėms rolėms priskirtas teises RBAC

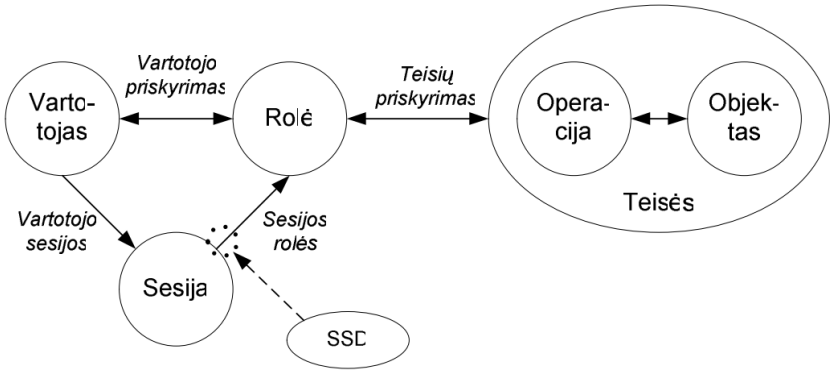
sistemoje gali kilti interesų konfliktas. RBAC su statiniu teisių atskyrimu (SSD) sumažina teisių, kurios gali būti suteiktos vartotojui, skaičių. Tai pasiekiama pritaikius apribojimus vartotojams, kurie gali būti priskirti kelioms grupėms. Apribojimai nustatomi konkrečiam vartotojui, t.y. visoje vartotojo teisių erdvėje (žr. 10 pav.).



10 pav. Ribojantis RBAC modelis.

1.4.3.4 Simetrinis RBAC modelis

RBAC su dinaminiu teisių atskyrimu (DSD), kaip ir su SSD, yra skirtas vartotojui priskiriamų teisių ribojimui. DSD nuo SSD skiriasi kontekstu, kuriame šie ribojimai įvesti. DSD atveju apribojimai nustatomi sesijos metu aktyvuojamoms rolėms (žr. 11 pav.). Kiekvienas vartotojas gali turėti skirtingo lygio teises atskirais laiko momentais, priklausomai nuo tuo momentu atliekamo vaidmens. Taip užtikrinama, kad turimos teisės naudojamos tik tuo metu, kai jos yra reikalingos darbui. Norėdamas įgyti kito lygio teises, vartotojas turi prisijungti prie kitos rolės, kuriai jis yra autorizuotas. Tokiu būdu išvengiama interesų konflikto [10].



11 pav. Simetrinis RBAC modelis.

1.4.4 Apibendrinimas

Atrandant prieigos valdymo modelį, kuris tenkintų visus įmonės reikalavimus yra labai sudėtingas procesas. Norint pritaikyti tinkamą prieigos valdymo modelį įmonei, pirmiausia reikia žinoti prieigos valdymo reikalavimus įmonės aplinkoje. Daugumoje kompanijų, skirtingose operacinėse sistemose, veikia didelis skaičius taikomųjų programų, ir atsižvelgiant į Gartner Group, jų skaičius ir įvairovė auga. Reikia atsižvelgti į įmonės dydį, veiklą, propaguojamą saugos politiką, vartotojų bendradarbiavimo reikalingumą tarp skirtingų pareigų ir vienodų ir t.t. Vidutinėms ir didelėms įmonės dažniausiai renkasi RBAC modelį siekiant palengvinti informacijos kūrimą ir dalinimąsi neprarandant jos konfidencialumo. Atsiradus naujam vartotojui vaidmenų pagalba yra lengvai priskiriami leidimai prie objektų, nereikia priskirti kiekvieną objektą atskirai (kaip DAC modelyje).

Įmonės aplinka keičiasi greitai (ekonominiai, techniniai veiksniai), tai reikalauja prieigos valdymo ir objektų greitų pasikeitimų. Tai daro prieigos autorizacijos administravimą labai sudėtingu dalyku, todėl čia reikalingas modelis, kuriuo galima būtų kuo greičiau ir lengviau vykdyti pakeitimus.

MAC modelis puikiai tinka ten, kur reikia didelio konfidencialumo, bet jis nėra lankstus, jį būtų gan sudėtinga administruoti, žinant kokių prieigos teisių detalumo lygio reikia įmonei. MAC strategijos tikslas – riboti subjektams priėjimą prie objektų priklausomai nuo objekte esančios informacijos saugumo laipsnio ir subjekto leidimo.

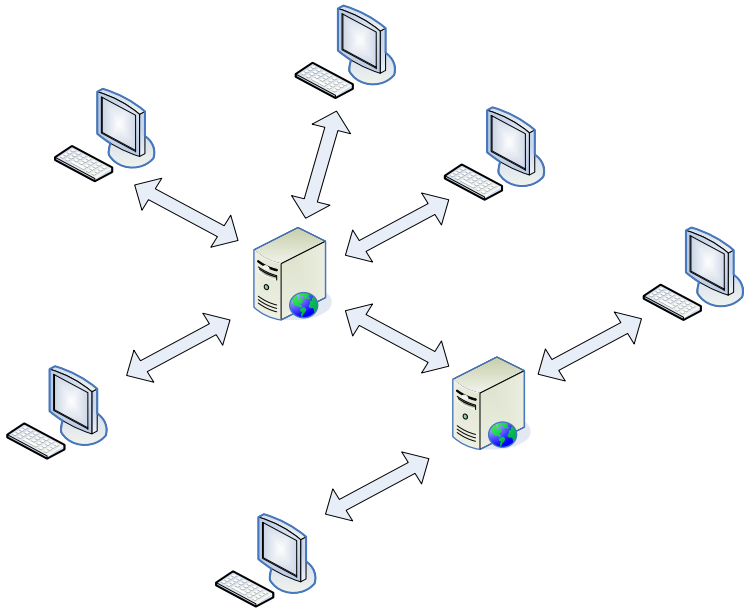
DAC modelis yra lankstus, bet nelengvai sukontroliuojamas, tokiu atveju reikia pasitikėti dauguma vartotojų, problemos su konfidencialumu. Sistemos priežiūra ir saugumo tikrinimas yra labai sudėtingas, nes vartotojai valdo savo objektų prieigos teises (be administracijos įsikišimo).

RBAC modelio pranašumas yra tas, kad jis yra naujausias iš visų trijų modelių, todėl jis vis dar yra tobulinamas, kuriamos visokios atmainos, pritaikant kiekvienai įmonei pagal poreikius. RBAC strategijos tikslas – riboti subjektų funkcionalumą ir informaciją. RBAC leidžia subjektams prieiti prie tam tikrų objektų, dinamiškai reguliuojant jų veiksmus atsižvelgiant į lankstumą, santykius ir apribojimus. Rolės vartotojams lengvai kuriamos, keičiamos ar atšaukiamos grupėmis, nereikia atnaujinti privilegijas kiekvienam atskirai, lengviau administruojamos.

1.5 Kliento serverio architektūros analizė

Kliento-serverio architektūra – tai architektūra, kuri atskiria klientą nuo serverio (sumodeliuota kaip serverių teikiamų paslaugų aibė ir jas naudojančių klientų aibė (žr. 12 pav.)). Klientai žino apie serverius, bet serveriai nebūtinai žino apie klientus, jie „klausosi“ ar niekas jų „nekviečia“. Procesų paskirstymas procesoriams nebūtinai yra 1:1.

Kliento–serverio tinklo modelis naudotinas, jei daugumai stambios įmonės darbuotojų reikalingas priėjimas prie didelių duomenų bazių ir jų valdymas, naudojantis duomenų bazių valdymo sistemomis. Tai tinklo terpė, kai kompiuteris klientas inicijuoja užklausą kompiuteriui serveriui ir pastarasis ją vykdo, tačiau dalis paruošiamojo darbo paliekama atlikti pačiam klientui [25]. Manipuliavimo duomenimis palengvinimui šiame modelyje kliento programinė įranga naudoja IBM sukurtą struktūrinių užklausų kalbą (angl. SQL, Structured Query Language). Manipuliavimą čia suvokiame kaip duomenų įvedimą, paiešką, išėmimą ir redagavimą. SQL interpretatorius verčia supaprastinta šnekamąja anglų kalba pateikiamas užklausas į mašininę kalbą.



12 pav. Kliento serverio dalių pasiskirstymas.

Kliento procesų funkcijos

Kliento procesas nori gauti reikiamus duomenis, pasinaudoti kito kompiuterio turimais ištekliais ar tiesiog atlikti skaičiavimus kitame kompiuteryje [25]. Visa tai yra dažniausiai daroma per vartotojo sąsają.

Atliekamos funkcijos:

- kliento procesas turi lokalizuoti serverį;
- pasirinkti orientuotą į susijungimą arba neorientuotą į susijungimą transportą;
- bendrauti su serveriu taikydamas sinchroninį ar asinchroninį būdą;
- siųsti serveriui užklausas pagal atitinkamo protokolo reikalavimus, priimti iš jo atsakymus;
- jis inicijuoja kontaktą su serveriu („šneka pirmas“).

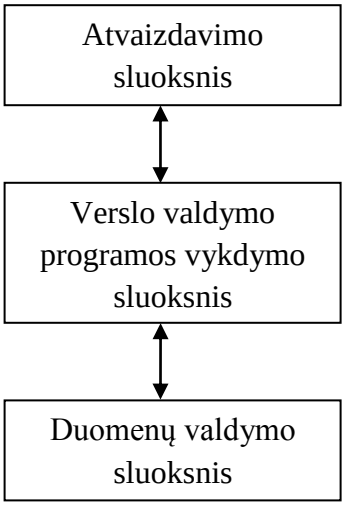
Serverio procesų funkcijos

Serverio procesas valdo duomenis bei kitus išteklius (resursus), leidžia klientams juos pasiekti, gali atlikti įvairius skaičiavimus. Teikia užprašytą paslaugą klientų procesams, pavyzdžiui, web serveris teikia klientams užprašytus web puslapius, pašto serveris teikia paštą.

Serverio procese turi būti numatytas klaidų apdorojimas, atstatymo po nulūžimo veiksmai, paprastai yra vykdomas kitame tinklo kompiuteryje nei kliento.

