KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PROGRAMŲ INŽINERIJOS KATEDRA

Tomas Žvirgžda

Bankinės sistemos komunalinių įmokų modulio sudarymas ir tyrimas

Magistro darbas

Darbo vadovas
dr. S. Vaičiulis

Kaunas, 2011

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PROGRAMŲ INŽINERIJOS KATEDRA

Tomas Žvirgžda

Bankinės sistemos komunalinių įmokų modulio sudarymas ir tyrimas

Magistro darbas

Recenzentas
A. Lenkevičius
2011-05

Vadovas
dr. S. Vaičiulis
2011-05-25

Atliko
IFM-9/2 gr. stud.
Tomas Žvirgžda
2011-05-25

Kaunas, 2011

Turinys

1 ĮVADAS 6
1.1 Dokumento paskirtis 6
1.2 Santrauka 6
2 ANALITINĖ DALIS 7
2.1 Literatūros apžvalga 7
2.1.1 Temos aktualumas 7
2.1.2 Tikslai 8
2.1.3 Egzistuojantys sprendimai 8
2.1.4 Architektūra 10
2.1.5 Programavimo kalba 10
2.1.6 Operacinė sistema 10
2.1.7 Duomenų bazė 11
2.2 Veiklos sudėtis (funkciniai reikalavimai) 11
2.2.1 Veiklos kontekstas (pateikiama konteksto diagrama) 11
2.2.2 Veiklos padalinimas 11
2.2.3 Sistemos sudėtis 12
2.2.4 Panaudojimo atvejų sąrašas 13
2.2.5 Reikalavimai duomenims 19
2.3 Nefunkciniai reikalavimai 21
2.3.1 Reikalavimai sistemos išvaizdai 21
2.3.2 Reikalavimai panaudojamumui 21
2.3.3 Reikalavimai vykdymo charakteristikoms 22
2.3.4 Reikalavimai veikimo sąlygoms 22
2.3.5 Reikalavimai sistemos priežiūrai 22
2.3.6 Reikalavimai saugumui 22
2.3.7 Kultūriniai ir politiniai reikalavimai 23
2.3.8 Teisiniai reikalavimai 23
3 PROJEKTINĖ DALIS 24
3.1 Architektūros tikslai ir apribojimai 24
3.2 Sistemos statinis vaizdas 24
3.2.1 Apžvalga 24
3.2.2 Paketų detalizavimas 24

3.3 Sistemos dinaminis vaizdas ... 28
3.3.1 Būsenos diagrama ... 28
3.3.2 Veiklos diagrama ... 29
3.4 Išdėstymo vaizdas ... 35
3.5 Duomenų vaizdas ... 36
4 TYRIMO DALIS ... 38
4.1 Liktinės ir naujos sistemos palyginimas ... 38
4.2 Kokybės vertinimas ... 41
4.2.1 Apklausų anketos ... 41
4.3 Programos išeities kodo metrikų analizė ... 42
4.4 Siūlomi tolimesni programos tobulinimai ... 44
4.4.1 Nauji funkcionalumai ... 44
4.4.2 Grafinė sąsaja ... 44
4.4.3 Nefunkciniai reikalavimai ... 45
5 IŠVADOS ... 46
6 LITERATŪRA ... 47
7 TERMINŲ IR SANTRUMPŲ ŽODYNAS ... 49
8 PRIEDAI ... 50

Summary

Objective of the project is to build a model of banking system utility payments module to the Lithuanian Central Credit Union (hereinafter referred to CKU). This module is intended to help customers to pay for their use of public utility services, data capture, and data distribution to companies for which customers accounted. This entire module is done centrally. Since CKU provide services to other unions, which means that CKU accepts full responsibility for the system works. The module acts as a center, which collects all data from all credit union clients of over Lithuania and distributes data to all public utility services. This project is unique in the fact that such a system in Lithuania is not like the other Baltic States (Lithuania neighboring) countries. The project has the advantage that it can be easily adapted to another country or the world market. During the quality analysis of the system, it was evaluated as a good system.

PAVEIKSLIUKŲ SĄRAŠAS

1 pav. Bankinės sistemos komunalinių įmokų modulio ir aplinkos kontekstas.

2 pav. Panaudojimo atvejų diagrama.

3 pav. Klasių diagrama

4 pav. Sistemos paketų aukščiausias lygis.

5 pav. BS branduolys.

6 pav. SelfService paketas.

7 pav. EbankWeb paketas.

8 pav. BS branduolio klasių diagrama.

9 pav. Kontrakto būsenos diagrama.

10 pav. Įmokos būsenos diagrama.

11 pav. Failo sugeneravimo būsenos diagrama.

12 pav. Registruoti kontraktus.

13 pav. Išsaugoti naują klientinę (KU nario) įmoką.

14 pav. Išsaugoti KU darbuotojų suvestą naują įmoką.

15 pav. Išsiųsti sugeneruotus įmokų failus paslaugų teikėjams.

16 pav. Pateikti operacijų išrašą.

17 pav. Pateikti įmokų archyvą.

18 pav. Pateikti įmokų ataskaitas.

19 pav. Atlikti darbuotojų suvestų įmokų pinigų nurašymą.

20 pav. Pateikti nurašytų įmokų ataskaitą.

21 pav. Atlikti komisinių paskirstymą į unijų sąskaitas.

22 pav. Pateikti komisinių ataskaitą.

23 pav. Keisti įmokos būseną.

24 pav. Bendras sistemos ir kliento/darbuotojo išdėstymo vaizdas.

25 pav. „i-Unija“ ir „i-Kubas“ veikimo ir pasiekiamumo vaizdas.

26 pav. „KUBAS“ veikimo ir pasiekiamumo vaizdas.

27 pav. Komunalinių įmokų modulio SourceMonitor metrikų santrumpos langas.

LENTELIŲ SĄRAŠAS

1 lentelė. Veiklos įvykių sąrašas.

2 lentelė. BS serverio IBM System x3500 M2 specifikacija [3].

3 lentelė. Analizės.

4 lentelė. Projektavimas.

5 lentelė. Įgyvendinimas.

6 lentelė. Testavimas.

7 lentelė. Įdiegimas.

8 lentelė. Modulio metrikų duomenys.

9 lentelė. Ciklomatinio sudėtingumo reikšmės vertinimas.

1 ĮVADAS

1.1 Dokumento paskirtis

Šiame dokumente nagrinėjama tema yra susijusi su informacinėmis bankinėmis sistemomis, o tiksliau su viena iš sudedamųjų dalių – komunalinių paslaugų įmokų atsiskaitymu informacinėje bankinėje sistemoje. Analitinėje dalyje yra išsamiau supažindinama su problemine sritimi. Supažindinama su panaudotais sprendimais. Projektinėje dalyje yra pateikti esminiai aspektai susiją su projekto architektūra, sprendimų įgyvendinimu. Tiriamoji ir eksperimentinė dalis pristato sukurtos programinės įrangos kokybės analizę. Palyginama lyginamuoju principu sukurta ir liktinė programos. Parodomos silpnosios ir stipriosios vietos abiejų programų.

1.2 Santrauka

Projekto tikslas – bankinės sistemos komunalinių įmokų modulio sudarymas Lietuvos centrinei kredito unijai (toliau CKU). Tai modulis skirtas atsiskaityti klientams už jiems suteiktas komunalines paslaugas, duomenų užfiksavimą, bei jų paskirstymą visoms įmonėms už kurias klientas atsiskaitė. Visas šis modulis yra daromas centralizuotai. Kadangi CKU teikia paslaugas kitoms unijoms, tai reiškia jog CKU priima visą atsakomybę už sistemos darbus. Sukurtas modulis veikia kaip centras, kuris surenka visus duomenis iš visos Lietuvos unijų ir klientų bei paskirsto juos įmonėms.

Šis modulis yra svarbus CKU, kaip žingsnis siekiant patenkinti klientų ir kredito unijų (toliau KU) poreikius, bei konkurencingumo sustiprinimas rinkoje. Šis modulis sustiprina KU pozicijas tiek rinkoje tiek paslaugų teikimo pasiūloje.

Šis projektas yra išskirtinis tuo, jog tokios sistemos Lietuvoje dar nėra kaip ir kitose pabaltijos (Lietuvos kaimynės) šalyse. Projekto privalumas yra tas jog jis gali būti lengvai pritaikytas kitos šalies ar pasaulio rinkai.

Toliau šioje ataskaitoje yra atliktas kokybės tyrimas bei lyginamuoju principu palyginta liktinė sistema ir nauja sistema su sukurtu komunalinių įmokų moduliu.

2 ANALITINĖ DALIS

2.1 Literatūros apžvalga

2.1.1 Temos aktualumas

Elektroninė bankininkystė yra plačiai paplitusi visame pasaulyje. Kiekvienas didesnis bankas ar kredito unija ar unijų grupė ar kita įmonė teikianti finansines paslaugas [1] turi savo bankinę sistemą. Dabar pervesti pinigus iš vienos sąskaitos į kitą (iš vieno banko sąskaitos į kito banko ar kredito unijos sąskaitą) užima kelias sekundes kliento brangaus laiko, o viduje operacijos atliekamos akimirksniu, nebent būna kokių apribojimų: savaitgaliais pinigai iš vieno banko sąskaitos į kito banko sąskaitą nepervedami, įstaiga nurodo kažkokius laiko limitus, per kurį pavedimas (čia pavedimas - pinigų pervedimu iš vienos sąskaitos į kitą) gali būti dar atšaukiamas, kitaip sakant pavedimas yra vykdymo stadijoje. Dažniausiai apribojimai būna susieti laiku ar/ir pinigais (pvz.: už skubų pavedimą yra papildomas mokestis). Šiais laikais beveik visos firmos, įstaigos, organizacijos ar žmogus turi savo sąskaitą (-ų) viename ar keliuose bankuose/unijose. Elektroninės bankininkystės pagrindiniai privalumai yra **greitis**, **kokybė**, **saugumas** ir **patogumas**. Visi išvardinti pagrindiniai privalumai reiškia **laimingas klientas.**

Finansinės įstaigos [1] turėdamos gerą infrastruktūrą vietovėje (miestuose, regionuose, šalyje ar pasaulyje) naudojasi ja teikdami tokias paslaugas kurios neša bent kokį pelną ar naudą tiek pačiai įstaigai tiek finansų įstaigos klientui [1] toliau - klientas. Vienas iš labiausiai žinomų paslaugų yra komunalinių įmokų surinkimas. Finansinė įstaiga patampa centralizuotu mokesčių surinkimo „padaliniu“ komunalines paslaugas teikiančioms firmoms. Komunalinių įmokų surinkimas per bankines sistemas pliusai:

- firmos, įstaigos ir organizacijos sutaupo išlaidas skirtas surinkti mokesčius už suteiktas paslaugas jų klientams;
- patogus atsiskaitymo būdas. Klientas gali susimokėti būdamas beveik bet kurioje pasaulio vietoje susimokėti už jam suteiktas paslaugas;
- kokybiškas aptarnavimas, tinkamas duomenų apdorojimas ir jų pasikeitimas su firmomis (joms patogiais būdais ir formomis) teikiančiomis paslaugas. Dažniausiai firmos gauna duomenų failus apie gautas įmokas iš klientų arba pavedimais, kuriuos apdoroja pati firma;

- greitas pinigų pervedimas į firmų sąskaitas. Taip apsaugojamas klientas sumokėjęs už paslaugą ir firma suteikusi paslaugas: klientas nuo delspinigių laiku neatsiskaičius už suteiktas paslaugas, firma nuo nepatenkintų klientų ar galimų teisinių bylų;

- saugumas bankinėse sistemose yra labai svarbus, kuris užtikrina jog atitinkami duomenys bus perduoti užkoduoti ir saugiu kanalu. „Niekas nežinos“ už ką ir kiek jūs sumokėjote, bei klientas bus užtikrintas jog jo privatumas bus išsaugotas (gali sumokėti už paslaugą ir niekas nežinos jog tai tu sumokėjai).

Viskas turi kainą, todėl finansinės įstaigos renka komisinį mokestį už komunalinių paslaugų rinkimą. Šis mokestis būna sutartinis, tačiau jis nėra toks didelis palyginus su išlaidomis jei tektų įmonei pačiai rūpintis mokesčių surinkimu. Šioje vietoje visi vienaip ar kitaip laimi. Tai viena iš nišų kur finansinė įstaiga gali siekti pelno.

2.1.2 Tikslai

Projekto tikslai – sukurti bankinės sistemos komunalinių įmokų modulį ir pritaikyti realioje aplinkoje. Susipažinti su technologijomis naudojamomis kuriant bankines ir į jas panašias sistemas bei pritaikyti jas praktiškai. Išanalizuoti stipriąsias ir silpnąsias naudojamų technologijų vietas ir jomis pasinaudoti kuriant, testuojant ir palaikant sistemą.

2.1.3 Egzistuojantys sprendimai

Komunalinių įmokų modulis nėra naujas dalykas bankinėse sistemose. Visos jos yra kažkuo panašios ir kartu skiriasi. Finansinės įstaigos labiau linkę slėpti visą informaciją. Informacija susijusi su jų naudojamomis technologijomis ar kita informacija, kuri padidintų riziką sistemos sutrikdymui ar neteisėtam sistemos panaudojimui bei joje esama informacija, yra slepiama labiausiai. Tai labiausiai galinti pakenkti ir didžiausius nuostolius galinti padaryti informacija. Nors tikslas yra panašus, tačiau realizavimas skiriasi. Skiriasi šiais punktais:

- Programavimo kalba. Nors oficialių duomenų šiuo metu negaliu pateikti dėl teisinių priežasčių, tačiau galiu pasakyti jog yra kelios pagrindinės (populiariausios) programavimo kalbos, kurias naudoja kuriant bankines sistemas ir jų modulius. Pati dažniausiai paplitusi (-ios) yra C/C++ [14] programavimo kalbos, paskui eina JAVA™[6], VB (Visual Basic), C# ir kitos nuo senesnių laikų užsilikusios kalbos.

- Technologijos. Kuriant tokią sudėtingą sistema tikrai neapseisi be technologijų. Sistemos serverio dalis dažnai yra kuriama su vienokiomis technologijomis, o

klientinė dalis su kitomis ar papildant esamas. Jei sistema yra kuriama su C#, tai ją seka .NET [7] technologija (-os) (platforma), jei JAVA™ kalba kuriama sistema, tai ją lydi SUN korporacijos JVM [6] (platforma) ir kitos su ja siejamos technologijos. Sistema pastoviai sąveikauja su duomenų baze per vieną ar kitą sąsają (API). Sąsają kaip JDBC ir kitos technologijos skirtos sujungti duomenų bazę su sistema. Tai tik dalis naudojamų technologijų iš svarbesnių paminėti. Kurti nuo pagrindo viską savo jėgomis ir tik naudojant „grynai“ programavimo kalbą yra labai nuostolinga ir neoptimalu, tam naudojamos platformos (framework). Mano kuriamame modulyje yra panaudotos Hibernate[12, 8], Spring[9], EJB[12, 9] technologijos. Tai su JAVA programavimo kalba susijusios technologijos, kadangi mano sistema yra kuriama su šia kalba. *iBATIS* [11] technologija, kaip ir *hibernate* yra duomenų siejimo platforma, nors mano nuomone *iBatis* turi didesnį pranašumą, kalbant lankstume ir nustatymuose, tačiau kuriant tokio dydžio sistemą, *hibernate* patogumas ir paprastumas bei kūrimo greitis verčia rinktis šį variantą. Aš nesigilinsiu į žemesnį lygį, kuris siekia ryšių protokolus ar sąsajas tarp programinės įrangos ir įrenginių. Klientinėje dalyje irgi neapsieinama su įvairiausiomis technologijomis. Tokios technologijos kaip JSP [20], Servlet[21], Applet[22]. Šios technologijos yra labai geras pasirinkimas dinamiškai kuriant svetainės puslapius ar jų komponentes, bei atliekant daug daugiau nei paminėjau (daugiau informacijos galima rasti nurodytuose šaltiniuose). JavaScript yra naudojama klientinėje dalyje, jei reikia papildyti klientinę dalį reikiamais funkcionalumais. Šiame projekte yra naudojama papildoma biblioteka susijusios su JavaScript, tai jQuery [19]. jQuery yra JavaScript biblioteka palengvinanti programuotojams gyvenimą dirbant su HTML, animacija, įvykiais bei rašant skriptus JavaScript (JavaScript nėra Java technologija ar kitaip su ja susijusi).