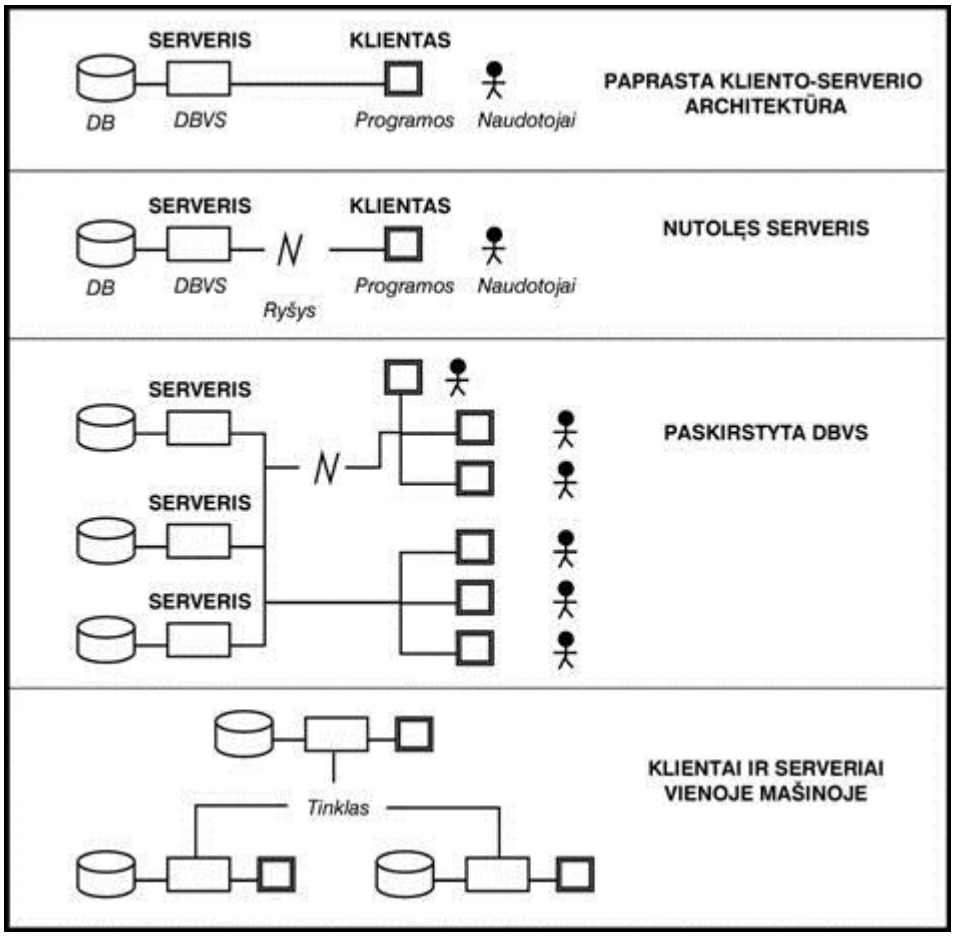
Verslo valdymo programų sluoksniuota architektūra

Verslo valdymo programų sluoksniuota architektūrą sudaro trys sluoksniai (žr. 13 pav.): atvaizdavimo sluoksnis, verslo valdymo programos vykdymo sluoksnis ir duomenų valdymo sluoksnis. Pirmasis skirtas vartotojo duomenų įvedimui bei rezultatų atvaizdavimui, antrasis skirtas programos specifiniam funkcionalumui (pvz., naujo vartotojo susikūrimas, ištrynimas) ir trečiasis skirtas duomenų bazės valdymui.



13 pav. Trys kliento-serverio architektūros sluoksniai.

Duomenų bazės serveris – techninė platforma, kurioje veikia DBVS (visuose trijuose lygmenyse). Klientas yra taikomoji programa arba individualus naudotojas. Paveiksle (žr. 14 pav.) parodytos paprasta ir paskirstytos duomenų bazių valdymo sistemos. Tais atvejais, kai sistema aptarnauja daug naudotojų, serveris ir klientas yra skirtingos mašinos. Galima ne tik atskirti klientą ir serverį, bet ir sujungti į tinklą daugelį mašinų taip, kad jos dirbtų lygiagrečiai. Galima paskirstyti ir pačią duomenų bazę tarp keleto mašinų, taip sudarant duomenų bazių serverių tinklą. Jei yra galimybė kliento užklausą apdoroti keliuose serveriuose vienu metu – tai yra paskirstyta DBVS. Bet kuriuo atveju klientui neturi būti svarbu, kas fiziškai sudaro serverį (viena ar kelios mašinos) ir kur jis yra.



14 pav. Kliento-serverio architektūros tipai.

1.5.1 Apibendrinimas

Kliento-serverio architektūra leidžia atsakomybę už darbą padalinti keliems nepriklausomiems kompiuteriams, tokiu būdu paprasta serverį taisyti, pakeisti, atnaujinti ar perkelti to net nepajaučiant klientams. Toks sprendimas leidžia duomenys saugoti centralizuotai, tokiu būdu paprasčiau užtikrinti duomenų apsaugą ir jų valdymą serveriuose tiksliai nurodant priėjimo prie duomenų taisykles. Palyginus su taško į tašką (angl. Point to point (P2P)) tinklais, duomenys greičiau administruojami. Norint atnaujinti duomenis taško į tašką tinkluose, reikėtų tai padaryti kiekvienam mazgui, o tai yra itin daug laiko užimanti užduotis.

Dauguma dabartinių prieinamų šios architektūros sprendimų turi integruotą pakankamai gerą sąsają ir neblogas saugumo funkcijas. Sistema suteikia galimybę dirbti įvairių tipų klientams.

Ši architektūra turi ir trūkumų, didėjančio klientų užklausų kiekio neatlaikęs serveris gali tiesiog persikrauti. Kai serveris būna perkrautas, klientų užklausos negali būti įvykdytos.

1.6 Verslo programos saugos architektūros analizė

Duomenų bazės saugumas skirstomas į dvi kategorijas. Sistemos apsauga ir duomenų apsauga. Sistemos apsauga apima priėjimą prie sistemos ir duomenų sistemos lygyje nustatant vartotojo vardą ir slaptažodį, diskinę erdvę ir sistemines operacijas leidžiamas vartotojui. Duomenų saugumas apima priėjimą prie duomenų bazės objektų ir veiksmus kuriuos vartotojas gali atlikti su objektais.

Leidimai yra teisė vykdyti ypatingus SQL operatorius. Duomenų bazės administratorius yra aukščiausio lygio vartotojas su galimybe leisti vartotojui pasiekti saugomus objektus. Vartotojui reikia sisteminių leidimų pasiekti duomenų bazę ir objektų leidimus, kad manipuliuoti objektais duomenų bazėje. Vartotojams gali būti suteikta papildomas leidimas suteikti leidimus kitiems vartotojams ar roles, kurios yra vadinamos grupinėmis.

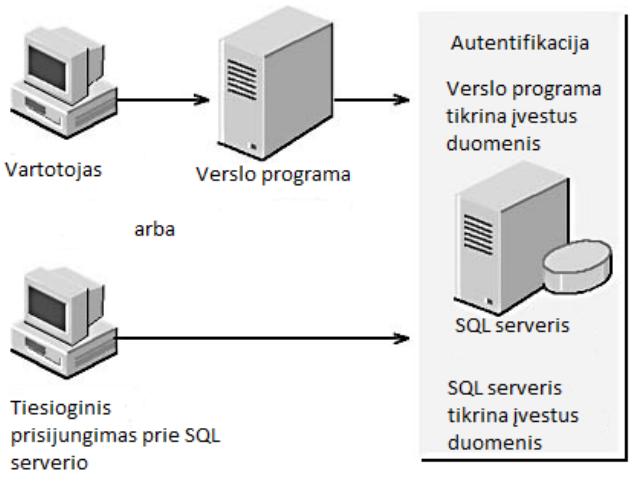
Bendrą leidimą prieiti prie duomenų bazės suteikia jos administratorius, užregistruodamas vartotojus ir sukurdamas duomenų bazėje naujus vartotojus. Užregistruodamas vartotoją, administratorius nurodo jo vardą ir slaptažodį bei leidimus. Tik po to vartotojas gali pradėti dirbti su duomenų baze prisijungdamas prie jos, nurodant savo vardą ir slaptažodį. Tada vartotojui taikomi sisteminai reikalavimai per visą prisijungimo seansą dirbant su duomenų bazės objektais. Priklausomai nuo modelio, kartą vartotojo sukurtas objektas, kuriuo vėliau vartotojas naudojasi su visom teisėm, gali arba negali būti perleistas kitiems vartotojams.

Administratorius gali vartotojui suteikti įgaliojimus dirbti tik su informacija, kuri skirta jam. Tokiu būdu vartotojas gali kreiptis tiktai į jam priskirtas lenteles ir atlikti su tom lentelėm tiktai jam skirtus veiksmus. Pats administratorius gali kreiptis į bet kurias duomenų bazės lenteles ir gali atlikti bet kuriuos veiksmus. Administratorius gali nustatyti, kokie vartotojai gali kurti lenteles ir atvaizdus, kokie vartotojai gali kurti lentelių sritis ir jas koreguoti ir kokie gali taisyti duomenis lentelėse. Privilegijos į objektą gali būti atšauktos.

Duomenų bazės valdymo sistema patvirtina vartotojus dviem lygiais. Prisijungimo metu naudojamas autentifikavimas užtikrina, kad vartotojas yra atpažįstamas, turintis teisę prisijungti prie SQL serverio. Leidimų suteikimas nurodo, kad vartotojas yra autorizuotas naudoti tam tikrus duomenų bazėje esančius objektus.

Autentifikavimas

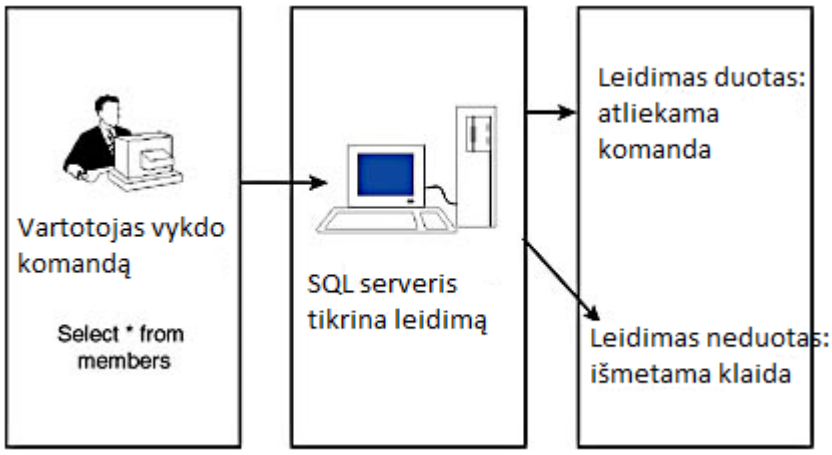
Norint prisijungti prie serverio, vartotojas privalo turėti savo prisijungimo duomenis (žr. 15 pav.). Prie serverio galima jungtis dvejopai: naudojant specialią verslo programą arba tiesiogiai naudojant SQL serverio prisijungimo langą. Projektuojant verslo valdymo programos prieigos teisiu metodą reikia atsižvelgti į šiuos abu dalykus, nes kas iš to jeigu bus įgyvendintas prieigos valdymo modelis programoje, bet bus kitas būdas kaip jį apeiti ir disponuoti neleistinais objektais.



15 pav. Du autentifikavimo būdai.

Leidimai

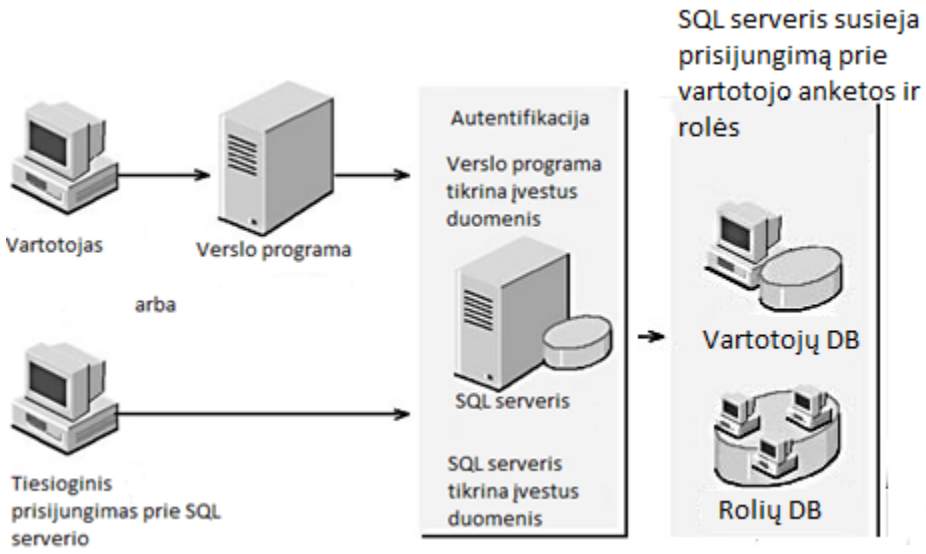
Prisijungęs vartotojas prie programos paprastai vykdo norimas užklausas, veda duomenis į formas ir t.t. Programose, kuriose nėra aiškiai apibrėžtų leidimų vartotojui iškyla papildomų sunkumų, nes kiekvieną kartą vartotojui siunčiant užklausą yra patikrinama ar vartotojas turi teisę tai daryti (žr. 16 pav.), jei ne – yra išmetama klaida. Šiuo atveju visa vartotojo įvesta ar koreguota informacija neišsisaugo duomenų bazėje.



16 pav. Leidimų tikrinimas.

Vartotojų paskyros ir rolės

Vartotojų vardai ir rolės yra susietos tarpusavyje ir nusako su kokiais objektais ką vartotojas gali daryti (žr. 17 pav.). Leidimai užklausom vykdyti ir naudoti objektus yra suteikti vartotojams ir rolėms. Vienas vartotojas gali turėti kelias roles, jei tik tai neviršija minimalų leidimų kiekį, kurio jam reikia savo darbui atlikti.



17 pav. Vartotojai ir rolės.

1.7 „Firebird“ duomenų bazės apžvalga

„Firebird“ yra gana populiari komercinio duomenų bazės serverio „Borland Interbase 6.x“ atviro kodo atšaka. Tai duomenų bazės valdymo sistema siūlanti daugelį ANSI SQL-92 ypatybių. Ji veikia „MS Windows“, „Linux“ bei kitose „UNIX“ platformose. Pasirinkau ją todėl, nes ji pastaraisiais metais gavo du apdovanojimus kaip geriausia duomenų bazė tinkanti įmonėms, bei palaiko vartotojų prieigos teisių valdymą prie bet kurio lentelės lauko atskirai.

Nesirinkau komercinių duomenų bazės valdymo sistemų norėdamas parodyti, kad ir su populiaria tarp įmonių, bet nemokama sistema yra galimybė įsidiegti į programą prieigos teisių valdymo modelį.

Saugus prisijungimas

Firebird duomenų bazėje nėra tam integruotų funkcijų, yra galimybė naudoti SSL, galbūt tai bus įdiegta vėlesnėse versijose. Dabar galima naudoti bendras „tunelio“ programas (SSH, SSL, ZeBeDee) [21].

Duomenų bazės šifravimas

Taip pat Firebird neturi integruoto šifravimo, bet yra galimybė pritaikyti įvairius sprendimus. Pirmas yra užšifruoti visus duomenis kliento pusėje prieš saugant juos į duomenų bazės stulpelius ir atšifruoti skaitant. Vienintele problema dėl to yra adresavimas ir paieškos vykdymas. Kitas sprendimas yra užšifruoti failų sistemą, kurioje yra patalpinta Firebird duomenų bazė. Tam yra sukurtos tokios platformos kaip TrueCrypt, EncFS or BestCrypt [21].

Pagrindinis TrueCrypt privalumas, lyginant su kitomis šifravimo programomis yra vadinamasis „įtikinamas neigimas“ (angl. plausible deniability). Tai reiškia, kad duomenys yra ne tik užšifruoti, bet ir paslėpti taip, kad

1. niekas nė neįtartų juos tokius esant kompiuteryje;
2. netgi žinant, kad kažkokių užšifruotų duomenų kompiuteryje esama, galima būtų išvengti jų dešifravimo su „atsarginiu“ slaptažodžiu.

Visų pirma TrueCrypt sukuria „indą“ failams – visą standųjį ar USB diską ar tiesiog failo pavidalo talpyklą kitiems failams. Visas turinys joje automatiškai užšifruojamas pasirinktu

šifravimo. Uždarius TrueCrypt ir failą, duomenys tampa nebeperskaitomi. Be to, failas gali turėti absoliučiai bet kokį pavadinimą ir išplėtimą, arba neturėti jokio išplėtimo. Pavadinę saugyklą „system.dat“, „xtmload.vbx“ ar „backup“ lengviau išvengiama smalsios akies.

TrueCrypt leidžia apsisaugoti ir nuo tokių situacijų, kai jūs esate dėl kokios nors priežasties priversti atidaryti archyvą. Tuomet galima įrašyti ne tikrąjį, bet atsarginį slaptažodį, kuris „atrakina“ saugyklą, bet rodo visai ne pagrindinį paslėptą turinį, o specialiai tam atvejui įdėtus apgaulingus duomenis – galbūt banko ar telefono ataskaitas, šiaip kokius „įtikinamai slaptus“ failus.

Teisės įrašui, laukui, lentelei

Valdant teises yra naudojamos GRANT ir REVOKE būsenos:
GRANT UPDATE ON table1(field1) TO USER1;
REVOKE UPDATE ON table2(field2) TO USER2;

Norint apriboti vartotojų skaitymo prieigą prie įvairių laukų yra naudojama:
create view v1 (limited column list)
as
select limited,column,list
from t1;

Vartotojui peržiūrint duomenų bazėje esančią lentelę taip pat galima apriboti kuriuos įrašus (eilutes) jis gali peržiūrėti:
create view v1 (column,list)
as
select column,list
from t1
where ...constraining clause...;

Taip pat, reikalui esant, galima sudaryti procedūrą, kuri tam tikriems stulpeliams peržiūrėti gražina nulio reikšmę

2. REIKALAVIMŲ SPECIFIKACIJA

Kuriamai sistemai yra keliami atitinkami reikalavimai. Jie yra skirstomi į dvi grupes: funkcinius ir nefunkcinius.

2.1 Funkciniai reikalavimai

1. Užsiregistruotam naujam vartotojui turi būti suteikta prieiga su nustatytomis žemiausiomis prieigos teisėm.
2. Prieš pradedant naudotis sistema, vartotojui privaloma pateikti prisijungimo langą.
3. Vartotojas turi būti autentifikuotas ir autorizuotas.
4. Metodas turi užtikrinti, kad prieigą prie įmonės visų duomenų, rolių bei leidimų konfigūravimo gali turėti tik administratoriaus tipo vartotojai.
5. Metodas turi turėti vartotojų klasifikavimo į roles logiką.
6. Metodas turi turėti leidimų rolėms suteikimo logiką.
7. Metodas turi leisti vartotojui turėti daugiau negu vieną rolę vienu metu.
8. Programoje rolių teisės yra aukštesnio prioriteto ir vartotojas, priklausantis tam tikroms rolėms, negali turėti teisių, aukštesnių nei rolių teisės.
9. Metodas turi turėti galimybę prieigos teisės nustatyti langų, lentelių, įrašų lygiais.
10. Metodas turi leisti programos teises nustatyti lango komponento lygiu, jei lango komponentas yra įgyvendintas kaip atskiras vienetas ir turi nuosavą teisę sistemoje.
11. Kiekvienas iškviečiamas langas turi turėti jam priskirtą teisę, kuri apspręstu jo prieigos lygį tam tikrai vartotojų rolei ir priklausomai nuo to jis turėtų būti aktyvus (matomas) arba neaktyvus (nematomas).

2.1 Nefunkciniai reikalavimai

1. Metodas privalo apdoroti klaidas taip, kad jose talpinama informacija neatskleistų programos sandaros ypatybių, SQL duomenų bazės lentelių, lentelių laukų ir jų tipų pavadinimų bei kitos svarbios informacijos.
2. Metodas turi sugeneruoti teisingas komandas, užtikrinti patikimumą ir kokybę.

3. Metodas turi leisti administratoriui stebėti sistemos veikimą, daryti sistemoje koregavimus, ir atsiradus problemoms jas greitai pašalinti.

4. Vartotojo sąsaja turi būti paprasta, lengvai suprantama ir valdoma vartotojui, kuris siekia greičiau pasiekti rezultatų/gauti atsakymą.

5. Vartotojo sąsajos tekstas turi būti aiškiai suprantamas, be lietuvių kalbos klaidų, tinkamo šrifto ir spalvų derinio (neerzinančių akių), vienprasmiškas.

6. Vartotojo sąsajos elementų išdėstymas turėtų būti nuoseklus ir išdėstytas tokiu principu, kad dažniausiai naudojami komponentai ar atskiro meniu punkto pasirinkimai būtų aukščiau, o mažiau naudojami – žemiau.

7. Meniu elementai turėtų būti suskirstyti į logiškas kategorijas, kad vartotojui būtų lengviau juos rasti.

8. Programa turi būti tokia, kad vartotojai galėtų ja lengvai naudotis turint minimalų kompiuterinį raštingumo lygį.

9. Vartotojų klaidos neturi sukelti programos kritinės būsenos.

10. Sistema turi dirbti nenutrūkstamai.

3. APIBENDRINTO VERSLO VALDYMO PROGRAMOS SAUGOS METODAS

Siekiant išvengti bereikalingų klaidų pranešimų, kuriuos vartotojai gauna kai bando įvesti ir išsaugoti informaciją, bet tam neturi leidimo, verslo programos modulis prisijungus vartotojui iškarto nustatys jo teises ir jam pateiks tik tuos laukus, įrašus ar lenteles, kuriais jis galės naudotis.