- Šablonai. Šioje vietoja šablonas – programavimo šablonas arba kaip vienas iš geriausių problemos sprendimo būdų [3]. Šablonai paspartina probleminės srities sprendimo greitį, tačiau reikia juos gerai išmanyti ir žinoti kada juos reikia naudoti, kad šablonas netaptų antišablonu (Singleton neteisingai panaudotas gali pakenkti sistemai). Front Controller, Intercepting Filter, View Helper, Dispather View, Session Facade, tai tik dalis šablonų naudojamų kuriamoje sistemoje [3].

- Papildomos programos. Papildomos programos, tai programos papildančios sistemą ar įgalinančios jos veikimą. Programos skirtos vienai ar kelioms sritims: apsauga, failų valdymas, duomenų kodavimas ir kita. Duomenų ir failų kodavimui pasirinkta

pati stipriausia šių dienų technologija, tai PGP [4, 5] kodavimo standartas. Tai hibridinė duomenų kodavimo sistema apjungianti konvencionalaus (simetrinio rakto) ir viešo rakto kodavimo sistemas. Duomenų saugumas yra labai svarbus aspektas, todėl PGP [4, 5] standartas atitiko mano keliamus reikalavimus.

2.1.4 Architektūra

Kadangi prie sistemos kūrimo prisidėjau vėlai, jau sistema turėjo branduolį, todėl man teko prisitaikyti prie jos radikaliai nekeičiant branduolio, struktūros, architektūros.Sistema buvo kuriama naudojant N-TIER [15] architektūra. Ši architektūra yra tuo gerai jog ji apima toliau vardinamas architektūras: MVC [9] (šioje knygoje yra aprašyta apie MVC) arba 3-TIER [16] ar Client/Server. Ši architektūra yra tobulesnė ir teikianti daug naudos: stabilumas, tvirtumas, inovacijos, sluoksnių nepriklausomumas, kodo efektyvumas ir perpanaudojimas, plėtojimas.

2.1.5 Programavimo kalba

Pasirinkti programavimo kalbos, kaip ir architektūros aš negalėjau, kadangi sistema jau turėjo branduolį ir kitas dalis, kurios buvo sukurtos Java programavimo kalba. Java programavimo kalba yra aukšto lygio programavimo kalba, kuri pati „padeda“ programuotojui apsaugodama jį nuo galimų jo klaidų. Visi žinom kaip yra sunku suvaldyti atmintį, Java parašytos programos nereikia perkompiliuoti, kad paleisti ją ant kitos operacinės sistemos. Java turi „garbage collector“ (šiukšlių rinkėją), kuris automatiškai išvalo “nereikalingus” objektus. Tokia ypatybė palengvina ir paspartina programuotojo darbą. Java programavimo kalba turi daug privalumų tačiau ir minusų, dalis jų yra ta pati teikiama pagalba iš pačios programavimo kalbos. Atminties valdymą galima pamiršti, programų veikimo galimybės yra smėledėžėje, programų vykdymo/veikimo greitis mažesnis nei C++, Pascal ar OP bei kita programavimo kalba suprogramuota programa, kuri yra sukompiliuojama ir vykdoma tiesiogiai, o ne per interpretatorių, kaip Java programa. Mano nuomone, sistema turėtų veikti greitai ir išnaudoti galimus resursus kuo optimaliau, todėl iš minėtų programavimo kalbų rinkčiausi C/C++ arba *Delphi* (Object Pascal).

2.1.6 Operacinė sistema

Sistema veikia *Debian GNU/Linux* operacinėje sistemoje. Daugiau informacijos apie šią operacinę sistemą ir jos teikiamus privalumus < http://www.debian.org/ >. Šis sprendimas buvo pasirinktas dėl saugumo, stabilumo ir kainos. *Debian* yra nemokama operacinė sistema, pasižyminti stabilumu, lankstumu ir saugumu. Oficialioje svetainėje yra parašyta, kad su šiai sistemai yra sukurta daugiau kaip 25000 paketų (įvairiausių taikomųjų programų), kurios yra nemokamos ir

atviro kodo kaip ir pati operacinė sistema. Nors sistema gali veikti ir *Windows* OS, tačiau pati OS yra mokama ir didžioji dalis taikomųjų programų veikiančių šioje OS yra mokamos. Taip pat stabilumu ši operacinė sistema labai nepasižymi, nes internete pilna nusiskundimų dėl OS darbo sutrikimų.

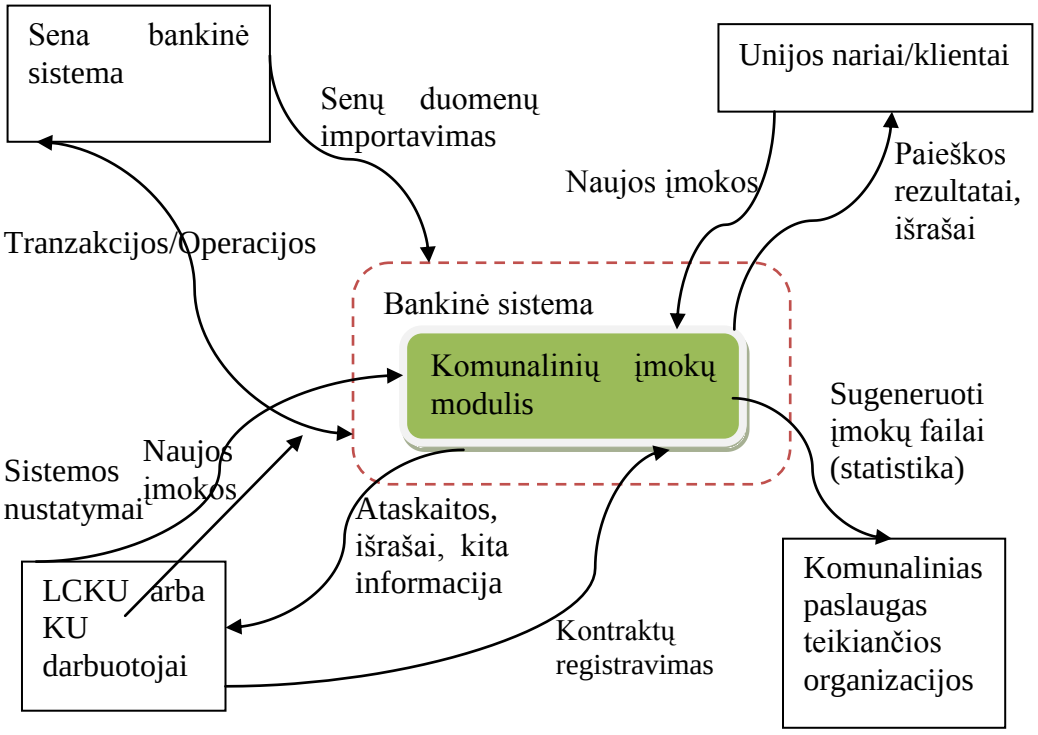
2.1.7 Duomenų bazė

MySql, Oracle, MSSQL, DB2, Sybase, tai tik trumpas sąrašas iš galimų duomenų bazių. Kiekviena iš jų turi savų pliusų ir minusų, tačiau tai nemano tyrimo objektas todėl jų nevardinsiu. Sistema naudoja *Oracle* duomenų bazę. Daugiau informacijos apie *Oracle* galima rasti oficialioje svetainėje adresu < http://www.oracle.com/index.html >.

2.2 Veiklos sudėtis (funkciniai reikalavimai)

2.2.1 Veiklos kontekstas (pateikiama konteksto diagrama)

Toliau pristatysiu modulio galimą veiklos konteksto diagramą. Kadangi tai bus esamos sistemos dalis, ji bus pavaizduota diagramoje kaip integruotas modulis korinėje sistemoje.



1 pav. Bankinės sistemos komunalinių įmokų modulio ir aplinkos kontekstas.

2.2.2 Veiklos padalinimas

Turėdamas veiklos konteksto diagramą (1 pav.), pateikiu veiklos įvykių sąrašą 1 lentelėje.

1 lentelė. Veiklos įvykių sąrašas.

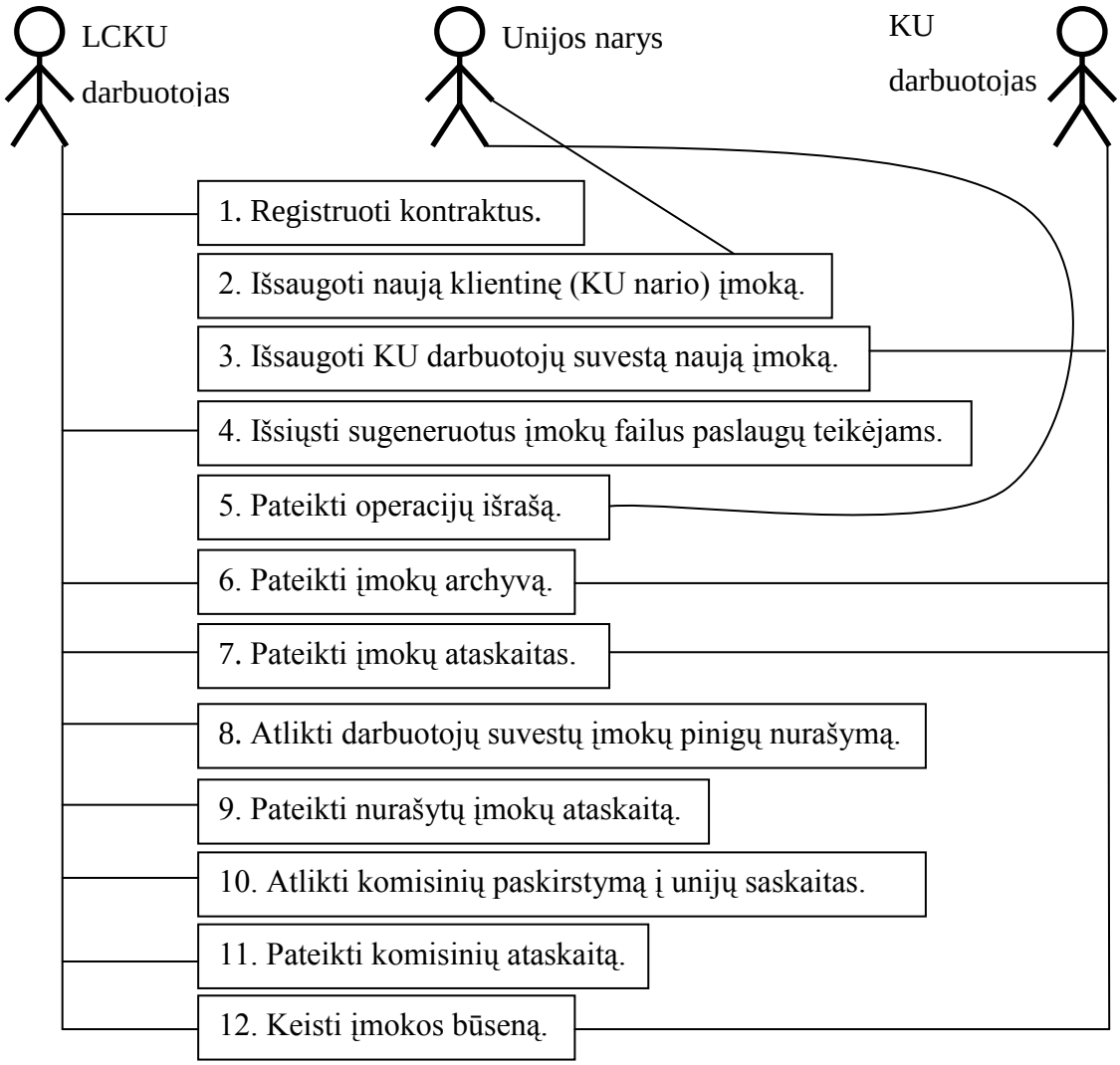
Eil. Nr.	Įvykio pavadinimas	Įeinantys ir išeinantys informacijos srautai
•	**Sena bankinė sistema perduoda senus duomenis.**	Senų duomenų importavimas (in).
•	**Senos bankinės sistemos keitimasis transakcijomis su nauja.**	Transakcijos/Operacijos (in, out).
•	**Unijos narys suveda naują įmoką.**	Naujos įmokos (in)
•	**Unijos narys užklausia atliktų įmokų suvestinės.**	Paieškos rezultatai, išrašai (out).
•	**LCKU darbuotojas nustato kontraktų parametrus arba nustato tam tikrus sistemos parametrus.**	Sistemos nustatymai (in).
•	**Sistema gražina darbuotojo įvairiausių užklausų atsakymus.**	Ataskaitos, išrašai, kita informacija (out).
•	**LCKU darbuotojos užregistruoja kontraktus, kurios unijos dalyvauja ir turi teisę surinkinėti komunalines įmokas.**	Kontraktų registravimas (in)
•	**KU darbuotojas suveda naują įmoką.**	Naujos įmokos (in).
•	**Gavusi komandą, sistema sugeneruoja failą ir išsiunčia į organizacijos el. paštą.**	Sugeneruoti įmokų failai (statistika) (out).

Iš esamų devynių įvykių tik septyni iš jų tiesiogiai susyja su moduliu, kiti du įvykiai tik dalinai (Senos bankinės sistemos keitimasis transakcijomis su nauja.) arba iš vis nesusyja su moduliu (Sena bankinė sistema perduoda senus duomenis.).

2.2.3 Sistemos sudėtis

2.2.3.1 Sistemos ribos

Sistemos ribas tarp vartotojų nusako žemiau pavaizduota panaudojimo atvejų diagrama (2 pav.).



2 pav. Panaudojimo atvejų diagrama.

2.2.4 Panaudojimo atvejų sąrašas

Šiame panaudojamo atvejų sąraše nėra minimas vartotojo prisijungimo procesas, nes manoma jog jis jau yra įvykdytas anksčiau.

1 PANAUDOJIMO ATVEJIS: Registruoti kontraktus.

Vartotojas/aktorius: LCKU darbuotojas

Aprašas: Apima procesą, kurio metu registruojamas kontraktas (-ai) su unija (-omis), kuri (-ios) galėtų rinkti komunalines įmokas nurodytas tame kontrakte. Taip pat nurodomi įvairiausi kontrakto parametrai, bei nurodoma ar leidžiama tos unijos nariams matyti ir saugoti tą komunalinę įmoką.

Prieš sąlyga: Tokio kontrakto nėra su tokia unija. Tokia sutartis buvo nutraukta su šia unija.

Sužadinimo sąlyga: Kontrakto lange suvedami visi atitinkami rekvizitai ir reikalingi parametrai. Unijų lange pažymima unija su kuria ar kuriomis sudaromas kontraktas ir pažymima ar šią komunalinę įmoką galės atlikti pažymėtų unijų nariai. Patvirtinamas kontrakto duomenų korektiškumas ir duomenų išsaugojimas vieno mygtuko paspaudimu.

Po sąlyga: Išsaugomas naujas ar nauji kontraktai, ar pakeičiamas seno kontrakto (-ų)

2 PANAUDOJIMO ATVEJIS: Išsaugoti naują klientinę (KU nario) įmoką.

Vartotojas/aktorius: Unijos narys

Aprašas: Tai procesas, kuriame unijos narys pasirinkdamas iš komunalinių įmokų sąrašo ar anksčiau išsaugotų šablonų, teisingai užpildęs duomenis išsaugo įmoką (apmoka sąskaitą už jam suteiktas komunalines paslaugas).

Prieš sąlyga: Įmoka nėra apmokėta. Pasirenkama įmokų sąraše (įmokų šablonų sąraše) komunalinė įmoka ir atsidaro įmokos užpildymo forma.

Sužadinimo sąlyga: Užpildoma visa teisinga informacija, kuri būtina, kad išsisaugotų įmoką. Toliau patvirtinus kortelėje esančiu numeriu, kurio sistema pareikalauja ir paspaudus mygtuką įvyksta saugojimo procesas.

Po sąlyga: Išsaugojama nario (kliento) įmoka ir pinigai iš nario pervedami specialią LCKU surenkamąją sąskaitą, kurie toliau keliaus į komunalines paslaugas teikiančias organizacijas už kurias klientai sumokėjo pinigus.

3 PANAUDOJIMO ATVEJIS: Išsaugoti KU darbuotojo suvestą naują įmoką.

Vartotojas/aktorius: KU darbuotojas

Aprašas: Procesas apimantis komunalinės įmokos apmokėjimą (įmokos duomenų suvedimu į sistemą) už unijos narį, kai narys sumoka grynais už įmoką bei darbuotojui pateikia mokėjimo dokumentą. Mokėjimo dokumento informaciją darbuotojas suveda į sistemą.

Prieš sąlyga: Įmoka nėra apmokėta. Pasirenkama įmokų sąraše komunalinė įmoka ir atsidaro įmokos užpildymo forma.

Sužadinimo sąlyga: Užpildoma visa teisinga informacija, kuri būtina, kad išsisaugotų įmoką ir mygtuko paspaudimu patvirtinama įmokos apmokėjimas ir jos išsaugojimas DB.

Po sąlyga: Nauja įmoka yra išsaugota.

4 PANAUDOJIMO ATVEJIS: Išsiųsti sugeneruotus įmokų failus paslaugų teikėjams.

Vartotojas/aktorius: LCKU darbuotojas

Aprašas: Šis procesas apima vieno kontrakto įmokų pagal tam tikra laikotarpį sugeneravimą į vieną failą (jei reikia užkoduojamas) ir išsiunčiama į komunalines paslaugas teikiančią organizaciją pagal kontrakte nurodytą (-us) elektroninį (-ius) paštą (-us).

Prieš sąlyga: Būtina jog būtų įmokų su būsena *Įvykdyta* siunčiamam laikotarpiui.

Sužadinimo sąlyga: Pasirinktam kontraktui ir nurodžius laikotarpį bei papildomus parametrus paspaudžiamas mygtukas atlikti įmokų failų generavimui ir siuntimui.

Po sąlyga: Sugeneruotas įmokų failas yra išsaugomas duomenų bazėje ir išsiunčiamas į organizaciją.

5 PANAUDOJIMO ATVEJIS: Pateikti operacijų išrašą.

Vartotojas/aktorius: Unijos narys

Aprašas: Nupasakojamas procesas, kuriame unijos narys pagal nurodytus parametrus gauna iš sistemos sąrašą operacijų.

Prieš sąlyga: Narys gaus rezultatą jei bus atlikta bent viena operacija, tai gali būti atlikta komunalinė įmoka arba pavedimas. Užklausos parametrai turi būti logiškai teisingi.

Sužadinimo sąlyga: Pasirinkus meniu *Operacijų archyvas* nurodžius užklausos parametrus paspaudžiamas mygtukas, kuris kreipiasi į DB su užklausa.

Po sąlyga: Sistema gražina rezultatą pagal anksčiau nurodytus užklausos parametrus.

6 PANAUDOJIMO ATVEJIS: Pateikti įmokų archyvą.

Vartotojas/aktorius: LCKU ir KU darbuotojas

Aprašas: Procesas, kai darbuotojas gauna įmokų archyvą, pagal atitinkamus užklausos parametrus, bei priklausomai nuo darbuotojų turimų teisių.

Prieš sąlyga: Turi turėti atitinkamas teises, kad matytų įmokų archyvą ar galėtų matyti įmokas atliktas kitų unijų. Tik LCKU darbuotojas gali matyti visų unijų darbuotojų ir narių suvestas įmokas. Turi būti bent viena įmoka.

Sužadinimo sąlyga: Paieškos formoje pasirenkami parametrai pagal kurią bus įvykdyta užklausa ir mygtuko paspaudimu sužadinama komandos vykdymas.

Po sąlyga: Priklausomai nuo užklausos parametrų, darbuotojas gali gauti eilę rezultatų arba nei vieno įrašo.

7 PANAUDOJIMO ATVEJIS: Pateikti įmokų ataskaitas.

Vartotojas/aktorius: LCKU ir KU darbuotojas

Aprašas: Procesas, kurio vykdymo metu darbuotojas gauna sugeneruotą specialią įmokų ataskaitą pagal atitinkamus užklausos parametrus.

Prieš sąlyga: Darbuotojas turi turėti teisias naudotis šiuo funkcionalumu, bei turi būti bent viena įmoka.

Sužadinimo sąlyga: Nustatoma užklausos parametrai ir mygtuko paspaudimu aktyvuojama užklausos vykdymo komanda.

Po sąlyga: Priklausomai nuo užklausos parametrų, darbuotojas gauną suformuotą ataskaitą arba gauna tuščią ataskaitą.

8 PANAUDOJIMO ATVEJIS: Atlikti darbuotojų suvestų įmokų pinigų nurašymą.

Vartotojas/aktorius: LCKU darbuotojas

Aprašas: Nusakomas procesas kuomet LCKU darbuotojas nurašo pinigus iš unijų kuriose buvo suvesta naujų įmokų ir pinigai pervedami į specialę LCKU sąskaitą iš kurios atsiskaitoma su paslaugų tiekėjais.

Prieš sąlyga: Būtina turėtu tam teises. Turi būti suvestos teisingos sąskaitos. Turi būti bent viena nauja įmoka.

Sužadinimo sąlyga: Pasirinkus iš įmokų sąrašo atitinkamą įmoką už kurią bus nuskaitomi pinigai ir mygtuko paspaudimu aktyvuojamas procesas.

Po sąlyga: Naujos darbuotojų įmokos yra nurašomos ir jų būsenos patampa *Įvykdyta*. Pinigai surenkami iš unijų į vieną sąskaitą.

9 PANAUDOJIMO ATVEJIS: Pateikti nurašytų įmokų ataskaitą.

Vartotojas/aktorius: LCKU darbuotojas

Aprašas: Procesas suformuojantis nurašytų įmokų ataskaitą.

Prieš sąlyga: Turėti atitinkamas teisias.

Sužadinimo sąlyga: Ataskaita automatiškai sugeneruojama po sėkmingo įmokų nurašymo.

Po sąlyga: Gaunama suformuota ataskaita nuo kokių unijų buvo nurašytos įmokos ir surinkti pinigai, bei kita naudinga informacija.

10 PANAUDOJIMO ATVEJIS: Atlikti komisinių paskirstymą į unijų sąskaitas.

Vartotojas/aktorius: LCKU darbuotojas

Aprašas: Procesas, kuris atliekamas LCKU darbuotojo, gauti komisiniai iš paslaugų teikėjų yra paskirstomi toms unijoms, kurios buvo surinkę tas įmokas ar jų nariai sumokėjo per internetinę bankininkystę.

Prieš sąlyga: Būtina turėti tam teisias. Paskirstyti galima tik už praeitą mėnesį.

Sužadinimo sąlyga: Parenkamas kontraktas/įmoka, parenkama data ir mygtuko paspaudimu aktyvuojamas procesas.

Po sąlyga: Komisiniai paskirstomi unijoms.

11 PANAUDOJIMO ATVEJIS: Pateikti komisinių ataskaitą.

Vartotojas/aktorius: LCKU darbuotojas

Aprašas: Procesas, kuris sugeneruoja komisinių ataskaitą. Čia komisiniai yra tie kuriuos CKU paskirstė unijoms gautus iš paslaugų teikėjų.

Prieš sąlyga: Turėti tam teisias ir būtina turėti kiek nors pradinių duomenų.

Sužadinimo sąlyga: Pasirenkamas laikotarpis (laikotarpis negali būti einamas mėnuo ar kitas) bei pasirenkama norimos įmokos ataskaita. Mygtuko paspaudimu aktyvuojama užklausa.

Po sąlyga: Gražinama paskirstytų komisinių unijoms ataskaita už praėjusį laikotarpį.

12 PANAUDOJIMO ATVEJIS: Keisti įmokos būseną.

Vartotojas/aktorius: LCKU ir KU darbuotojas

Aprašas: Tai svarbus procesas, kuomet unijos darbuotojas tam tikru momentu dar gali pakeisti įmoks būseną iš *Laukianti apmokėjimo* į *Negaliojanti*. Tai įmokos trynimo arba jos padarymas negaliojančia procesas.

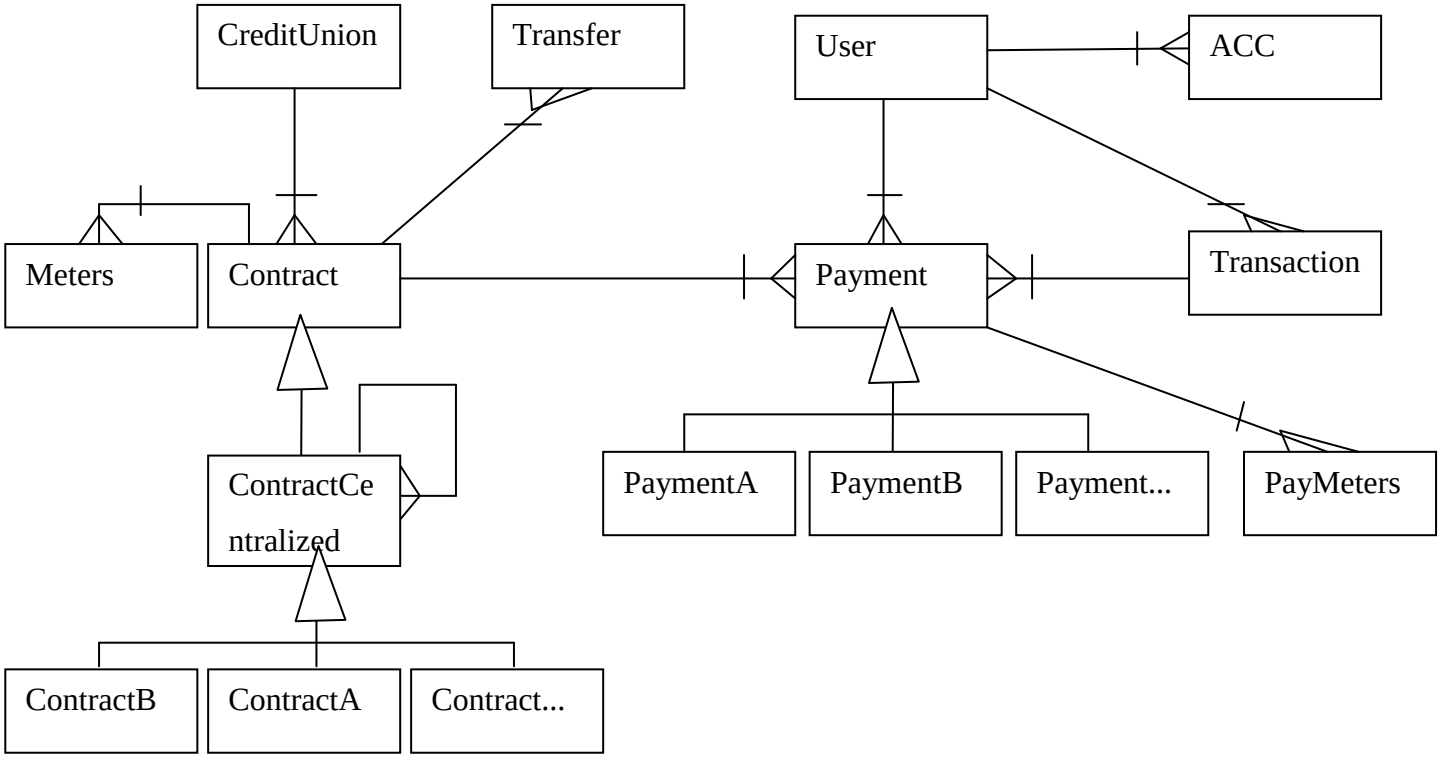
Prieš sąlyga: Turi turėti tam teisias. Jei tai tik paprastas unijos darbuotojas jis gali ištrinti tik įmokas kurių būsena yra *Laukianti apmokėjimo* ir tik savo ar priklausomai nuo teisiu tik savo unijos darbuotojų. LCKU unijos darbuotojas gali trinti visas įmokas jei turi tam teisias.

Sužadinimo sąlyga: Įmokų archyve įvykdžius užklausą gaunamas įmokų sąrašas. Pasirinkus įmoką ir šalia įrašo paspaudus mygtuką *Šalinti*, yra aktyvuojamas įmokos būsenos keitimas į *Negaliojanti*.

Po sąlyga: Sėkmingu atveju įmokos būsena pakeičiama į *Negaliojanti*.

2.2.5 Reikalavimai duomenims

Toliau pateikiama galima sistemos duomenų modelio struktūra (3 pav.).



3 pav. Klasių diagrama.

2.3 Nefunkciniai reikalavimai

2.3.1 Reikalavimai sistemos išvaizdai

Sistemos išvaizda turi būti suderinta su marketingo ir elektroninių paslaugų skyriumi. Sistema turi naudoti jau sukurtus mygtukus ir kitus standartinius grafinės sąsajos elementus. Puslapio struktūra turi atitikti bendram vaizdui ir stipriai neišsiskirti. Privaloma laikytis standartinių spalvų tai žalios ir baltos. Visi laukai kurie traktuojami kaip skaičiais, turi būti išlygiuoti iš dešinės pusės, o tekstas iš kairės arba per centrą. Laukai turi būti tvarkingai sudėstyti kaip ir mygtukai ar kiti komponentai. Būtina „taupyti“ mygtukų paspaudimo skaičių. Pranešimai apie klaidas turi būti raudonos spalvos. Prireikus turi būti galimybė pakeisti spalvas ar šriftą visai svetainei ar jos kažkuriai daliai.

2.3.2 Reikalavimai panaudojamumui

Reikalavimas #:	*14*	**Reikalavimo tipas:**	*12*	**Įvykis/panaudojimo atvejis #:**	*1-12*
Aprašymas:	***Komponentai gali būti panaudoti kituose moduliuose.***				
Pagrindimas:	***Tai sumažins programuotojo laiką iš naujo sukurti tą patį objektą, komponentę.***				
Šaltinis:	**Tomas Žvirgžda**				
Tikimo kriterijus:	***Komponentas ar objektas veikia kitoje formoje ar modulyje be jokiu priklausomybių.***				
Užsakovo tenkinimas:	*1*	**Užsakovo netenkinimas:**			*5*
Priklausomybės:	*Nėra*	**Konfliktai:**	*Nėra*		
Papildoma medžiaga:	*Nėra*				
Istorija:	***Užregistruotas 2010 balandžio 25 d.***				

Reikalavimas #:	*15*	**Reikalavimo tipas:**	*12*	**Įvykis/panaudojimo atvejis #:**	*1-12*
Aprašymas:	***Sistemos navigacija lengvai įsisavinama.***				
Pagrindimas:	***Kad greičiau asmuo įsisavintų į sistema ir mažiau patirtų streso bei pykčio..***				
Šaltinis:	**Tomas Žvirgžda**				
Tikimo kriterijus:	***Navigacija paprasta ir mygtukų pavadinimai aiškiai nusako savo prasmę.***				
Užsakovo tenkinimas:	*2*	**Užsakovo netenkinimas:**			*5*
Priklausomybės:	*Nėra*	**Konfliktai:**	*Nėra*		

Papildoma medžiaga:	*Nėra*
Istorija:	*Užregistruotas 2010 balandžio 25 d.*

2.3.3 Reikalavimai vykdymo charakteristikoms

Būtina jog užklausas įvykdytų kuo greičiau, todėl tam reikia:

- Spartaus tinklo tarp DB serverio ir aplikacijos serverio.
- Spartaus DB ir aplikacijos serverio.
- Sumažinti kur galima tiek DB tiek aplikacijos serverio apkrovimą (jei galima dalį duomenų apdorojimo darbo perkelti pas vartotoją).
- Jei galima duomenis DB sandėliuoti.

2.3.4 Reikalavimai veikimo sąlygoms

Sistemai užtikrinti veikti nesustojus, reikia dviejų aplikacijos serverių, kad vienam serveriui sugedus būtų galima kitą paleisti kaip atsarginį. Būtina turėti ir du DB serverius, kuris vienas iš jų būtų atsarginis, jei kitas sugestų. Serveriai turi būti saugomi specialiose patalpose, kad jie būtų apsaugoti nuo vandens, ugnies ar įtampos iškrovų bei kitų stichinių nelaimių. Serveriai turi būti visą laiką stebimi ir reguliariai patikrinami.

2.3.5 Reikalavimai sistemos priežiūrai

Sistemos priežiūros išlaidos turi būti optimalios. Prie sistemos gali prieiti tik tam teises turintys asmenys. Turi būti atliekami tik priežiūros veiksmai, o ne įvairiausi testavimai ar bandymai išmėginti sistemos galingumą ir kaip ji elgsis tam tikromis sąlygomis.

2.3.6 Reikalavimai saugumui

Sistema turi būti saugi serverio pusėje ir kiek įmanoma vartotojo pusėje. Prisijungimai turi būti dviejų pakopų. Pirma pakopa tai vartotojo identifikavimo numeris ir slaptažodis. Antra pakopa kortelės numeris. Sistemos nustatymus gali atlikti tik tam teises turinys asmenys. Slaptažodžiai turi būti keičiami periodiškai. Padaromas priverstinis vartotojų slaptažodžių pakeitimas kas tam tikrą laikotarpį. Būtina laikytis konfidencialumo ir duomenų integralumo ir vientisumo.