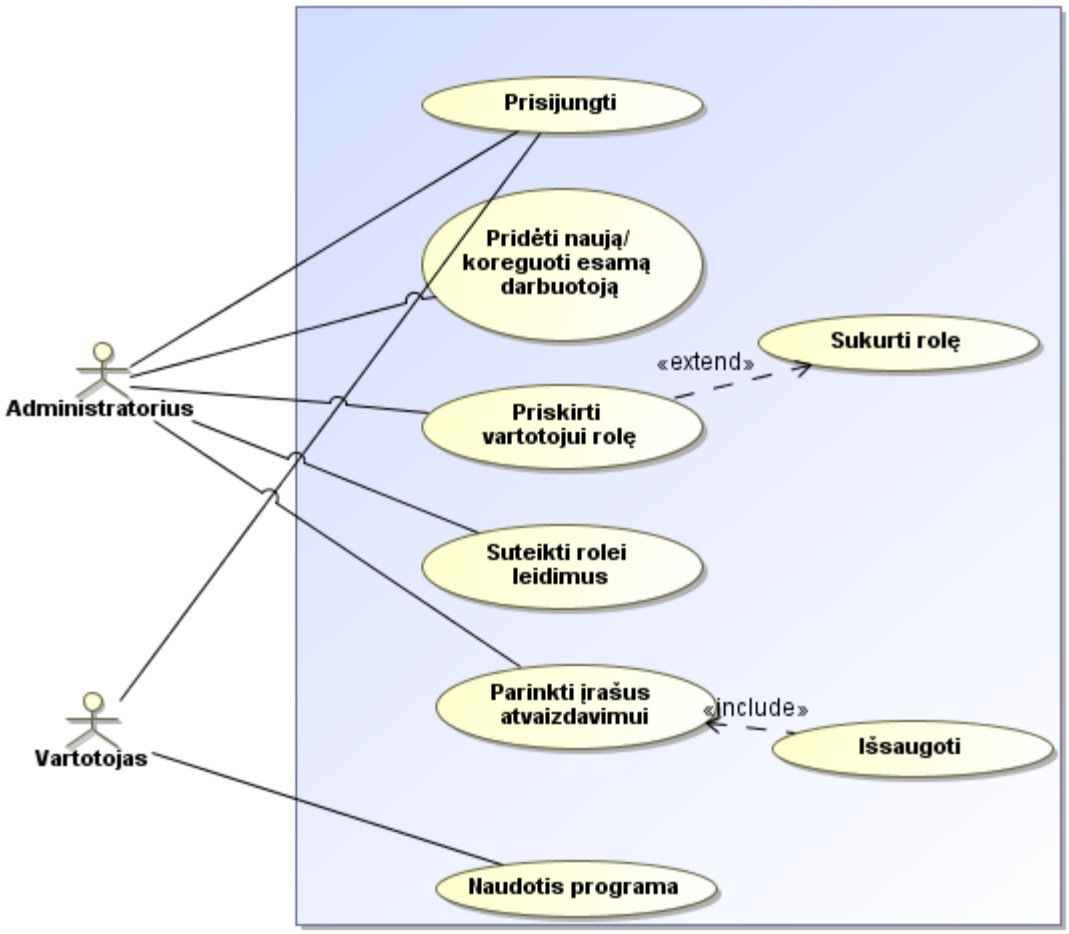
Tokiu būdu programa tampa „draugiškesnė“, patogesnė darbuotojams, tiek paprastiems tiek tą programą administruojantiems žmonėms. Vartotojas iškarto mato tik tai ką gali padaryti, kas yra jo kompetencijoje. Šiuo atveju prieš išsaugant informaciją vartotojas bus tikras, kad šią veiklą jis galėjo padaryti pagal jam suteiktas prieigos teises.

Administruojant leidimų priskirimus rolėms, kartais kyla problemos, kai norima rolei priskirti daugiau leidimų iš skirtingų pareigų kategorijų, bet sistema to padaryti neleidžia. Administratoriui yra siūloma vartotojui priskirti dvi ar daugiau rolių, bet tokiu atveju vartotojas gali turėti pernelyg daug teisių, negu, kad jam reikia darbui atlikti, iškyla grėsmė informacijos konfidencialumui. Todėl metode bus įgyvendintas prieigos teisiu valdymas visiem programos komponentams nepriklausomai nuo kitų komponentų.

Vartotojai, jų rolės ir leidimai bus saugomi tiek pačioje duomenų bazėje naudojant lenteles (vartotojai, roles, leidimai), tiek ir naudojant pačios duomenų bazės vartotojus bei jų roles (kai yra jungiamasi tiesiogiai prie duomenų bazės per SQL serverį).

3.1 Metodo panaudojimo atvejų modelis

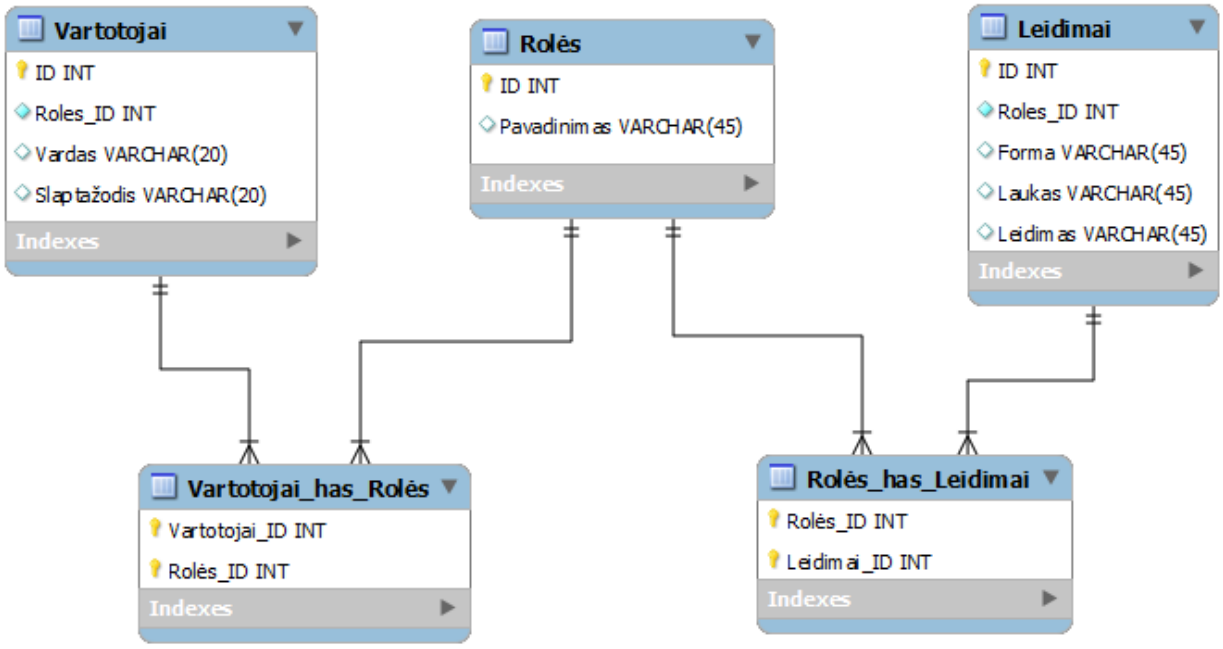
Panaudojimo atvejų diagrama (žr. 18 pav.) parodo kokius veiksmus gali atlikti vartotojas. Ši diagrama atspindi metodo administravimą suteikiant, tam tikrus leidimus vartotojams. Jei autentifikuotas vartotojas yra administratorius, tai jis sukuria naujus vartotojus, priskiria jiems roles, nustato joms leidimus formos, mygtuko, lauko ir įrašo lygiais. Jei autentifikuotas vartotojas yra paprastas darbuotojas, tai jam sudaroma programa ir jis ją gali naudotis.



18 pav. Vartotojų panaudojimo atvejų modelis.

3.2 Metodo duomenų bazės struktūra

Duomenų bazės koncepcijoje vartotojų lentelė (žr. 19 pav.) pavaizduota diagrama, kurioje pateikta vartotojų, vartotojų rolių, teisių ir komponentų logika, kuria remiantis įgyvendinama teisių politika. Rolių lentelė gali turėti daug vartotojų ir vartotojai gali turėti kelias roles, todėl jos yra susietos daug su daug ryšiu (per papildomą lentelę). Rolių lentelė su leidimų lentele yra surištos daug su daug ryšiu (taip pat prie papildomą lentelę).



19 pav. Lentelių ryšiai.

3.3 Rolių ir leidimų valdymas jungiantis per SQL serverį

Tiesioginis prisijungimas prie „Firebird“ duomenų bazės galimas tiek per komandinę eilutę, tiek naudojant kitų gamintojų specialiai suprojektuotas vartotojo sąsajas. Prisijungimo principas yra panašus, reikia nurodyti vietą, kurioje yra duomenų bazė ir tada įvesti prisijungimo duomenis. Pagal juos bus suteikti leidimai prie objektų, administratoriui taip pat prie rolių valdymo.

Teisių valdymas SQL duomenų bazės vartotojams:

1. Sukuriamos rolės

   CREATE ROLE firstdbadmin;
2. Rolėms priskiriamos/atšaukiamos teisės lentelėms

   GRANT SELECT, UPDATE, INSERT, DELETE ON sales_catalog TO ROLE firstdbadmin;

   REVOKE DELETE ON sales_catalog TO ROLE firstdbadmin;
3. Vartotojams priskiriamos rolės

   GRANT firstdbadmin TO TestAdmin;

Tiesiogiai su SQL duomenų baze dirbant visi pakeitimai daromi/matomi priklausomai nuo naudojamos aplinkos (komandinės eilutės, trečių šalių programos), bei leidimų, suteiktų atitinkamam vartotojui.

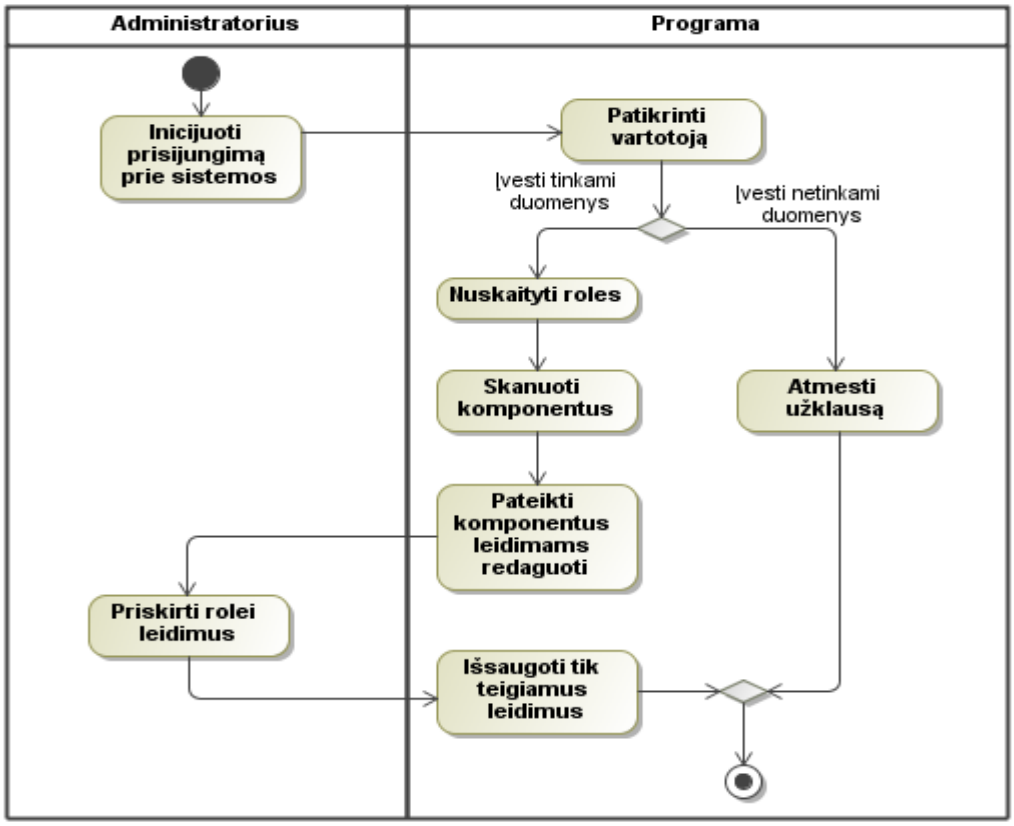
3.4 Prieigos valdymo metodo įgyvendinimas

Metodo realizavimui buvo panaudotas C++ programavimo kalbos įrankis „CodeGear“. Buvo pasinaudota šio įrankio komponentais lengvesniam „bendravimui“ su duomenų baze.