2.3.7 Kultūriniai ir politiniai reikalavimai

Visas matomas tekstas vartotojui turi būti lietuvių kalba. Sistemai kurti naudojami tik nemokami įrankiai. Sistemoje turi būti naudojamos nemokamos bibliotekos ir komponentai. Pranešimai sistemoje negali būti įžeidžiantys ar rasistinio pobūdžio.

2.3.8 Teisiniai reikalavimai

Sistema negali pažeisti jokio įstatymo galiojančio Lietuvoje. Sistema gali naudotis tik unijos darbuotojai ir unijos nariai.

3 PROJEKTINĖ DALIS

3.1 Architektūros tikslai ir apribojimai

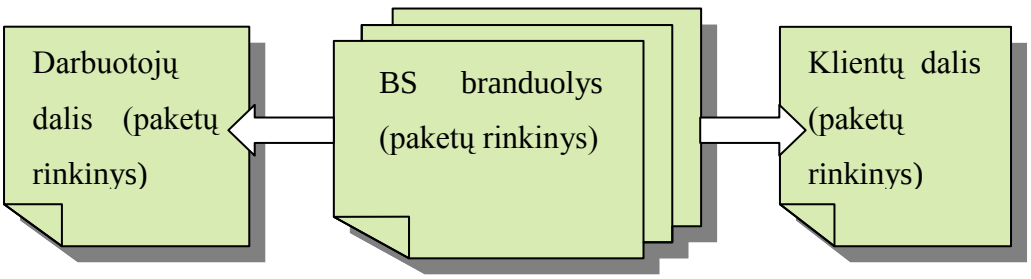
Sistemos tikslai ir apribojimai:

- Sistema turi būti centralizuota.
- Sistema turi būti lengvai plečiama.
- Sistema turi būti saugi.
- Klientinė ir darbuotojų dalys turi būti atskiros aplikacijos, tačiau naudotis bendru branduoliu.
- Galimybė riboti modulio matymo ar naudojimosi teisės.
- Sistema turi veikti kliento/serverio modeliu.
- Serverinė dalis turi būti realizuojama su Java™ programavimo kalba ir objektiškai orientuota sistema.
- Sudėtingus ir/ar didelius duomenų apdorojimus paskirstyti tarp duomenų bazės ir aplikacijos.
- Sistema turi veikti Linux aplinkoje.
- Sistemos kūrimui ir naudojami komponentai ar technologijos turi būti nemokamos.

3.2 Sistemos statinis vaizdas

3.2.1 Apžvalga

Dokumente yra anksčiau paminėta, jog sistema turi būti sudaryta iš dviejų dalių, tai klientinė dalis ir darbuotojų dalis. Kiekviena dalis turi bendrą branduolį. 4 pav., diagramoje yra pavaizduotas vaizdinis sistemos paketų aukščiausias modelis. Paketų detalesnis skaidymas yra pateiktas kitame skyriuje.

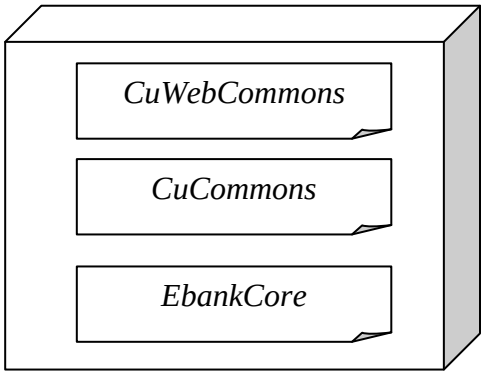


4 pav. Sistemos paketų aukščiausias lygis.

3.2.2 Paketų detalizavimas

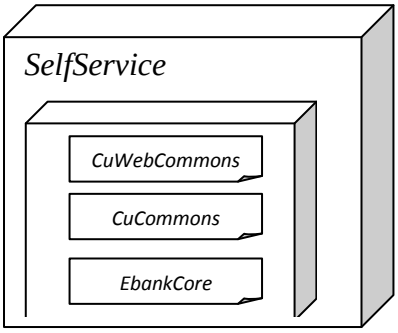
Kadangi sistema sudaryta bus iš dviejų aplikacijų, tačiau vis tiek naudos bendrą branduolį, todėl pirmas paketas bus išskaidytas ir pristatytas – branduolys. Branduolį sudaro keli paketai, kiekvienas iš jų turi tam tikrą skaičių paketų, kuriuose yra naudojamos klasės – sudarančios visą branduolio visumą. Tie pagrindiniai branduolio paketai yra: *CuCommons*, *EbankCore*, *CuWebCommons*. Šiuose paketuose yra dar daugiau mažesnių paketų, tačiau iki čia sustosiu ir

toliau pristatinėsiu tik su moduliu susijusias klases arba paketus. 5 pav., yra pateiktas BS branduolio vaizdas, kur kiekvienas paketas yra šiek tiek susijęs su vienu iš kitų vidinių paketų.

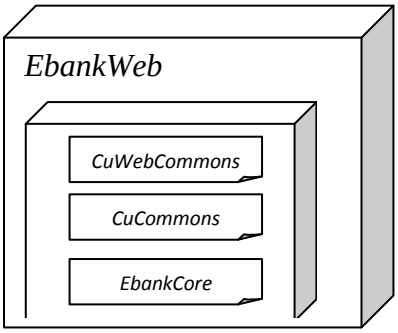


5 pav. BS branduolys.

Klientinės dalies paketą kaip ir darbuotojų dalies paketą sudaro vienas didelis paketas. Kiekvienas paketas naudoja BS branduolio paketus ir jose esančias klases. Šių paketų grafinis vaizdas yra pateiktas 6 pav., ir 7 pav. Klientinės dalies paketas yra *SelfService*, o darbuotojų *EbankWeb* pavadinimais.



6 pav. SelfService paketas.



7 pav. EbankWeb paketas.

Didžioji dalis esamų klasių šiuose paketuose negaliu atskleisti dėl teisinių priežasčių. Pateiksiu tik tas klases, kurios tiesiogiai susijusios su moduliu ir kai kurios bus nedetalizuojamos dėl teisinių priežasčių.
Korinio paketo klasės:

- **Contract** – klasė reprezentuojanti kontraktą, kurį „pasirašė“ kredito unija su LCKU.
- **CreditUnion** – klasė reprezentuojanti kredito uniją.
- **Meters** - klasė reprezentuojanti skaitiklių nustatymus.
- **ContractCentralized** – tai *Contract* subklasė, kuri nurodo jog tai centralizuotas kontraktas.

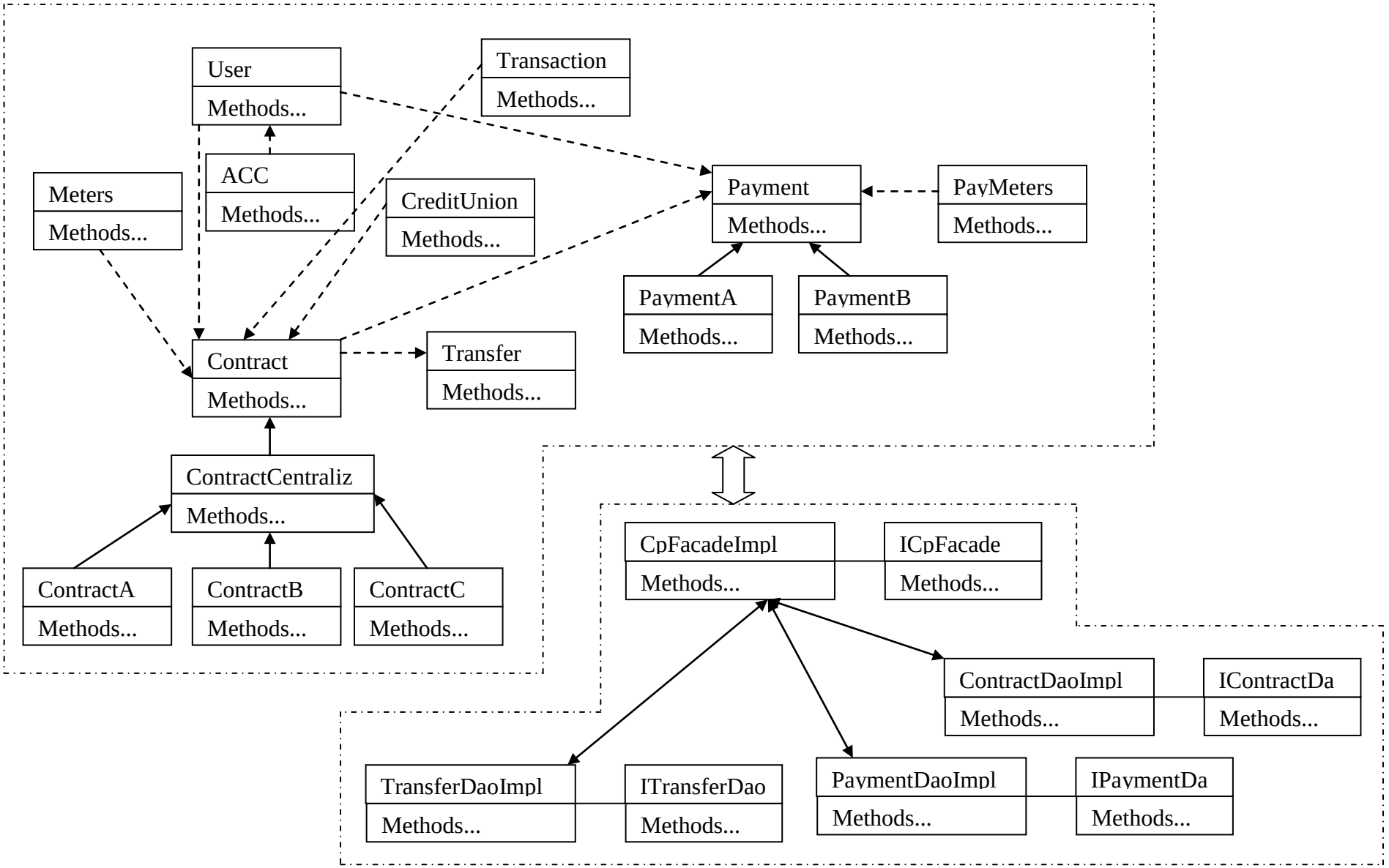
- **ContractA, ContractB, ContractC ... –** tai subkontraktai centralizuotų kontraktų klasės, turintys skirtingus atributus ar skirtingus atributų parametrus.
- **Transfer** – failų siuntimus ir juos pačius reprezentuojanti klasė.
- **Payment** – komunalinę įmoką reprezentuojanti klasė.
- **PaymentA, PaymentB...** – tai *Payment* klasės subklasė, kuri turi papildomus skirtingus atributus arba skirtingus atributų parametrus.
- **PayMeters** – įmokos skaitliukus reprezentuojanti klasė.
- **User** – vartotoją reprezentuojanti klasė.
- **ACC –** sąskaitos klasė.
- **Transaction** – pavedimo klasė.

Šių klasių diagrama yra pavaizduota 8 pav. Anksčiau išvardintos klasės yra tik duomenis reprezentuojančios klasė. Duomenų apdorojimui ir gavimui yra sukuriamos kitos klasės.

- **IContractDao** – kontrakto duomenų gavimo/apdorojimo klasės sąsaja.
- **ContractDaoImpl -** kontrakto duomenų gavimo/apdorojimo klasė.
- **ICpFacade** – įvairiausių duomenų apdorojimo ir nuorodos į DAO klases sąsaja.
- **CpFacadeImpl** – įvairiausių duomenų apdorojimo ir nuorodos į DAO klasė.
- **IPaymentDao –** įmokų duomenų gavimo/apdorojimo klasės sąsaja.
- **PaymentDaoImpl –** įmokų duomenų gavimo klasė.
- **ITransferDao** – siuntimų (failų siuntimų) duomenų gavimo/apdorojimo klasės sąsaja.
- **TransferDaoImpl** – siuntimų (failų siuntimų) duomenų gavimo/apdorojimo klasė.

Visos šios klasės taip pat pavaizduotos 8 pav. 8 paveikslėlyje pavaizduotoje klasių diagramoje nėra detalizuoti metodai dėl didelio jų kiekio. Dalį detalizuotų metodų bus pateikta detalesnėje architektūroje.

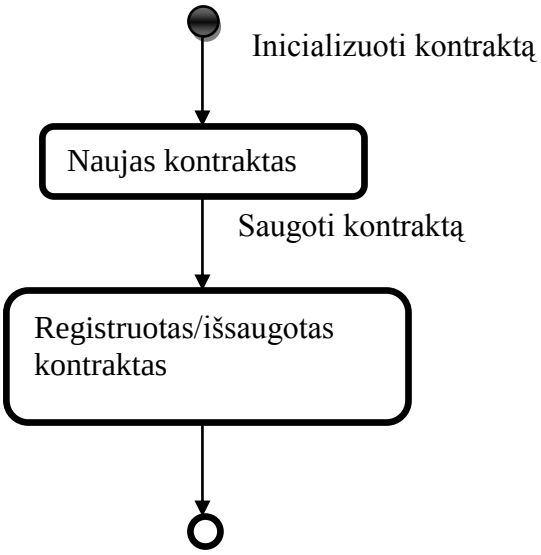
Klientinės dalies ir darbuotojų dalies paketus sudaro daugybe klasių kurios atitiktų tokį *Model-View-Control*. Model klasės skirtos apdoroti duomenis ir juos perduoda į View klases, kurios sugeneruoja *html* ar kitą reikalingą kodą ir siunčia duomenų atvaizdavimą su visa svetainės informacija į *Control* klases, kuri paskirsto, priima duomenų srautus vartotojams (siunčia užklausų atsakymus arba priima užklausas) arba jų nepraleidžia. Toliau architektūros specifikacijoje bus pristatinėjamos anksčiau minėtos korinės klasės, tačiau prireikus papildyti, bus pridėta papildoma klasė iš vieno ar kelių kitų paketų.



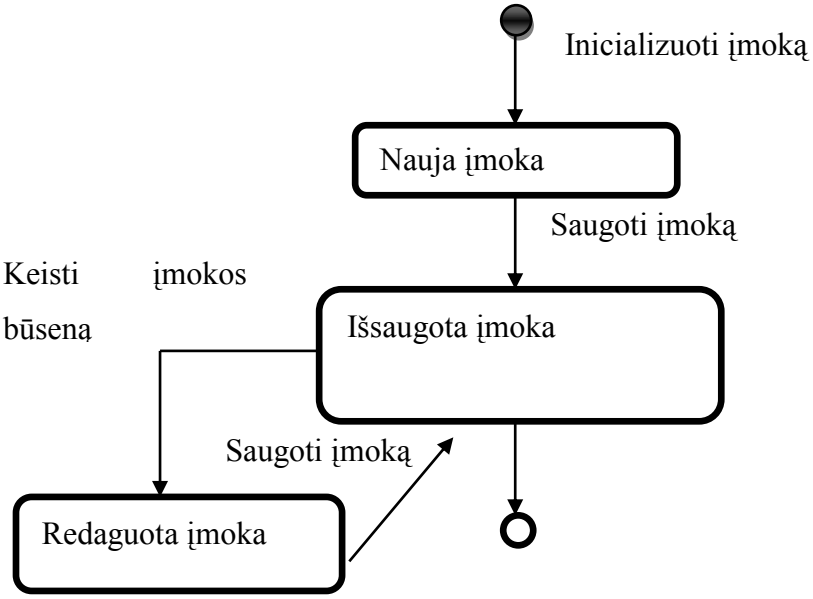
8 pav. BS branduolio klasių diagrama.

3.3 Sistemos dinaminis vaizdas

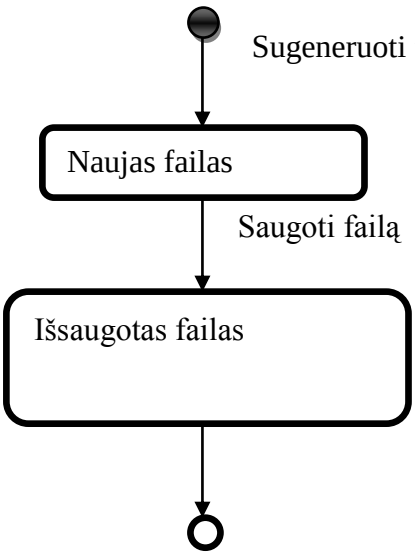
3.3.1 Būsenos diagrama



9 pav. Kontrakto būsenos diagrama.



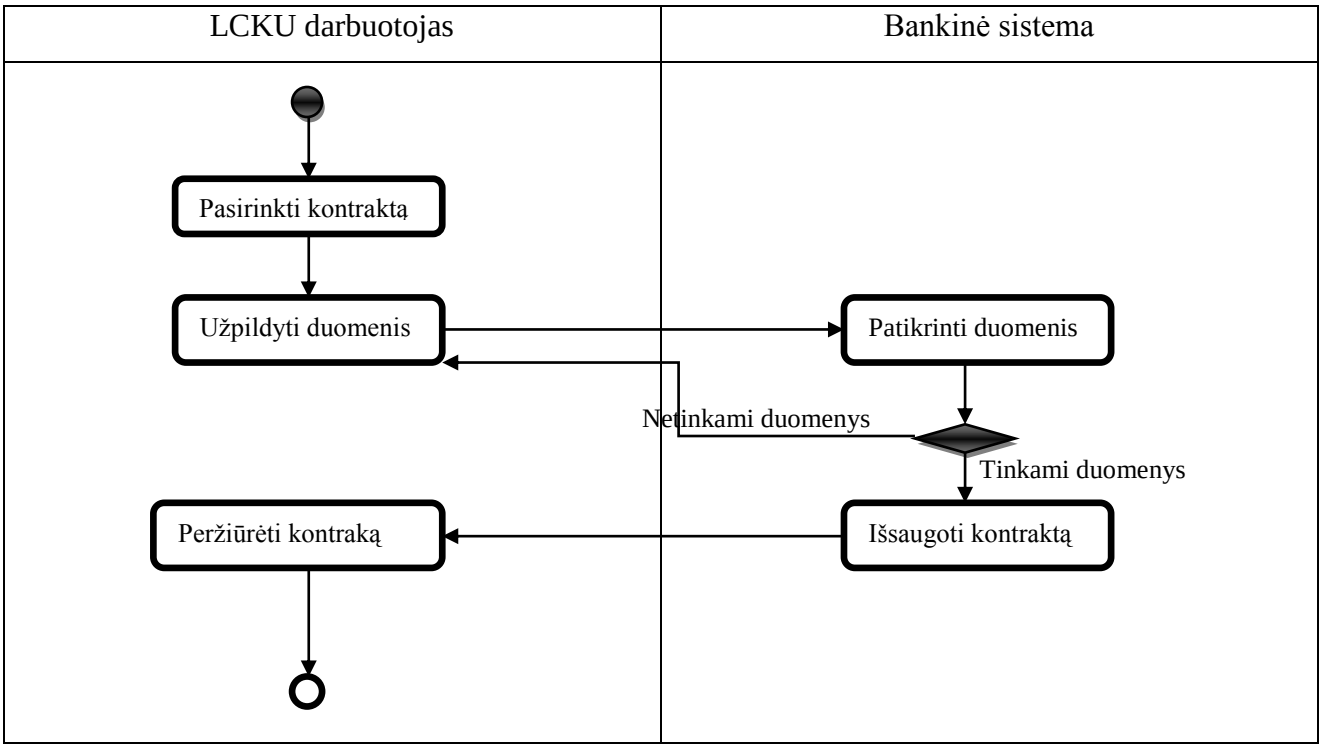
10 pav. Įmokos būsenos diagrama.



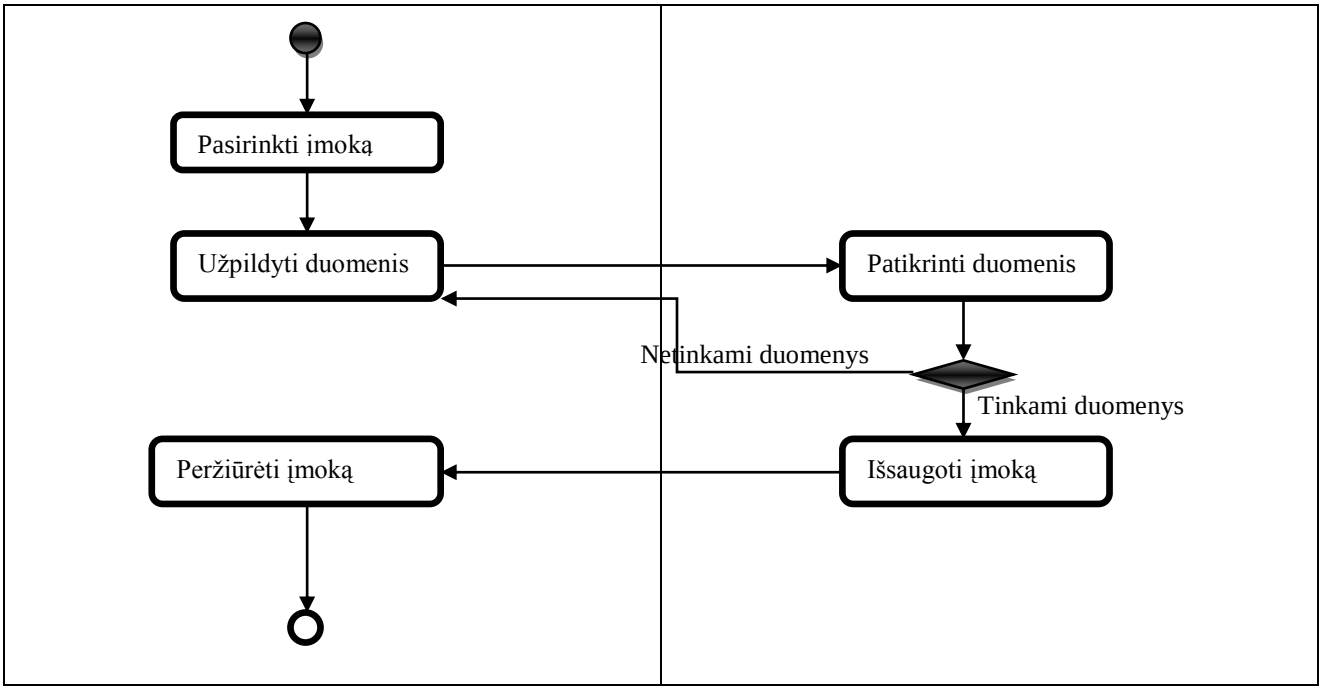
11 pav. Failo sugeneravimo būsenos diagrama.

3.3.2 Veiklos diagrama

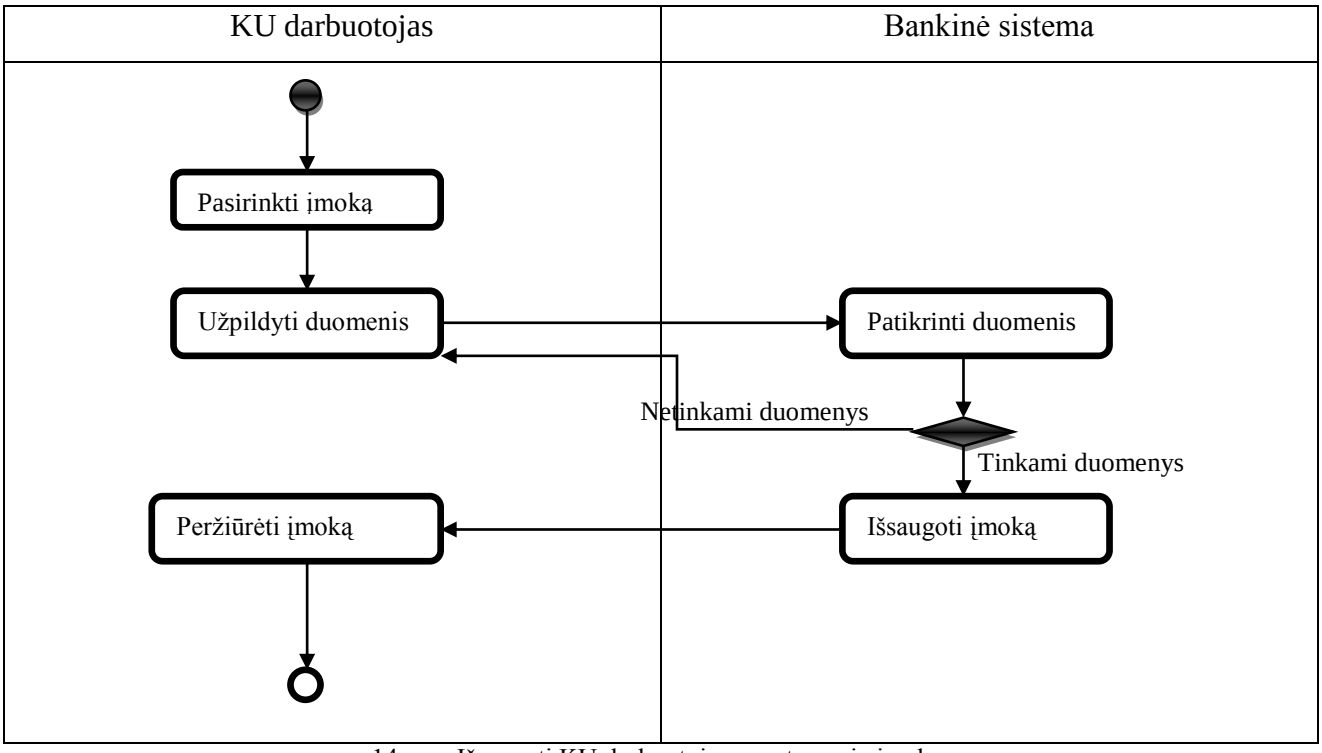
Toliau bus pateikta kiekvienam panaudojimo atvejui veiklos diagrama.



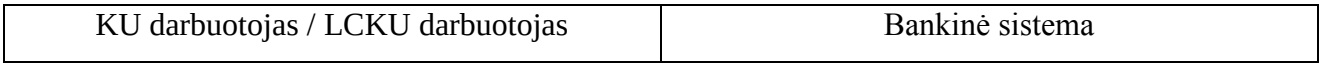
12 pav. Registruoti kontraktus.

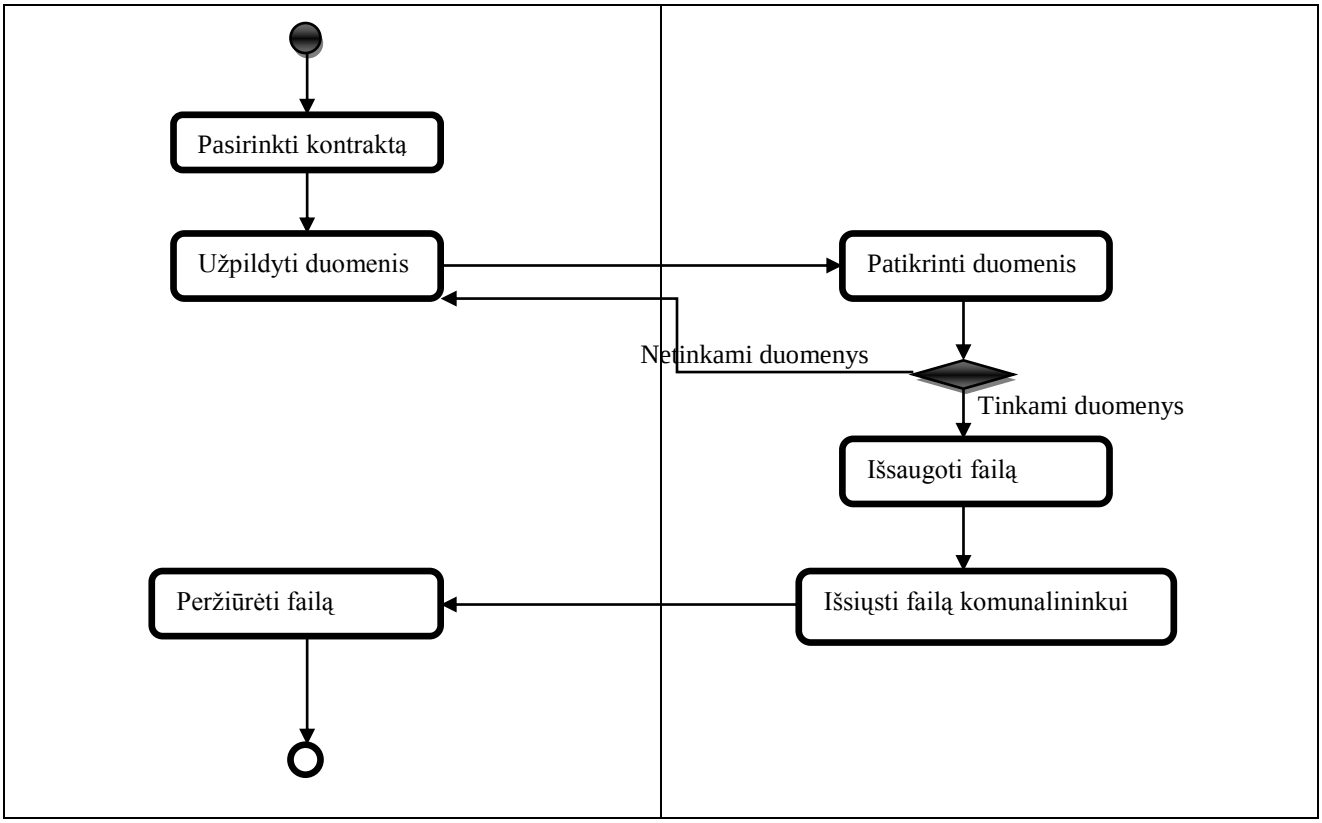


13 pav. Išsaugoti naują klientinę (KU nario) įmoką.

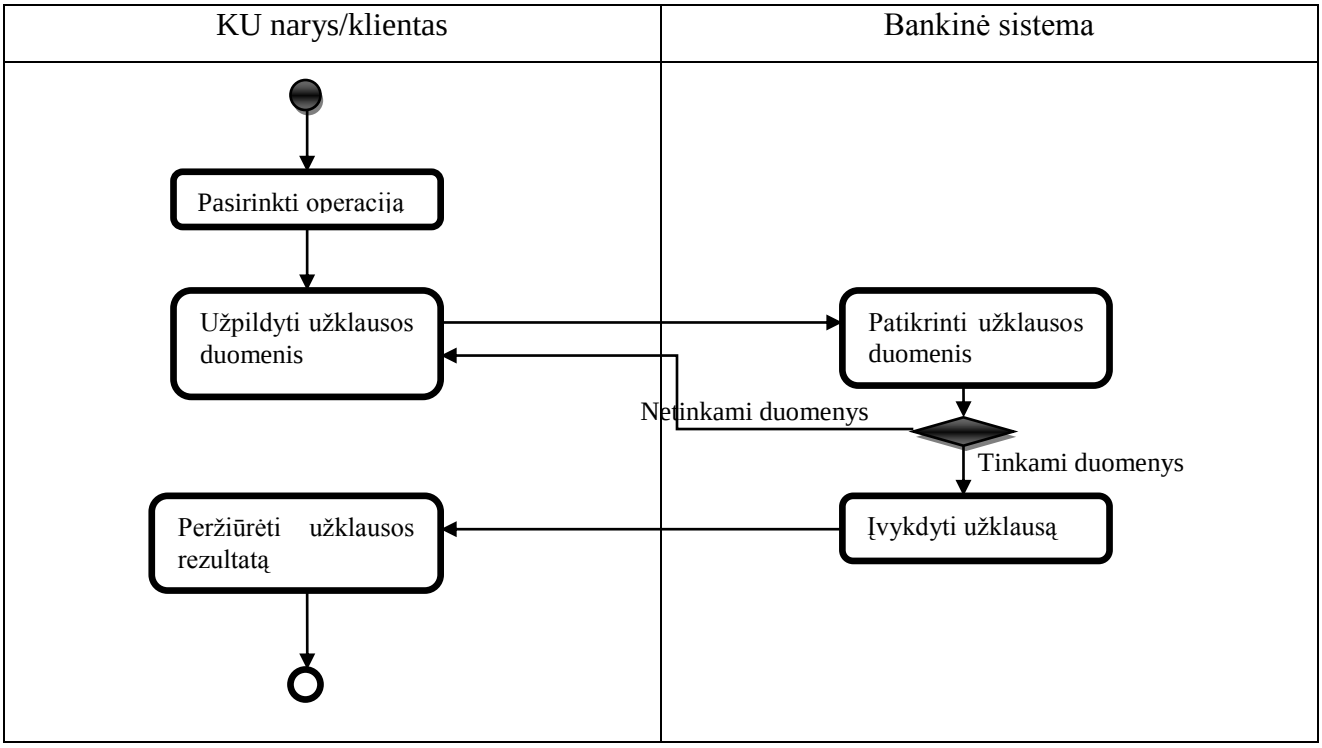


14 pav. Išsaugoti KU darbuotojų suvestą naują įmoką.