3.4.1 Leidimų veikimo principas (admistratorius)

Kaip metodo realizacija buvo sukurta programos dalis administratoriui, kurios pagalba jam yra (žr. 20 pav.):

1. surenkami verslo valdymo programos visi laukai kurių klasė yra viena iš: TDBEdit, TDBMaskEdit, TDBComboBox. Dėl šių įrankių susiejus DataSource (skirta nuskaityti duomenis ir perduoti į lentelę) ir DataField su atitinkamais komponentais, ir pasirinkus įrašą lentelėje, jis bus atvaizduotas tuose komponentuose automatiškai (be papildomo kodo rašymo).
2. šie laukai surašomi į sąrašą, ir pateikiami rolių redagavimo lange;
3. kiekvienam laukui administratorius gali parinkti vieną iš leidimų: nerodyti, rodyti neleidžiant redaguoti, leisti redaguoti;
4. parinkus leidimus jie išsaugojami duomenų bazėje. Joje saugomi tik „rodyti neleidžiant redaguoti“ (skaityti) ir „leisti redaguoti“ (rašyti) leidimai; standartiškai programa paslepia visus laukus, tad nėra tikslo duomenų bazėje saugoti dar ir „nerodyti“ leidimą.



20 pav. Metodo veikimas administratoriaus dalyje.

Leidimų nustatymo laukams principas pavaizduotas lentelėje (žr. 3 lent.). Pavaizduota kaip pagal teises yra įjungiami programos laukai ir kaip visa tai „supranta“ duomenų bazė.

3 lentelė

Teisės	**TDBEdit**	**Uzsak_nr**
Jokia	TDBEdit->Visible = false;	REVOKE UPDATE, SELECT ON prekes(pavadinimas) TO ROLE vartotojai;
Tik skaitymo	TDBEdit->Visible = true; TDBEdit->Enabled = false;	REVOKE UPDATE ON prekes(pavadinimas) TO ROLE vartotojai; GRANT SELECT ON prekes(pavadinimas) TO ROLE vartotojai;
Skaitymo ir rašymo	TDBEdit->Visible = true; TDBEdit->Enabled = true;	GRANT UPDATE, SELECT ON prekes(pavadinimas) TO ROLE vartotojai;

3.4.2 Laukų surinkimo formoje principas

Į programą yra įdedamas žemiau pateiktas kodo fragmentas, kuris eina per visas programos formas ir ieško ar yra TEdit, TButton, TMaskEdit, TComboBox tipo elementų, jei yra – tada yra tie laukai pateikiami administratoriaus leidimų redagavimo sąraše.

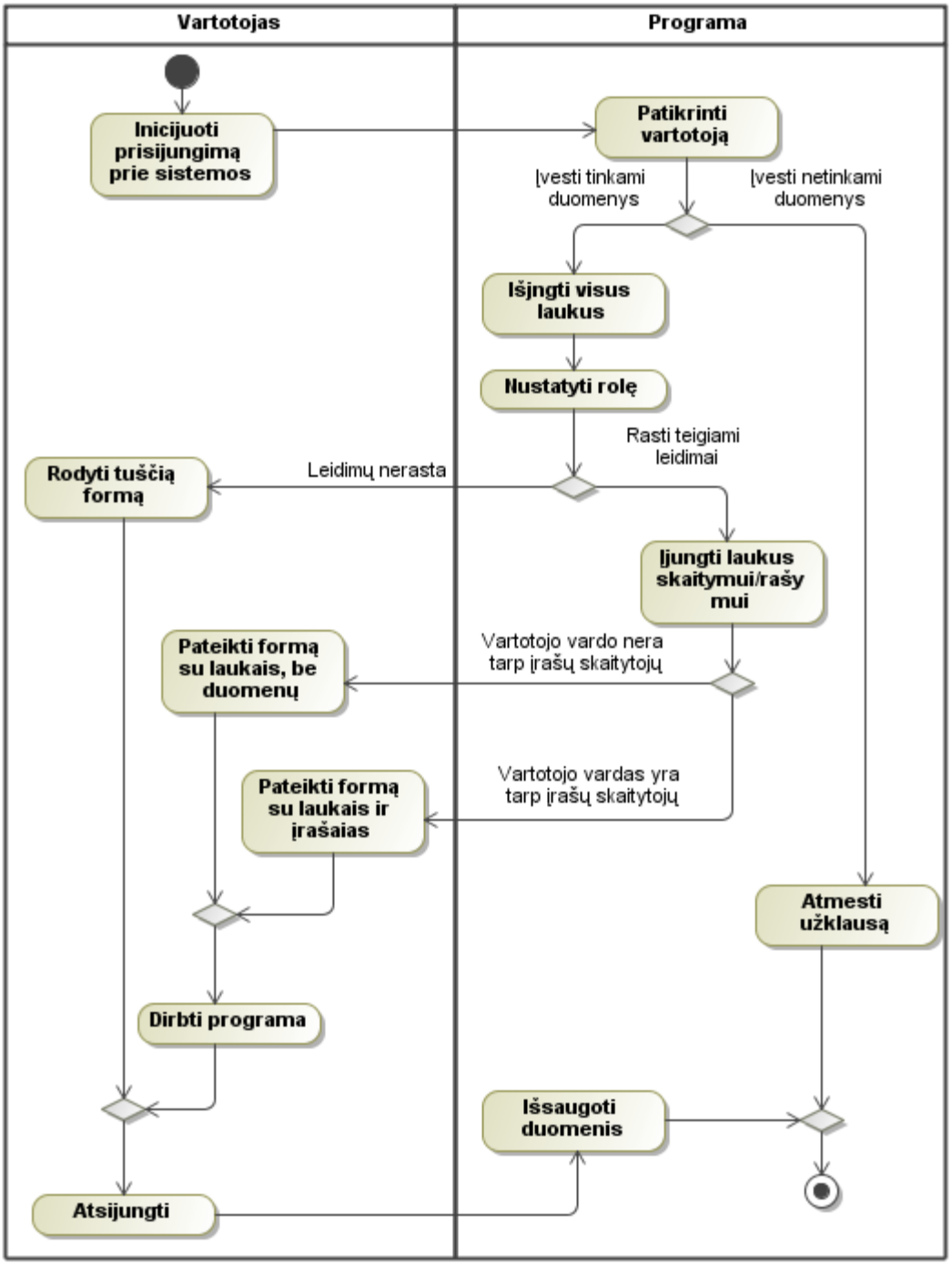
```
for (int i=0; i<Form->ComponentCount; i++){
  if (Form->Component[i] is in [TEdit, TButton, TMaskEdit, TComboBox])
    <veiksmai>
}
```

3.4.3 Leidimų veikimo principas (vartotojas)

Prisijungusiam, autentifikuotam vartotojui kiekvienas interaktyvus formų komponentas paslepiamas ir tik tada nuskaitoma jam priklausanti rolė iš duomenų bazės (žr. 21 pav.). Pagal rolę nustatomi leidimai visų formų kiekvienam interaktyviam komponentui (formai, mygtukui, laukui ir įrašui). Duomenų bazėje yra saugomi tik tie leidimai, kurie suteikia tam tikram interaktyviam komponentui skaitymo arba koregavimo funkcijas (teigiami leidimai). Jei tokių leidimų suteikta nebuvo, t.y. pagal nustatytą rolę duomenų bazėje leidimu nerasta – tada vartotojui bus rodomas tuščias programos langas. Jei visgi leidimų buvo, tai pagal juos yra įjungiami interaktyvūs laukai.

Po to, kai yra įjungiami matomi ir/ar redaguojami laukai, yra žiūrima į laukų lenteles, kuriose yra parašyta kokius būtent įrašus gali matyti prisijungęs vartotojas. Pavyzdžiui, jei vartotojui yra suteikiamas leidimas koreguoti kliento vardą, tai yra kreipiamasi į visų klientų lentelę „Klientai“ ir žiūrima prie kokių įrašų „skaitytojo“ skiltyje yra prirašytas jo vartotojo vardas.

Visi programoje daromi pakeitimai iškart matomi tame pačiame lange esančiame sąraše. Pavyzdžiui, pakeitus pirkėjo adresą, pakeitimas iškart atvaizduojamas pirkėjų sąraše.



21 pav. Metodo veikimas vartotojo dalyje.

3.4.4 Teisių valdymas programos vartotojams

1. Sukuriamos rolės

   INSERT INTO roles VALUES (id,'pavadinimas');

2. Rolėms priskiriamos/redaguojamos/naikinamos teisės komponentams
   INSERT INTO leidimai VALUES (id,role,'forma->komponentas','leidimas');
   UPDATE leidimai SET leidimas='leidimas' WHERE id=id
   DELETE FROM leidimai WHERE id=id
3. Vartotojams priskiriamos rolės
   UPDATE vartotojai SET role='role';

3.4.5 Duomenų nuskaitymas iš duomenų bazės

Kadangi programa turi priėjimą prie visų duomenų, ji pasirūpina kuriuos duomenis atvaizduoti, o kuriuos – ne. Tai daroma paslepiant/rodant atitinkamus komponentus. Todėl programai keliami du reikalavimai:

1. Programoje naudojami komponentai turi turėti žmonėms pritaikytus komponentų vardus, kad administratorius žinotų kokiam būtent komponentui jis redaguoja leidimą.
2. Administratorius turi turėti bent menkiausią supratimą, kaip veikia programa.

3.4.6 Duomenų išsaugojimas į duomenų bazę

Standartiškai, programoje išsaugojus atitinkamo įrašo pakeitimus, išsaugomi visi įrašai, nepriklausomai nuo leidimo, tačiau, jei vartotojas neturėjo leidimo keisti atitinkamą lauką, jis išsaugomas toks koks buvo, t.y. be pakeitimų. Pavyzdžiui, jei vartotojui neleidžiama redaguoti kliento adreso, tai kai vartotojas įterpia naują vartotoją, tai adreso laukeliai turėtų būti išvalomi, t.y. nustatoma numatytoji reikšmė (gali būti koks skaičius, tai galima būtų įrašyti 0).

3.4.7 Naujų duomenų pridėjimas į duomenų bazę

Kadangi nauji duomenys išsaugomi atitinkamai kaip ir seni (t.y. redaguojami), tai išsaugomi ir tie laukai, kurių vartotojas nemato/negali redaguoti. Tokiu atveju reikia nurodyti numatytas reikšmes programiškai.

3.4.8 Auditas

Audito metodo kodas, įdėtas į pagrindinę programos formą, įrašo duomenis į duomenų bazę:

```
IBSQL1->SQL->Add("INSERT INTO auditas VALUES
('"+id+"','"+laikas+"','"+vartotojas+"','"+pakeitimas+"');");
```

Kita pusė kodo yra įdedama į kiekvienos formos mygtukų metodus, kuriuos nuspaudus duomenys yra pakeičiami ar ištrinami. Tokiu atveju automatiškai nustatomas ID, laikas kada tai buvo padaryta, vartotojo vardas, kuris paspaudė mygtuką ir pats pakeitimas (jei tuo metu buvo pakeista parametro reikšmė į naują – tai yra išsaugojama sena ir nauja reikšmė). Pakeitimą sudaro priklausomai nuo to ką norima audituoti (kiekviename mygtuko metode yra nurodomas skirtingas parametras). Taigi vartotojui paspaudus mygtuką visa informacija yra siunčiama į audito metodą, esantį pagrindinėje programos formoje, kuris ir įrašo informaciją į duomenų bazę.