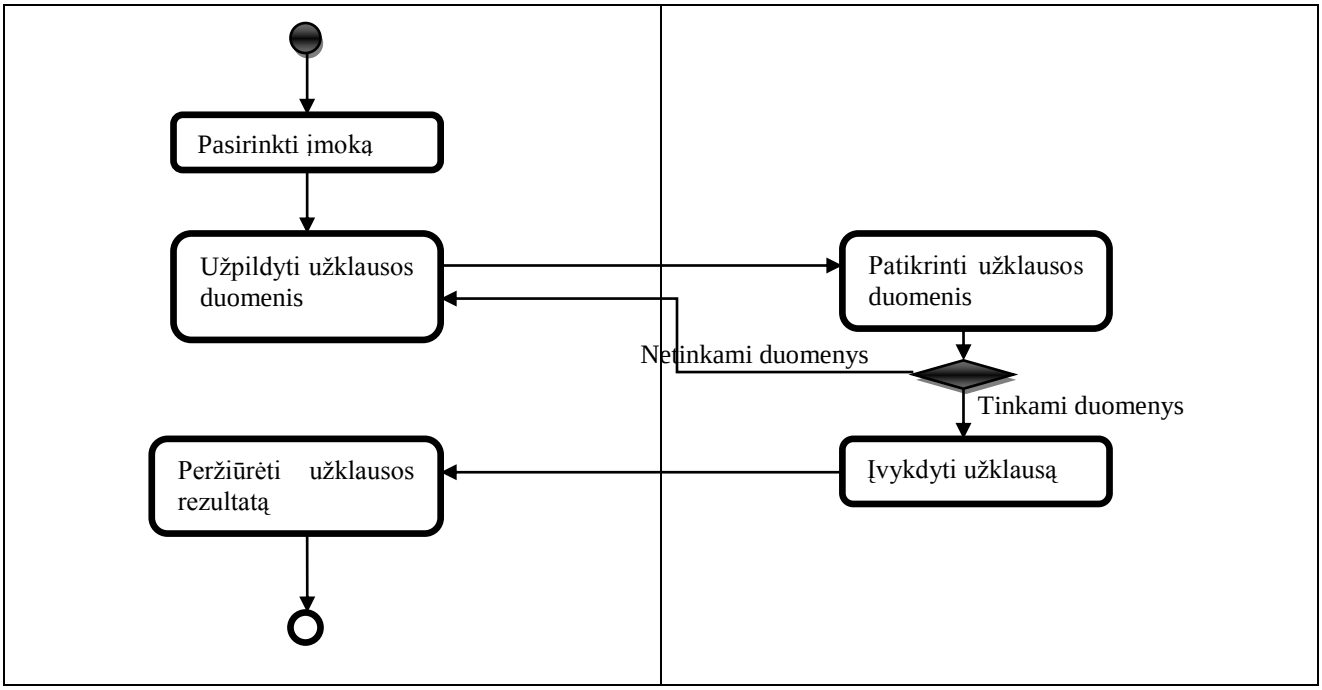




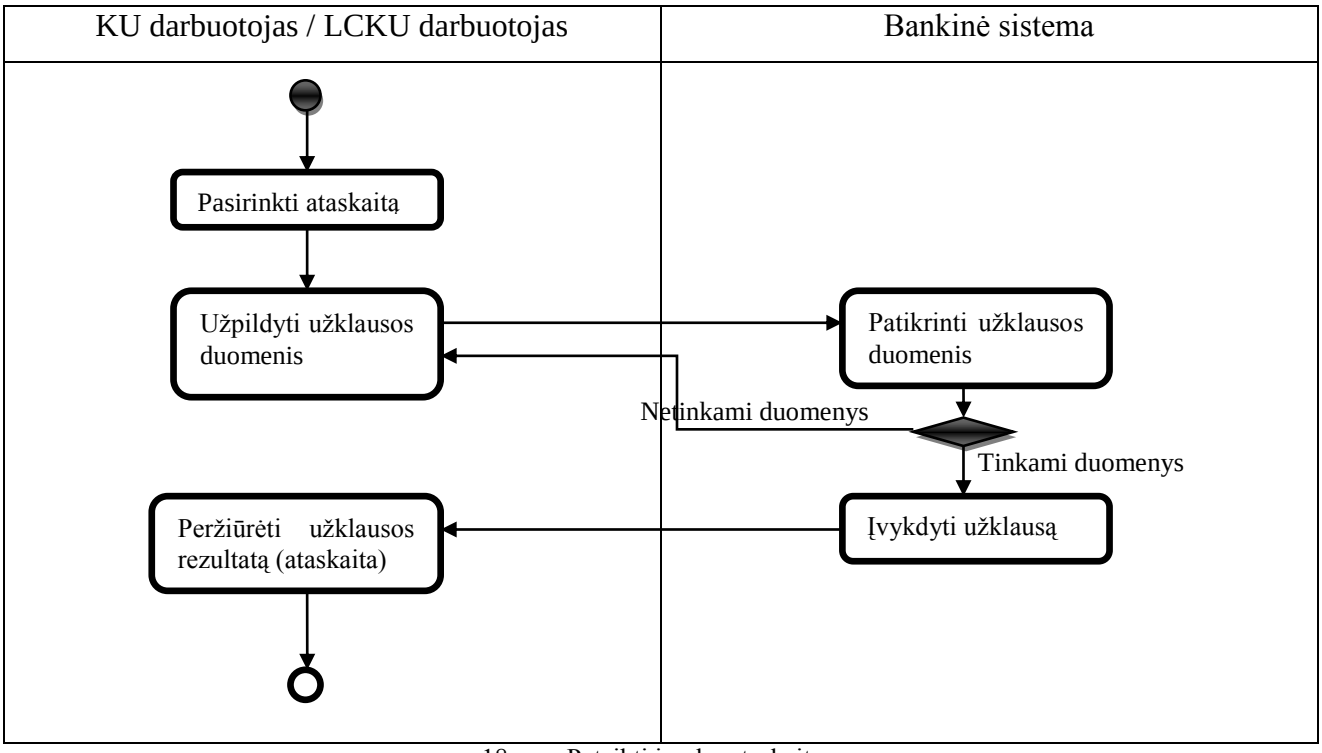
15 pav. Išsiųsti sugeneruotus įmokų failus paslaugų teikėjams.



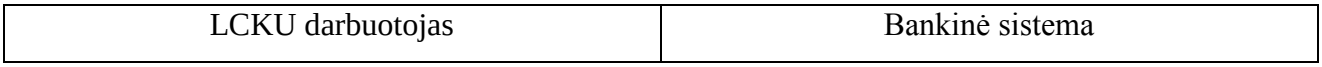
16 pav. Pateikti operacijų išrašą.

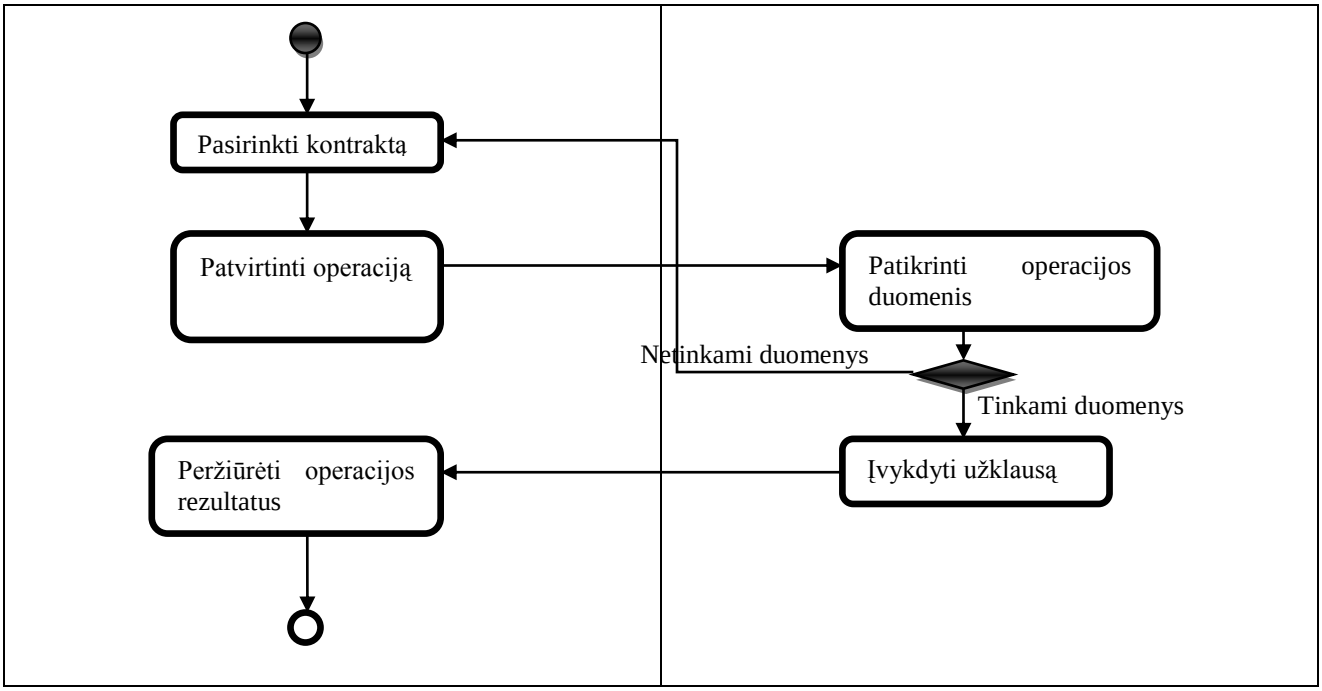


17 pav. Pateikti įmokų archyvą.

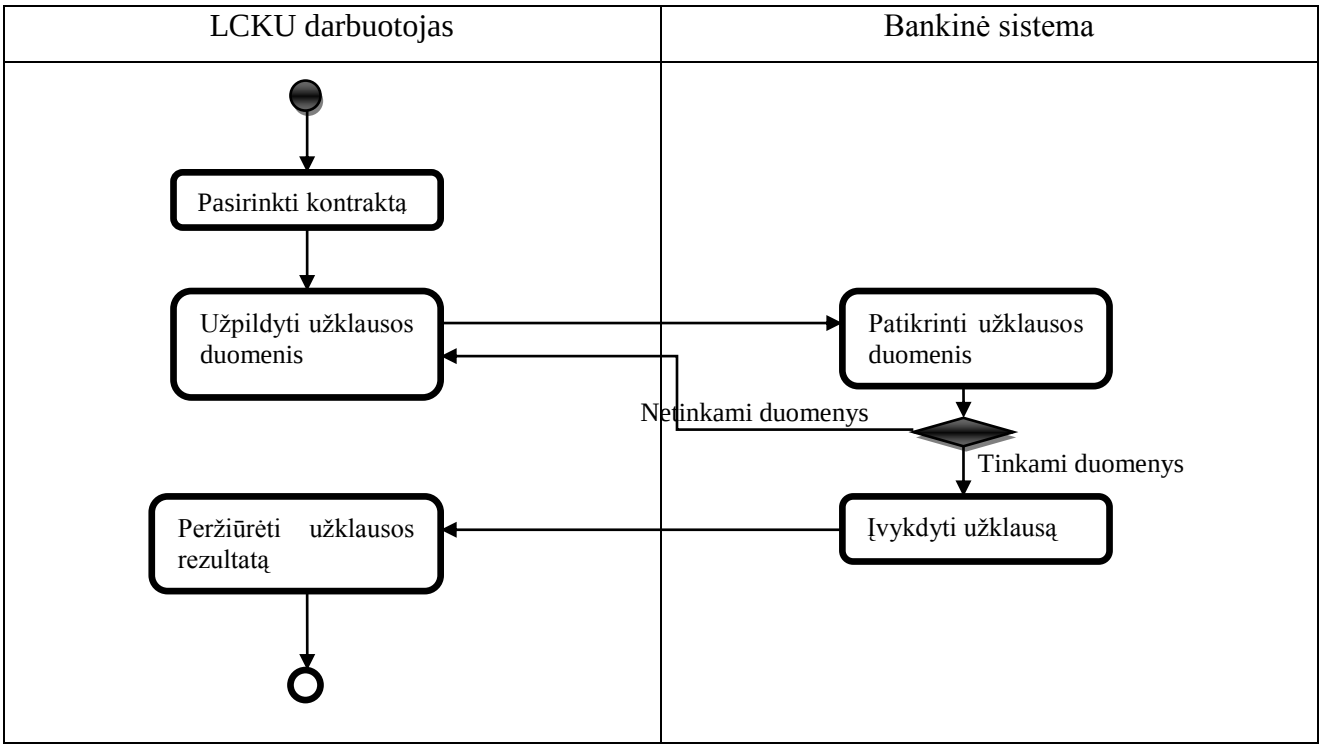


18 pav. Pateikti įmokų ataskaitas.

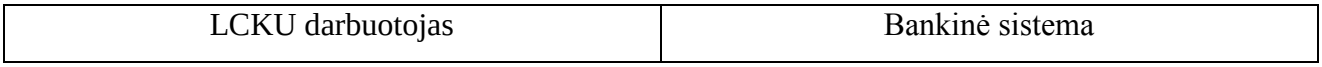


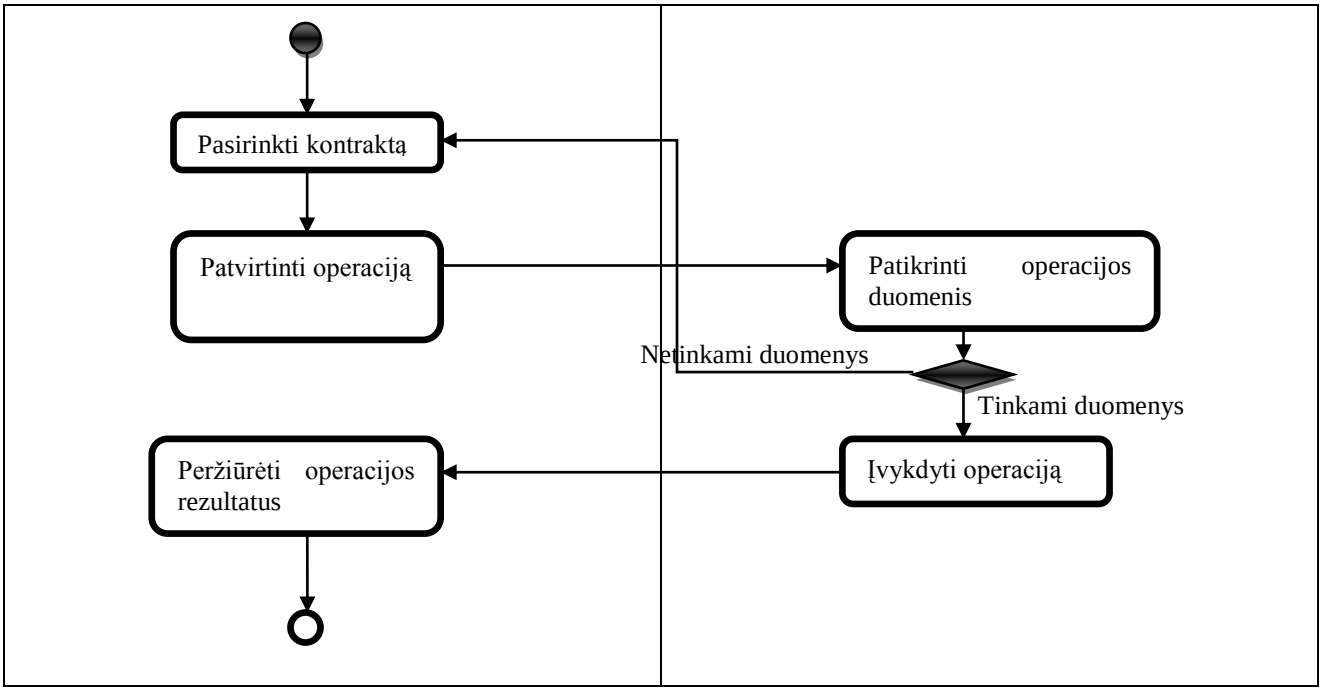


19 pav. Atlikti darbuotojų suvestų įmokų pinigų nurašymą.

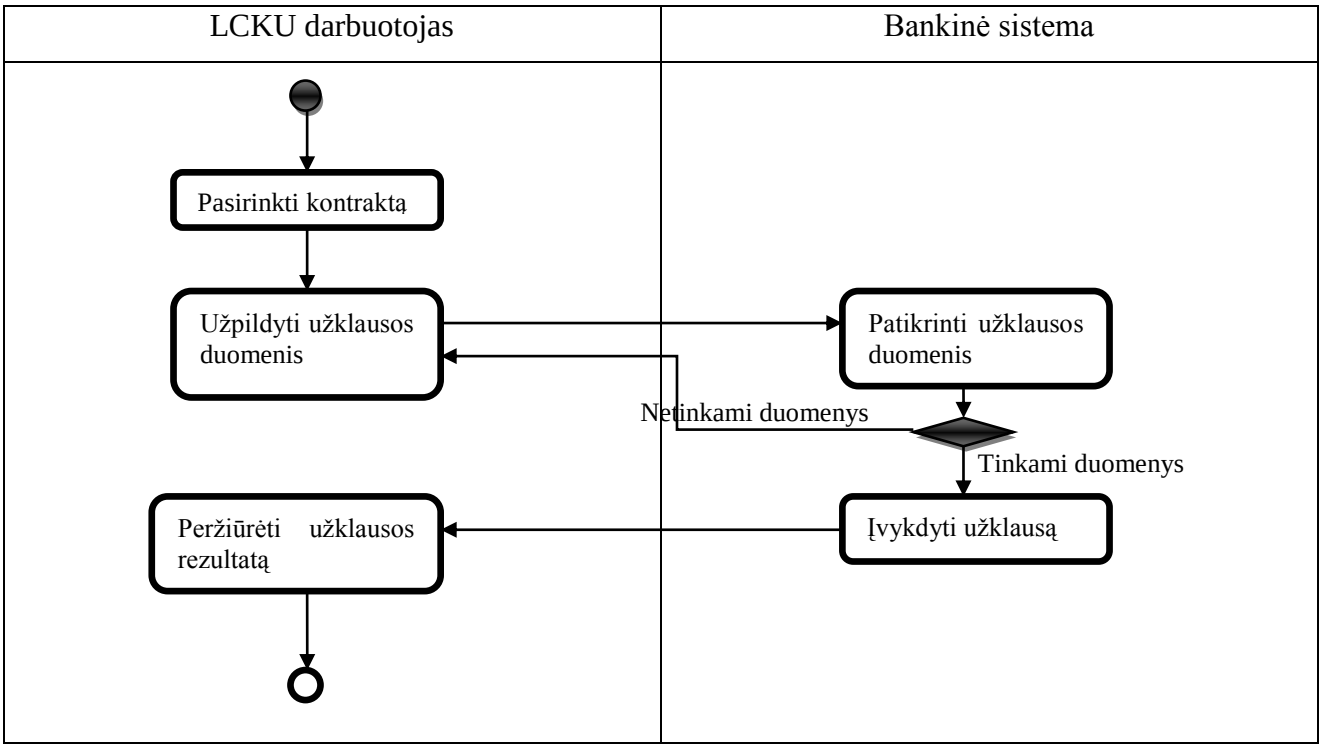


20 pav. Pateikti nurašytų įmokų ataskaitą.

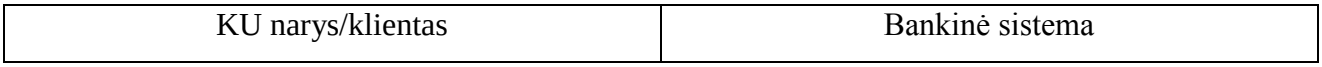


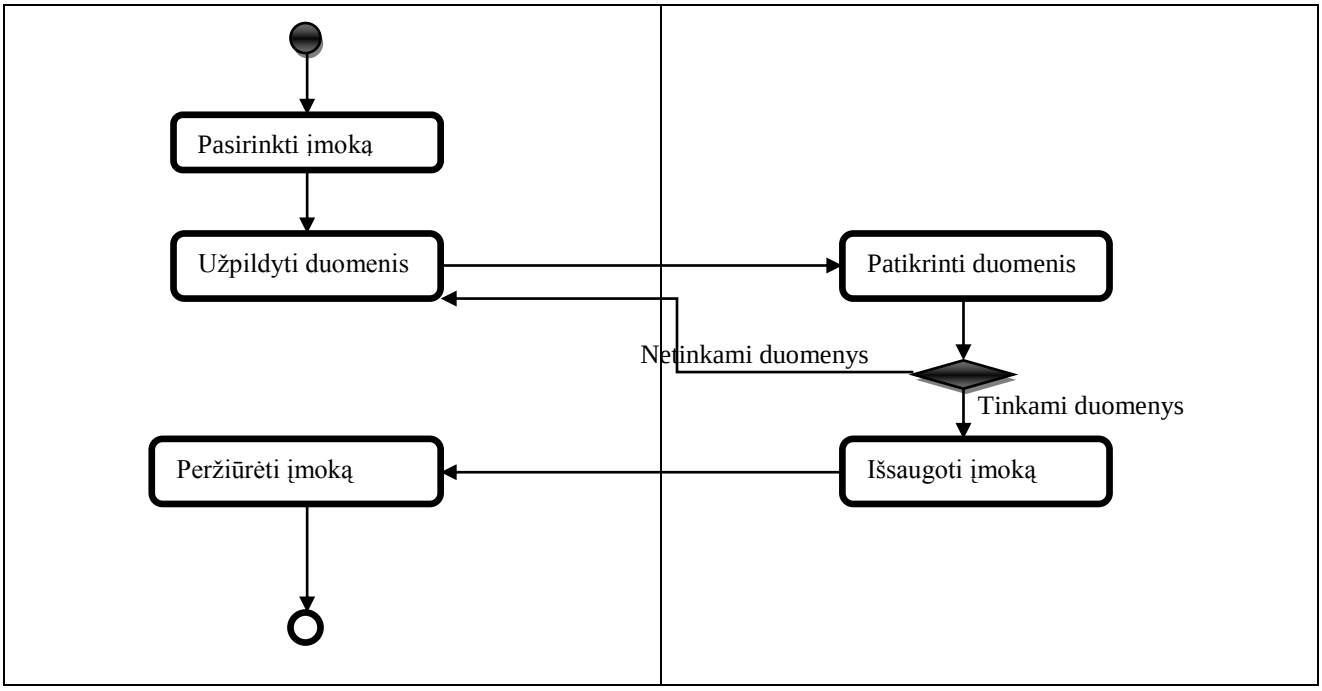


21 pav. Atlikti komisinių paskirstymą į unijų sąskaitas.



22 pav. Pateikti komisinių ataskaitą.

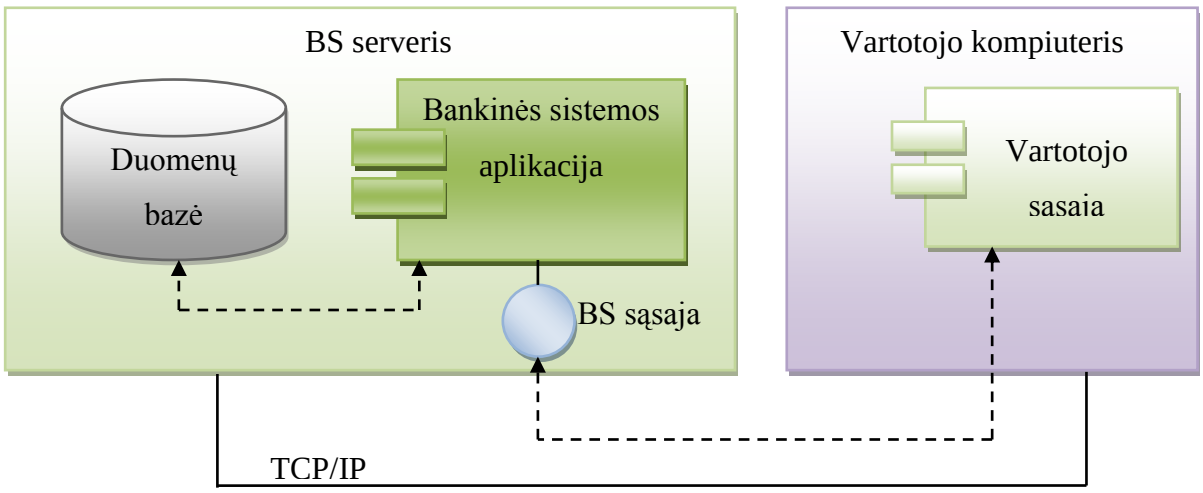




23 pav. Keisti įmokos būseną.

3.4 Išdėstymo vaizdas

Sistema veikia kliento/serverio principu, tai centralizuota sistema. Sistema sudaro duomenų bazė ir bankinės sistemos aplikacija. Klientas arba darbuotojas jungiasi per internetinę naršyklę prie sistemos, kad galėtų naudotis bankinės sistemos paslaugomis. Toliau yra pavaizduotas bendras sistemos ir kliento/darbuotojo išdėstymo vaizdas (24 pav.).



24 pav. Bendras sistemos ir kliento/darbuotojo išdėstymo vaizdas.

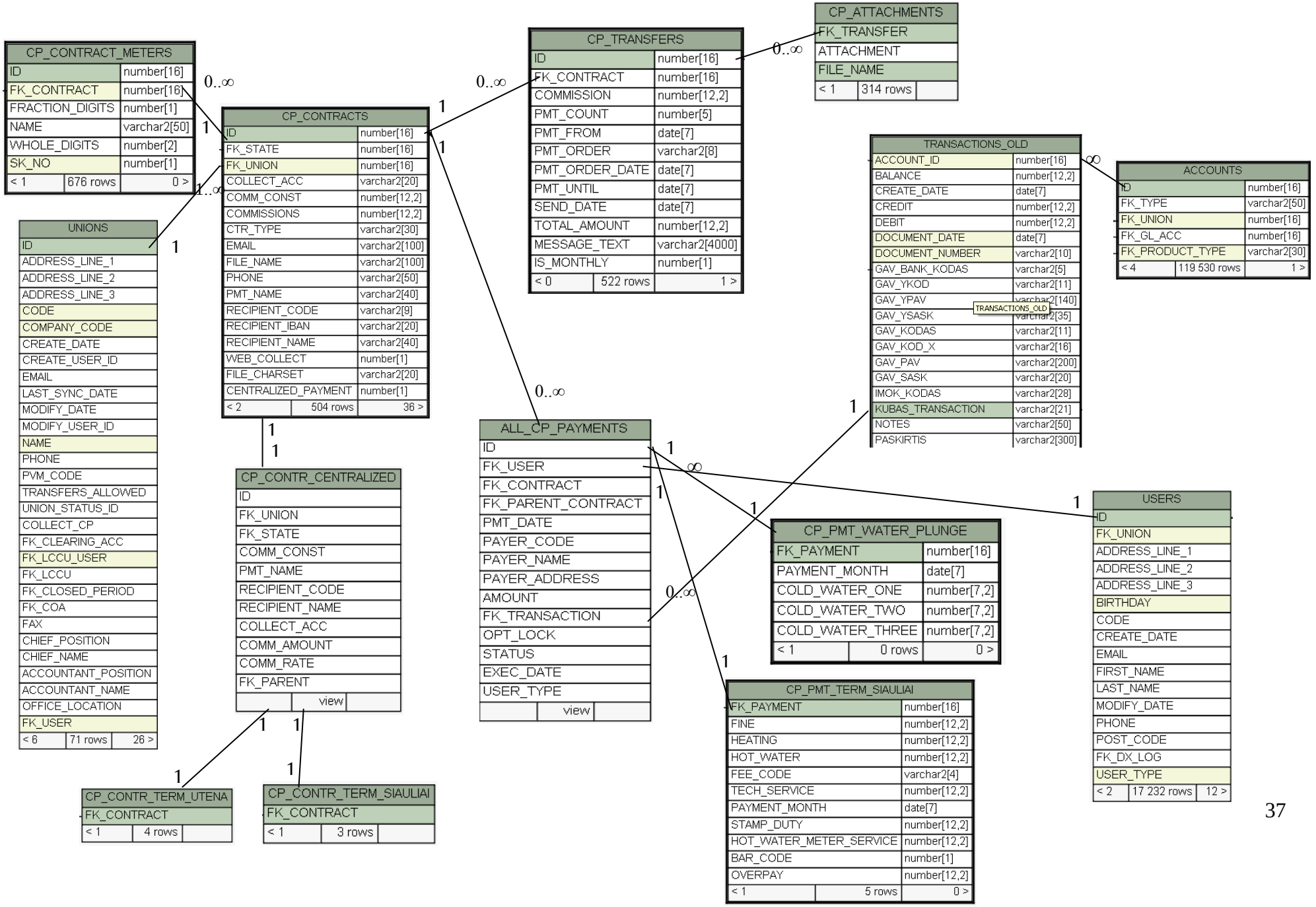
Bankinės sistemos serverio parametrai yra 2 lentelėje. Serveryje veiks *Linux* operacinė sistema. Sistema naudos *Oracle* DBVS.

2 lentelė. BS serverio IBM System x3500 M2 specifikacija [3].

Form factor/height	Tower/5U (rack-mountable)
Processor (max)	Intel® Xeon® X5570 up to 2.93 GHz and up to 8 MB cache
Number of processors (std/max)	1/2
Cache (max)	8 MB per processor socket
Memory (max)	2 GB/128 GB max 1333 MHz DDR-3 registered DIMMs via 16 DIMM slots
Expansion slots	6 PCI-Express, 1 PCI, 2 PCI-X (optional—requires removal of 1 PCI-Express)
Disk bays (total/hot-swap)	16/16 (SFF) (8 standard with additional 8 available)
Maximum internal storage	2.3 TB hot-swap Serial Attached SCSI (SAS)
Network interface	Integrated dual Gigabit Ethernet
Power supply (std/max)	920 W 1/2
Hot-swap components	Power supply, fans and hard disk drives
RAID support	Integrated Hardware RAID-0, -1, -1E, optional RAID-5, -6, -10, -50, -60
Systems management	Automatic Server Restart; IBM Predictive Failure Analysis on hard disk drives, processors, voltage regulator modules (VRMs), fans and memory; light path diagnostics; integrated management module; optional remote presence hardware key; IBM Systems Director and IBM Systems Director Active Energy Manager™
Operating systems supported	Microsoft® Windows®, Red Hat Enterprise Linux®, SUSE Linux Enterprise, VMware ESX and ESXi

3.5 Duomenų vaizdas

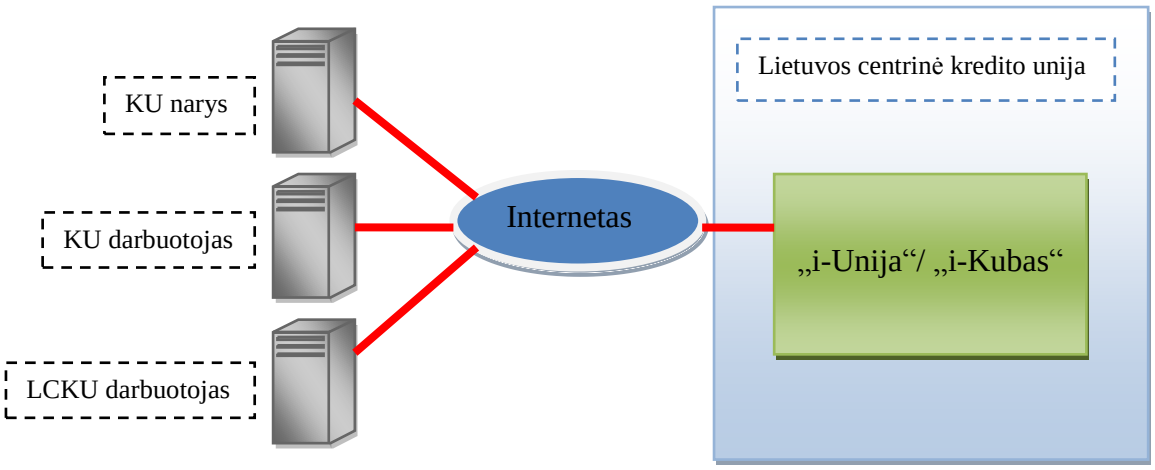
Kadangi tikra duomenų bazė yra labai didelė ir dėl saugumo bei teisinių priežasčių pavaizduosiu tik esmines lenteles susietas su kuriamu moduliu. Čia pateikta tik ribotas kiekis lentelių ir su ne visais laukais dėl kompaktiškumo.