Mygtuke esančio audito kodo dalis gali atrodyti taip:

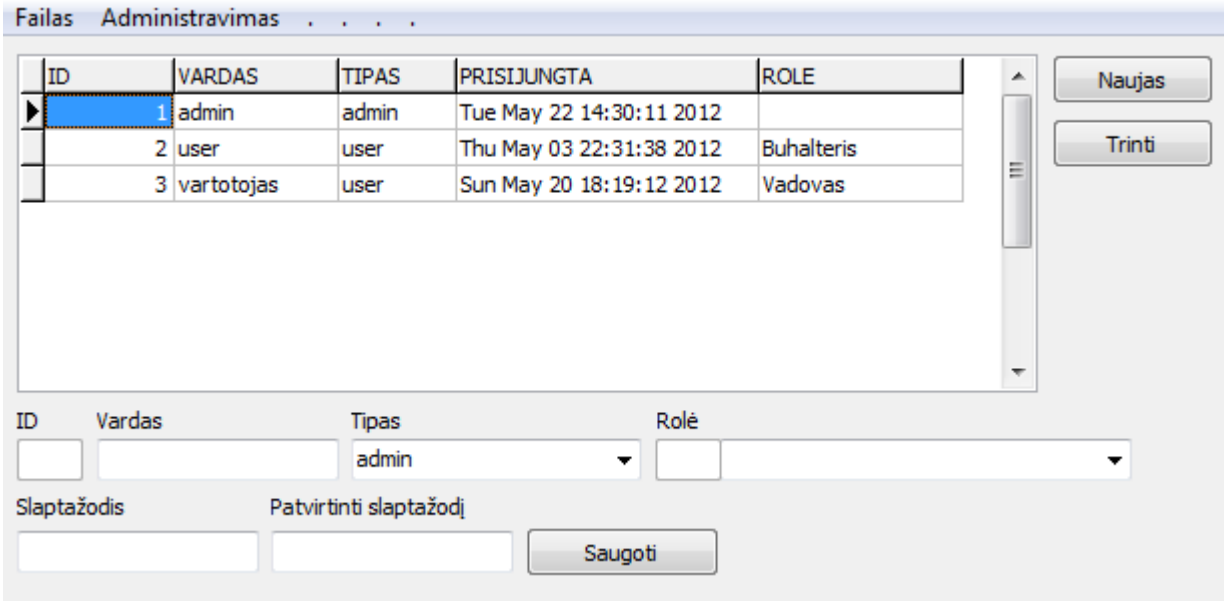
```
Form1->Auditas("Ištrintas užsakymas nr. "+id);
```

4. EKSPERIMENTINĖ REALIZACIJA

Eksperimentinei realizacijai panaudota maža, bet pakankama verslo programos dalis, kuri galėtų parodyti metodo praktinį veikimą. Ji susideda iš vartotojų administravimo, klientų, užsakymų, prekių ir audito meniu punktų.

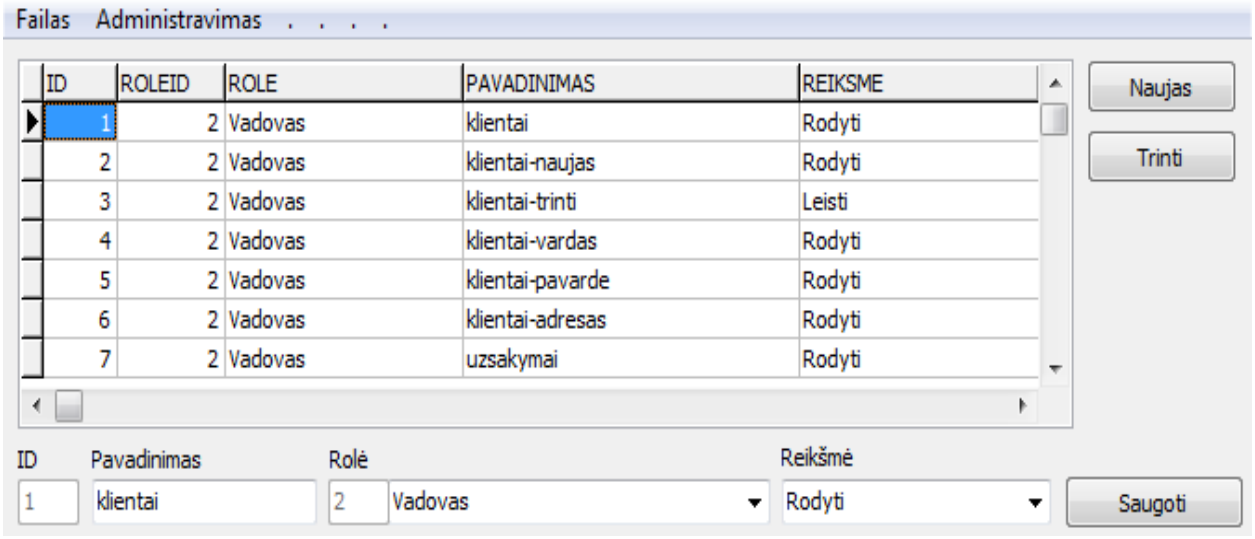
4.1 Administravimas

Administravimą sudaro vartotojų priskirimo rolėms langas (žr. 22 pav.) bei rolėms priskirtų leidimų langas (žr. 23 pav.). Šį meniu punktą gali matyti tik administratoriaus teises turintis vartotojas. Pirmajame yra išvardinti vartotojų prisijungimo vardai su jiems susietomis rolėmis, bei paskutinės prisijungimo dienos data, taip pat yra galimybė sukurti naują vartotoją, koreguoti senąjį.



22 pav. Vartotojų priskirimas rolėms langas.

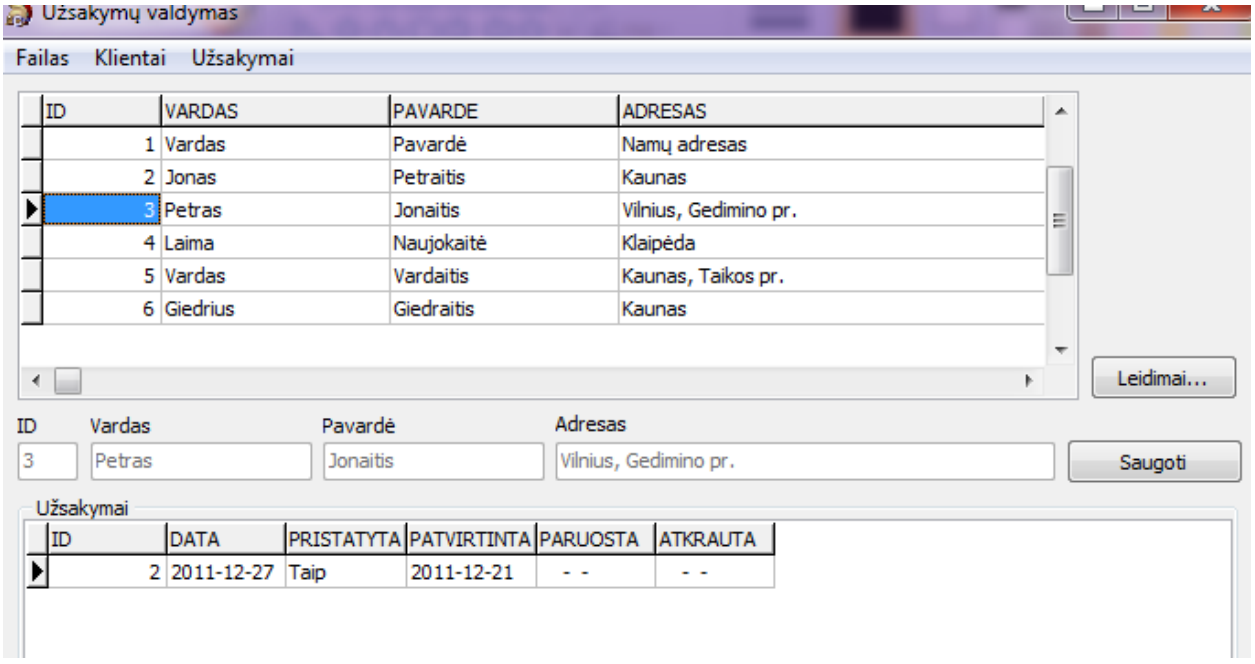
Rolėms priskirtų leidimų lange yra pasirenkama rolė, ir jai pagal sąraše esančius komponentus nustatomi leidimai. Kai administruojama didelė verslo programa, kur yra nemažai komponentų, tai pavadinimai turėtų būti tokie, kad administratorius galėtų lengvai atpažinti koks pavadinimas kokį komponentą atitinka (pvz, mygtukas_klientai_naujas).



23 pav. Leidimų priskyrimas rolėms langas.

4.2 Klientų langas

Klientų lange yra išvardinta asmeninė informacija apie klientus ir, jei yra užsakymas, jų užsakymo pagrindiniai duomenys (žr. 24 pav.). Ši asmeninė informacija yra naudojama užsakymams priimti.

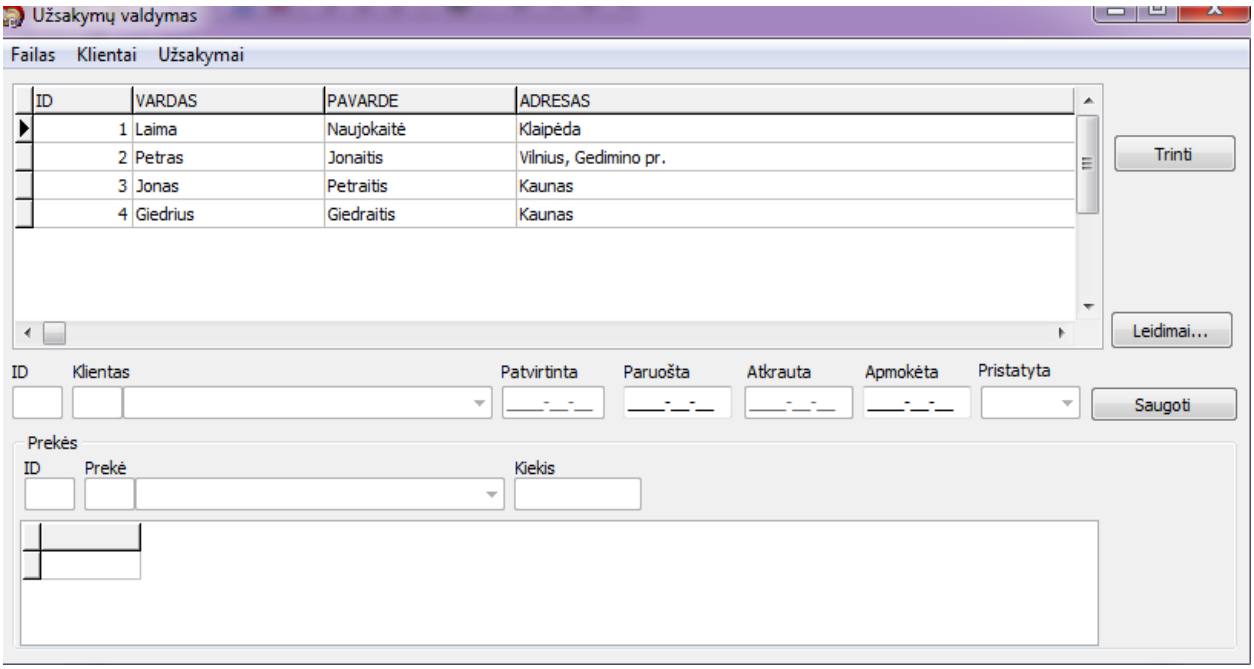


24 pav. Informacijos apie klientus langas

4.3 Užsakymo langas

Šiame lange yra pateikti esami detalūs užsakymai, pasirinkus klientą rodomas jo atitinkamas užsakymas (žr. 19 pav.). Pritaikius roles, šia forma gali naudotis kelių skirtingų profesijų darbuotojai, pavyzdžiui užsakymo priėmėjui (užpildant kliento užsakymo formą), gali būti rodomi tik užsakymui priimti būtini laukai, taip pat atsakingam už pakrovimą ir to patvirtinimą sandėlininkui (jam visiškai nereikalingi laukai tokie kaip klientas, adresas ir t.t. gali būti išjungti), vairuotojui, kuris nuveža krovinį ir pažymi formoje „pristatyta“, jam nėra reikalo suteikti teisę redaguoti užsakymo sudėtį.

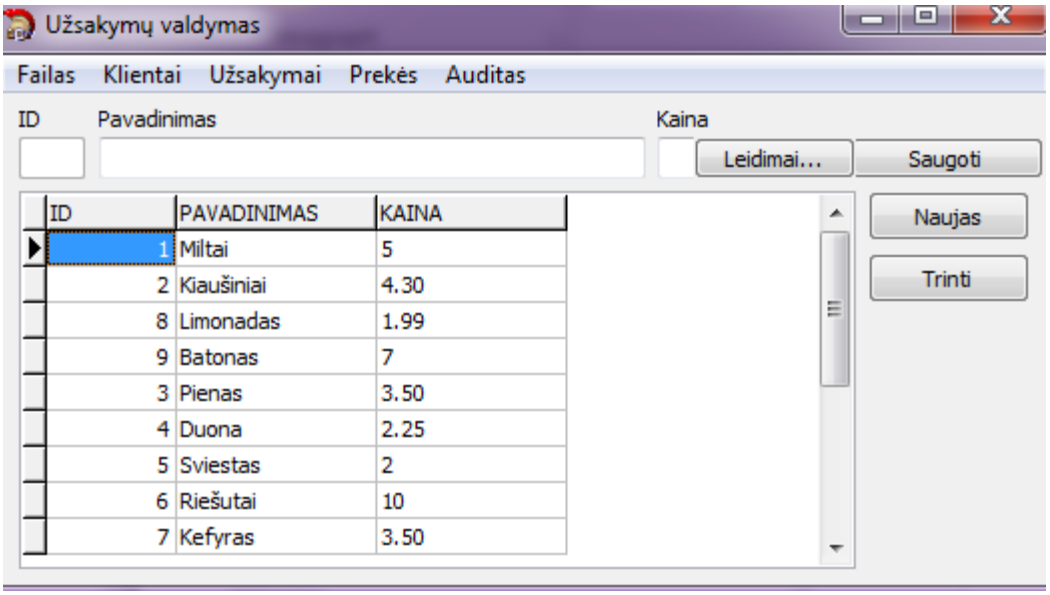
Pavyzdyje galima matyti kaip veikia rolės (žr. 25 pav.), leidimai čia sujunginėti be jokios logikos, tiesiog parodyti, kad yra galimybė valdyti kiekvieną komponentą autonomiškai, bei prisijungusiam vartotojui yra įjungti keturi įrašai.



25 pav. Klientų užsakymo sąrašai.

4.4 Prekių langas

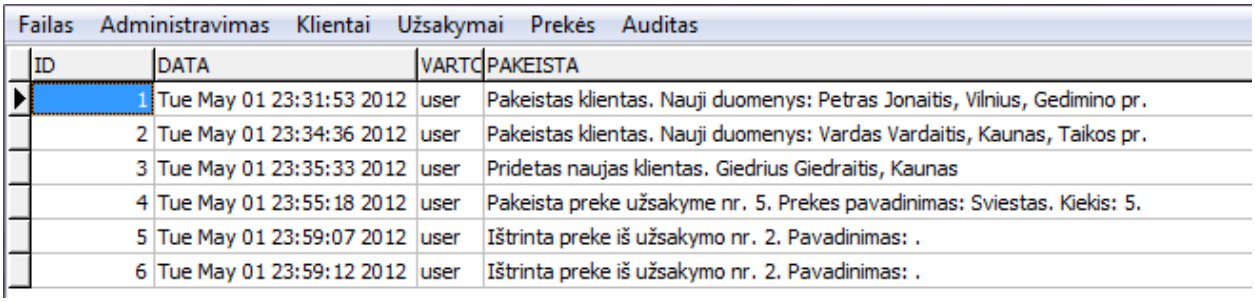
Prekių lange yra prekių sąrašas, kurios reikalingos klientų užsakymams priimti (žr. 26 pav.). Šios rolės vartotojui su prekėmis yra leidžiama darytį viską (sujungti visi laukai ir rodomos visų prekių įrašai).



26 pav. Inormacija apie prekes.

4.5 Audito langas

Audito lange yra rašoma informacija apie kintančius duomenis duomenų bazėje (žr. 27 pav.). Čia rašoma koks vartotojas, ką būtent atliko ir kada.



27 pav. Audito kontrolės langas.

5. EKSPERIMENTO REZULTATAI

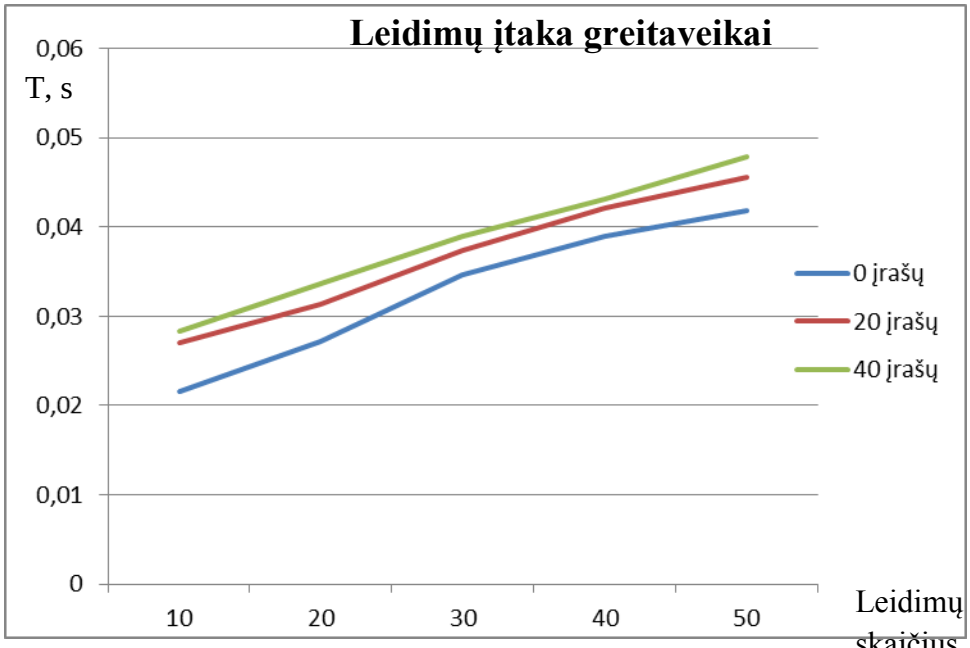
5.1 Eksperimento metu gauti rezultatai

Ištirta maža verslo valdymo programos dalies greitaveika, kurią įtakoja skirtingų parametrų skaičius, kadangi skaičiai tokie maži ir labai „jautrūs“ pašalinei kompiuterio apkrovai, tai buvo matuojama po penkis kartus ir vedamas vidurkis.

Pirmu atveju ištirta kaip greitaveiką veikia vartotojo turimų leidimų kiekis ir su tais leidimais susijusių matomų įrašų kiekis (žr. 28 pav.). Kadangi, kai vartotojui nėra priskiriamas nei vienas leidimas, tada ir metodas netikrina ar yra įjungti įrašai, tai šiuo atveju nebuvo matuota greitaveika kai vartotojas neturi leidimų (žr. 3 lent.).

3 lentelė

Leidimų skaičius, vnt.	Kai rodymui įjungta 0 įrašų, s	Kai rodymui įjungta 20 įrašų, s	Kai rodymui įjungta 40 įrašų, s
10	0,0216	0,0271	0,0283
20	0,0272	0,0313	0,0337
30	0,0347	0,0374	0,0389
40	0,039	0,0421	0,0431
50	0,0419	0,0456	0,0478



28 pav. Leidimų priklausomybė nuo įrašų skaičiaus.

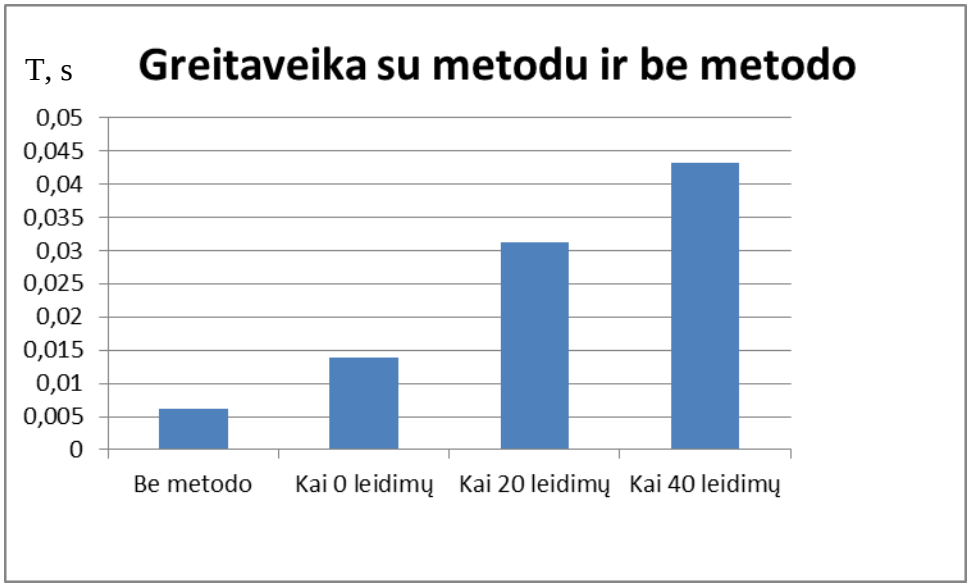
Išanalizavę grafiką (žr. 28 pav.) matome, kad atsako laikas didėja, kai leidimų skaičius ir įrašų skaičius taip pat didėja, tai yra todėl, kad kuo daugiau leidimų, įrašų programa turi nuskanuoti ir įjungti, tuo ilgiau užtrunka laiko tai darydama. Priklausomybė tarp laiko ir leidimų skaičiaus yra beveik tiesinė.

Padidinus matomų įrašų skaičių nuo 0 iki 20 gaunamas vidutinis 0,00382 sekundžių vėlinimas arba tai būtų 1,13 karto, o kai įrašų skaičius yra nuo 20 iki 40, tai vidutinis vėlinimas yra 0,00166 sekundės arba 1,0466 karto. Iš to galime teigti, kad metodui vis mažiau reikia užtrukti laiko dar tokiam pat kiekiui įrašų įjungimo rodymui (atsiima metodo iškvietimui laikas, nes nereikia jo kviesti antrą kartą).

Antru atveju išmatuota greitaveika, kai prisijungęs prie verslo valdymo programos vartotojas neturi jokių leidimų, turi 20 ir turi 40, bei sprendimas kai išvis nėra įdiegto prieigos teisių valdymo modulio (žr. 4 lent., 29 pav.).

4 lentelė

Be metodo, s	Kai 0 leidimų, s	Kai 20 leidimų, s	Kai 40 leidimų, s
0,0062	0,0139	0,0313	0,0431



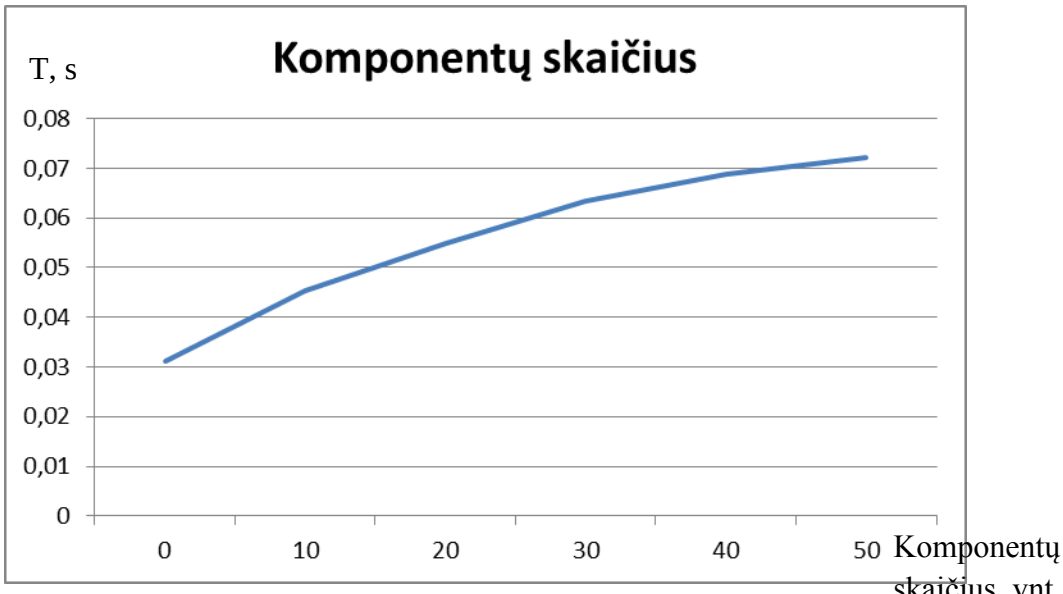
29 pav. Greičio priklausomybė su metodu ir be metodo.

Kaip matome iš histogramos (žr. 29 pav.) programos našumas didžiausias, kai verslo valdymo programoje nėra įdiegto prieigos teisių valdymo, o kai metodas yra, tai našumas atitinkamai mažėja.

Trečiame atvejyje ištirtas greitaveikos kitimas priklausomai nuo administratoriui į sąrašą (leidimams sudarinėti) pateikiamų komponentų skaičiaus (žr. 5 lent., 30 pav.).

5 lentelė

Komponentų skaičius, vnt.	Greitaveika, s
0	0,0312
10	0,0453
20	0,0549
30	0,0635
40	0,0687
50	0,0721



30 pav. Komponentų priklausomybės nuo laiko grafikas.

Ištyrus komponentų skaičiaus grafiką (žr. 29 pav.), pastebėta, kad kai komponentų skaičius didėja, tai laikas, kuris skirtas juos nuskanuoti ir pateikti „vartojimui“ taip pat didėja. Iš grafiko matosi, kad kreivė po truputį pradeda „lenktis“, tai yra vis mažiau laiko užtrunkama, kai reikia pateikti į sąrašą tokiam pačiam kiekiui interaktyvių komponentų (žr. 6 lent.).

6 lentelė

	Nuo 0 iki 10	Nuo 10 iki 20	Nuo 20 iki 30	Nuo 30 iki 40	Nuo 40 iki 50
Reikia laiko, s	0,0141	0,0096	0,0086	0,0052	0,0034

5.2 Eksperimento išvados

Eksperimento rezultatas yra realizuoto metodo įvertinimas greitaveikos požiūriu, ištirta prieigos teisių valdymo metodo greitaveikos priklausomybės nuo komponentų leidimų kiekio (su skirtingų įrašų kiekiu). Greitaveika ištirta kai duomenų bazė su informacija yra patalpina lokaliai.

Iš gautų rezultatų galime teigti, kad administratoriui skirtos programos našumas yra lėtesnis negu vartotojų programos, nes metodas skanuoja visas verslo valdymo programos formas ir ieško interaktyvių komponentų, kai vartotojui metodas įjungia tik tuos laukus, įrašus, kurie yra išsaugoti duomenų bazėje.

Iš antro eksperimento metu atliko atvejo matosi, kad šis prieigos teisių valdymo modelio sprendimas sulėtina verslo valdymo programą 2,2 karto.

6. IŠVADOS

Sukūrus ir įgyvendinus pasiūlytą verslo valdymo programų prieigos teisių valdymo modelį prieita prie išvadų.

1. Literatūros šaltinių ir mažų bei vidutinių įmonių poreikių didėjimo analizė parodė, kad prieigos teisių valdymo problema aktuali nuolat atnaujinant verslo valdymo programą, atsirandant naujiems poreikiams, todėl tokioms programoms vartotojo teisių valdymo modulis turi būti prisitaikantis prie naujų modulių prijungimo ir naujų vartotojų atsiradimo įmonėje.

2. Prieigos saugos modelių analizė parodė, kad labiausiai įmonės poreikius atitinkantis bei pagrindinę įmonės saugos politiką atitinkantis modelis yra rolėmis paremtas prieigos teisių valdymas. Jis realizuotas su griežtu bei labai lanksčiu prieigos teisių valdymu.

3. Pagal nustatytus funkcinius ir nefunkcinius reikalavimus buvo sukurtas rolėmis paremto prieigos valdymo metodas, kuris automatiškai prisitaiko prie naujų verslo programos formų pakitimų ir nuskanavęs interaktyvius komponentus pateikia juos administravimui (leidimų suteikimui). Siekiant kuo griežčiau apriboti prieigą buvo pridėta galimybė riboti prieigą įrašų lygyje, o, kad autorizuoti vartotojai jaustų atsakomybę už savo veiksmus - buvo pridėta auditavimo funkcija.

4. Metodo universalus pritaikymas verslo valdymo programoms parodė, kad prieigos teises galima realizuoti praktinėse verslo valdymo programose turint minimalų programavimo žinių lygį bei bent minimaliai suprantant kaip veikia programa.

5. Eksperimento metu buvo ištirta greitaveikos priklausomybė nuo įvairių programos pokyčių ir gauti tokie pastebėjimai:

- didėjant leidimų ir įrašų skaičiui programos našumas laipsniškai mažėja beveik tiesiškai;
- šis prieigos teisių valdymo modelio sprendimas sulėtina verslo valdymo programą 2,2 karto.
- didėjant interaktyvių komponentų skaičiui laikas, kuris skirtas juos nuskanuoti ir pateikti „vartojimui“, taip pat didėja, bet ne taip žymiai.

LITERATŪRA

[1] R. Needham, R. Maybury. Security Engineering: A Guide to Building Dependable Distributed Systems. 51-71 p.2010

[2] Vinay Purohit. Authentication and Access Control- The Cornerstone of Information Security. 2007.

[3] S. Smalley. Which operating system access control technique will provide the greatest overall benefit to users? 147 p. New York, NY, USA © 2001

[4] Vincent C. Hu, David F. Ferraiolo, D. Rick Kuhn. Assessment of Access Control Systems. National Institute of Standards and Technology 2006-09.

[5] System Security, John Mitchell. Access Control and Operating.

[6] Ryan Ausanka-Crues. Methods for Access Control: Advances and Limitations. Claremont, California 2005.

[7] Vinith Bindiganavale and Dr. Jinsong Ouyang. Role Based Access Control in Enterprise Application – Security Administration and User Management, IEEE.

[8] Ravi S. Sandhu, Edvard J. Coyone. Role-Based Access Control Models. SETA Corporation.

[9] Hazen A. Weber. Role-Based Access Control: The NIST Solution. SANS Institute InfoSec Reading Room. October 8, 2003

[10] Ravi Sandhu, David Ferraioloy and Richard Kuhny. The NIST Model for Role-Based Access Control: Towards A Unied Standard.

[11] Andrey D. Petrov, Suzanne Gysin, and Carl Schumann. User autthentification for Role-Based Access Control. Fermi National Accelerator Laboratory, Batavia, IL 60510, U.S.A 2008

[12] Elisa Bertino. RBAC models – concepts and trends. University of Milano

[13] Axel Kern. Advanced Features for Enterprise-Wide Role-Based Access Control. Koln, Germany 2002.

[14] Srinivasan Vanamali. Role Engineering: The Cornerstone of Role-Based Access Control. CISA, CISSP. 2009-12.

[15] Jan Kasprzak, Michal Brandejs, Matej Cuhel, Tomáš Obšívac. Access Rights in Enterprise Full-text Search. Faculty of Informatics Masaryk University Brno.2010-06.

[16] David Ferraiolo, D. Richard Kuhn, Ramaswamy Chandramouli. Role-based access control. 1-23 p.

[17] Rick Kooker, Stephan Kane. Identity Management: Role Based Access Control for Enterprise Services.

[18] Amir H. Chinaei, Ken Barker, Frank Wm. Tompa. Comparison of Access Control Administration Models. Special Issue on ICIT 2009 Conference - Web and Agent Systems.

[19] Sejong Oh, Seog Park. Task–role-based access control model. Department of Computer Science, Sogang University, 121-742 Seoul, South Korea 2002-05.

[20] Sejong Oh. New role-based access control in ubiquitous e-business environment.

[21] Geoff Worboys. Firebird File and Metadata Security.

[22] Jyotindra Zaveri. Enterprise Resource Planning. Himalaya Publishing House, 2010

[23] Chee Wee Tan, Shan Ling Pan. ERP success: The Search For A Comprehensive Framework. Information Systems Journal (2002)

[24] Moutaz Haddara, Ondrej Zach.ERP Systems in SMEs: A Literature Review. Kauai, Hawaii USA January 04-January 07 ISBN: 978-0-7695-4282-9.

[25] Client-Server Architecture. Computer Science Program, The University of Texas, Dallas [interaktyvus]. [Žiūrėta 2012-]. Prieiga per internetą: <http://www.utdallas.edu/~chung/SA/2client.pdf>

TERMINŲ IR SANTRUMPŲ ŽODYNAS

ERP (angl. Enterprise Resource Planning) – verslo valdymo programa.

DBMS (angl. Database management system) – duomenų bazės valdymo sistema (**DBVS**).

DB (angl. Database) – duomenų bazė.

DAC (angl. Discretionary access control) – diskretiškas prieigos valdymo modelis.

MAC (angl. Mandatory access control) – privalomasis prieigos valdymo modelis

RBAC (Role Based Access control) – rolėmis paremtas prieigos valdymo modelis.

MLS (angl. Multi-Layer Secure) – daugiasluoksnės saugos.

ACL (angl. Access Control List) – prieigos teisių sąrašai.

SSD (angl. Static Separation of Duty) – statiniu teisių atskyrimas.

DSD (angl. Dinamic Separation of Duty) – dinaminiu teisių atskyrimas.

SQL (angl. Structured Query Language) – struktūrinių užklausų kalbą.

P2P (angl. Point to point) – taško į tašką architektūra.