CP_CONTRACT_METERS
ID number[16]
FK_CONTRACT number[16]
FRACTION_DIGITS number[1]
NAME varchar2[50]
WHOLE_DIGITS number[2]
SK_NO number[1]
< 1 676 rows 0 >
CP_CONTRACTS
ID number[16]
FK_STATE number[16]
FK_UNION number[16]
COLLECT_ACC varchar2[20]
COMM_CONST number[12,2]
COMMISSIONS number[12,2]
CTR_TYPE varchar2[30]
EMAIL varchar2[100]
FILE_NAME varchar2[100]
PHONE varchar2[50]
PMT_NAME varchar2[40]
RECIPIENT_CODE varchar2[9]
RECIPIENT_IBAN varchar2[20]
RECIPIENT_NAME varchar2[40]
WEB_COLLECT number[1]
FILE_CHARSET varchar2[20]
CENTRALIZED_PAYMENT number[1]
< 2 504 rows 36 >
CP_TRANSFERS
ID number[16]
FK_CONTRACT number[16]
COMMISSION number[12,2]
PMT_COUNT number[5]
PMT_FROM date[7]
PMT_ORDER varchar2[8]
PMT_ORDER_DATE date[7]
PMT_UNTIL date[7]
SEND_DATE date[7]
TOTAL_AMOUNT number[12,2]
MESSAGE_TEXT varchar2[4000]
IS_MONTHLY number[1]
< 0 522 rows 1 >
CP_ATTACHMENTS
FK_TRANSFER
ATTACHMENT
FILE_NAME
< 1 314 rows
TRANSACTIONS_OLD
ACCOUNT_ID number[16]
BALANCE number[12,2]
CREATE_DATE date[7]
CREDIT number[12,2]
DEBIT number[12,2]
DOCUMENT_DATE date[7]
DOCUMENT_NUMBER varchar2[10]
GAV_BANK_KODAS varchar2[5]
GAV_YKOD varchar2[11]
GAV_YPAV varchar2[140]
GAV_YSASK varchar2[35]
GAV_KODAS varchar2[11]
GAV_KOD_X varchar2[16]
GAV_PAV varchar2[200]
GAV_SASK varchar2[20]
IMOK_KODAS varchar2[28]
KUBAS_TRANSACTION varchar2[21]
NOTES varchar2[50]
PASKIRTIS varchar2[300]
TRANSACTIONS_OLD
ACCOUNTS
ID number[16]
FK_TYPE varchar2[50]
FK_UNION number[16]
FK_GL_ACC number[16]
FK_PRODUCT_TYPE varchar2[30]
< 4 119 530 rows 1 >
UNIONS
ID
ADDRESS_LINE_1
ADDRESS_LINE_2
ADDRESS_LINE_3
CODE
COMPANY_CODE
CREATE_DATE
CREATE_USER_ID
EMAIL
LAST_SYNC_DATE
MODIFY_DATE
MODIFY_USER_ID
NAME
PHONE
PVM_CODE
TRANSFERS_ALLOWED
UNION_STATUS_ID
COLLECT_CP
FK_CLEARING_ACC
FK_LCCU_USER
FK_LCCU
FK_CLOSED_PERIOD
FK_COA
FAX
CHIEF_POSITION
CHIEF_NAME
ACCOUNTANT_POSITION
ACCOUNTANT_NAME
OFFICE_LOCATION
FK_USER
< 6 71 rows 26 >
CP_CONTR_CENTRALIZED
ID
FK_UNION
FK_STATE
COMM_CONST
PMT_NAME
RECIPIENT_CODE
RECIPIENT_NAME
COLLECT_ACC
COMM_AMOUNT
COMM_RATE
FK_PARENT
view
ALL_CP_PAYMENTS
ID
FK_USER
FK_CONTRACT
FK_PARENT_CONTRACT
PMT_DATE
PAYER_CODE
PAYER_NAME
PAYER_ADDRESS
AMOUNT
FK_TRANSACTION
OPT_LOCK
STATUS
EXEC_DATE
USER_TYPE
view
CP_PMT_WATER_PLUNGE
FK_PAYMENT number[16]
PAYMENT_MONTH date[7]
COLD_WATER_ONE number[7,2]
COLD_WATER_TWO number[7,2]
COLD_WATER_THREE number[7,2]
< 1 0 rows 0 >
CP_PMT_TERM_SIAULIAI
FK_PAYMENT number[16]
FINE number[12,2]
HEATING number[12,2]
HOT_WATER number[12,2]
FEE_CODE varchar2[4]
TECH_SERVICE number[12,2]
PAYMENT_MONTH date[7]
STAMP_DUTY number[12,2]
HOT_WATER_METER_SERVICE number[12,2]
BAR_CODE number[1]
OVERPAY number[12,2]
< 1 5 rows 0 >
USERS
ID
FK_UNION
ADDRESS_LINE_1
ADDRESS_LINE_2
ADDRESS_LINE_3
BIRTHDAY
CODE
CREATE_DATE
EMAIL
FIRST_NAME
LAST_NAME
MODIFY_DATE
PHONE
POST_CODE
FK_DX_LOG
USER_TYPE
< 2 17 232 rows 12 >
CP_CONTR_TERM_UTENA
FK_CONTRACT
< 1 4 rows
CP_CONTR_TERM_SIAULIAI
FK_CONTRACT
< 1 3 rows
0..∞
1
∞

4 TYRIMO DALIS

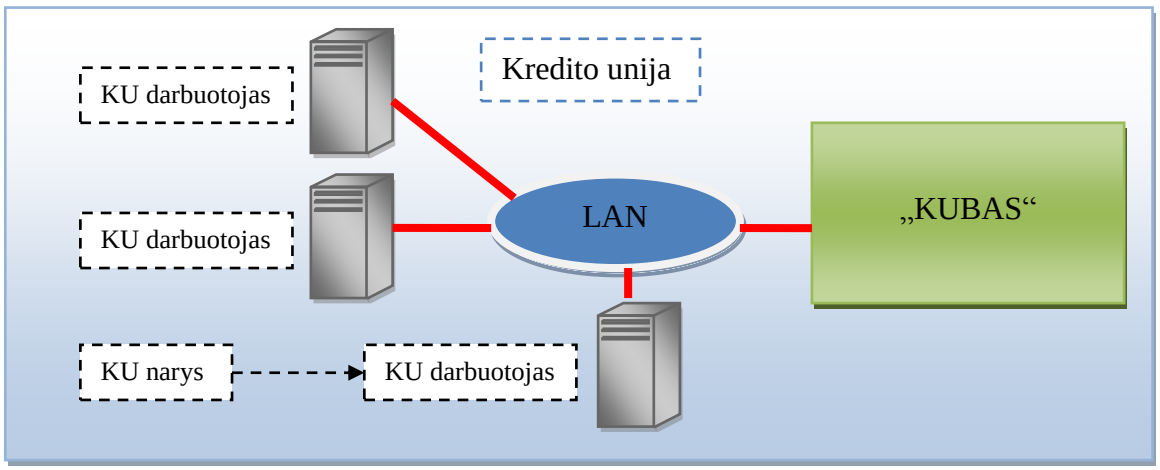
4.1 Liktinės ir naujos sistemos palyginimas

Lietuvos kredito unijos turėjo ir vis dar turi liktinę programą – „KUBAS“. Tai bankinė sistema sukurta xBase++ programavimo kalba. Laikui bėgant kito naudotojų reikalavimai. Atsirandant naujoms technologijoms, programavimo kalbos, naujoms galimybėms atsirado reikalavimas naujai ir tobulesnei sistemai. „i-Kubas“ ir „i-Unija“ naujos bankinės sistemos, kuri vis dar kuriama pavadinimas. Nors ši sistema yra viena tačiau naudotojų reikalavimu ji buvo „atskirta į dvi dalis“, tai darbuotojų dalis („i-Kubas“) ir klientų dalis („i-Unija“). Kadangi „KUBAS“ yra sukurta pagal senas technologijas, todėl jo ir galimybės yra mažesnės. „KUBAS“ taip pat turėjo komunalinių įmokų modulį, tačiau jo galimybės palyginus su naujai sukurtu komunalinių įmokų moduliu yra menkesnės. Tiek funkcionalumu tiek galimybe šia sistema pasinaudoti yra ribotos. Toliau paveikslėliuose yra pateiktas „KUBAS“ ir naujos bankinės sistemos veikimo ir pasiekiamumo vaizdas.



25 pav. „i-Unija“ ir „i-Kubas“ veikimo ir pasiekiamumo vaizdas.

Aukščiau 25 paveikslėlyje pavaizduotas vaizdas kaip naudotojai gali naudotis naujos bankinės sistemos funkcionalumu. Ši bankinė sistema veikia klientas/serveris principu. Bankinės sistemos programa ir duomenų bazė yra išsidėsčiusi Lietuvos centrinėje kredito unijoje. Kredito unijos narys, kredito unijos darbuotojas ir LCKU darbuotojas prie sistemos gali prieiti per internetą (teoriškai iš bet kurios pasaulio vietos). Taip pat kiekvienas asmuo (KU narys, KU darbuotojas, LCKU darbuotojas) gali dirbti su programa bet kada ir bet kiek laiko.



26 pav. „KUBAS“ veikimo ir pasiekiamumo vaizdas.

Priešingai nei nauja bankinė sistema, su „KUBAS“ programine įranga gali dirbti tik tos kredito unijos darbuotojas. Klientas norėdamas susimokėti mokesčius turi ateiti į kredito uniją pas darbuotoją, kuris atlieka mokėjimą už mokesčius. Darbuotojo darbo laikas yra ribotas, todėl darbuotojas gali dirbti su programa tik darbo metu. Darbuotojas gali naudotis „KUBAS“ programine įranga tik ten kur ji yra įdiegta. Priešingai nei „KUBAS“, naudotojas prie bankinės sistemos gali prisijungti su bet kokia naršykle. Esant tokiai galimybei naudotojas gali panaudoti ir telefoną, ir planšetinį kompiuterį, ir asmeninį bei nešiojimą kompiuterį (mini kompiuterį) ir taip gali naudotis bankinės sistemos paslaugomis.

Toliau yra pristatomos silpnosios ir stipriosios abiejų sistemų architektūrų ir komunalinių modulių funkcionalumų stipriosios ir silpnosios savybės.

„KUBAS“ teigiamos savybės:

- Modulio sukūrimo laikas mažas.
- Modulio kodo eilučių kiekis mažesnis.
- Nenaudojamos sudėtingos technologijos.
- Lengviau ištestuoti.
- Naudojama tik viena programavimo kalba.
- Dėl sistemos uždarumo padidėja saugumas.

„KUBAS“ neigiamos savybės:

- Mažas komunalinių įmokų sąrašas už kurį galima atsiskaityti.
- Programa veikia tik Windows operacinėje sistemoje.
- Sistema yra necentralizuota:
    - Siunčiamas failas ir pinigai už komunalines įmokas kiekvienos kredito unijos atskirai.

    - Sistema veikia mažose vietovėse, dažniausiai viename pastate.
- KU nariai negali pasinaudoti šia sistema tiesiogiai.
- Galimos įvairios duomenų suvedimo klaidos. Atlikus apklausą, paaiškėjo jog dauguma darbuotojų po ilgesnio sistemos panaudojimo, duomenis veda netikrindami ar laukų išsidėstymas ar jų pavadinimai yra pakitę. Kaip darbuotojai sako „vedu duomenis kaip robotas, viskas automatiškai“.
- Lėta greitaveika.
- Naudojama *dbf* failų formato sistema duomenų laikymui.
- Grafinės sąsajos spalvos yra varginančios akis.
- Nepatogus sistemos atnaujinimas. Kas kartą reikia sistemą iš naujo įdiegti.

„i-Unija“/„i-Kubas“ teigiamos savybės:

- Patogus sistemos pasiekiamumas (bet kur, bet kada).
- Didelė greitaveika.
- Didelis sąrašas komunalinių įmokų už kurias galima atsiskaityti (virš 112 sutarčių).
- Sistema yra centralizuota:
    - Viena sistema visoms kredito unijoms.
    - Už pinigų pervedimą ir failų siuntimą atsakinga tik LCKU (sutaupoma pinigų pavedimams).
- Patogus sistemos atnaujinimas ir klaidų ištaisymas (naudotojai praktiškai to nepajunta).
- KU nariai bet kada gali pasinaudoti šia sistema.

„i-Unija“/„i-Kubas“ neigiamos savybės:

- Modulio sukūrimo laikas didesnis nei liktinės sistemos.
- Modulio kodo eilučių kiekis didelis.
- Naudojamos sudėtingos technologijos.
- Sunkiau ištestuoti.
- Naudojama daugiau nei viena programavimo kalba.
- Dėl sistemos pasiekiamumo sumažėja saugumas.

Kaip matosi iš teigiamų ir neigiamų savybių naujos ir liktinės sistemos teigiamos savybės yra kitos sistemos neigiamos savybės. Tai tarsi paradoksas, nors nauja sistema turėtų būti geresnė ir tobulesnė už liktinę tačiau šiuo atveju tai yra diskutuojamas klausimas.

4.2 Kokybės vertinimas

Toliau bus pristatytas sukurto komunalinių įmokų modulio kokybės vertinimas, kurį sudarys dvi dalys: apklausos anketos lentelės ir programos išeities kodo metrikų analizė.

4.2.1 Apklausų anketos

Kokybės vertinimo anketos, tai užsakovo požiūris į produkto kokybę ir modulio kūrimo procesą. Toliau yra pateiktos anketos, kuriose prašoma įvertinti nuo 0 iki 10 (10 aukščiausias kokybės įvertinimas).

3 lentelė. Analizės.

Atliktos probleminės srities analizė	**10**
Naudotojo reikalavimų surinkimo procesas	**10**
Reikalavimų fiksavimo kokybė (dokumentacija)	**9**
Kita:	

4 lentelė. Projektavimas.

Projektavimo dokumentacija	**9**
Technologiniai sprendimai	**10**
Projektavimo procesas	**10**
Resursų panaudojimas	**10**
Kita:	

5 lentelė. Įgyvendinimas.

Sistemos įgyvendinimo kokybė	**10**
Sistemos grafinė sąsaja	**10**
Sistemos greitaveika	**8**
Išeities kodų kokybė	**9**
Standartų įgyvendinimas	**9**
Sistemos bendradarbiavimas tarp kitų sistemų išpildymas	**9**
Kita:	

6 lentelė. Testavimas.

Testavimo rezultatai	**10**
Naudojamos testavimui technologijos	**10**
Naudojami testavimo metodai	**10**
Kita:	

7 lentelė. Įdiegimas.

Modulio integracija į bendrą sistemą	**10**
Sistemos įdiegimo paprastumas	**10**
Sistemos įdiegimo proceso laikas	**10**
Sistemos įdiegimo procesas	**10**
Kita:	

Kaip matosi iš apklausos rezultatų užsakovas projekto kokybę įvertino labai gerai. Pagrindinė problema kurioje užsakovas vertino mažiau nei 10 tai dokumentacija. Kadangi reikalavimai dažnai keisdavosi, todėl dokumentacijos kokybė truputį nukentėjo. Įgyvendinime sistemos greitaveika gavo mažiausiai balų, kadangi naudojama techninė įranga nepajėgi užtikrinti geros greitaveikos. Dėl didėjančio duomenų kiekio sistemos greitaveika gali dar labiau sumažėti ateityje. Būtina paskirti techniniai daliai daugiau lėšų, bei patikrinti duomenų apdorojimo kodą, kur galima optimizuoti bei sumažinti resursų panaudojimą. Testavimo procesą ir jo rezultatus užsakovas įvertino puikiai. Įdiegimo procesą kaip ir testavimo, užsakovas įvertino puikiai.

4.3 Programos išeities kodo metrikų analizė

Programos išeities kodo metrikų analizei buvo pasirinkta SourceMonitor (daugiau informacijos 1 priede). Kadangi tikrinamas kodas yra Java programavimo kalbos todėl šios kalbos kodo metrikos yra tikrinamos tokios:

- Kodo eilutės.
- Išraiškos (angl. Statement).
- Procentinės šakų išraiškos (angl. Percent Branch Statements).
- Metodų iškvietimai išraiškose.
- Procentinės komentarų eilutės.
- Klasės ir sąsajos.
- Metodai per klasę.
- Vidurkis išraišku per metodą.
- Maksimalus metodo sudėtingumas.
- Maksimalus blokų gylis.
- Blokų gylio vidurkis.

- Sudėtingumo vidurkis

Toliau yra pateikta lentelė, kurioje yra bendra metrikų suma gauta naudojantis SourceMonitor programa.

8 lentelė. Modulio metrikų duomenys.

Metrikos pavadinimas	Reikšmė
Failų skaičius	1655
Kodo eilučių skaičius	131114
Išraiškos	73085
Šakų %	10,3
Kvietimai	51100
Komentarai %	8,7
Klasės	1589
Metodai per klasę	5,05
Vidut. išraiškų skaičius metode	5,49
Maksimalus sudėtingumas	138
Maksimalus gylis	9+
Vidutinis gylis	1,92
Vidutinis sudėtingumas	2,32

Modulio metrikų lentelėje esantys duomenys gana daug pasako apie kodo kokybę, taip pat perspėja apie galimas kodo vietas, kurias reikėtų peržiūrėti dėl kodo optimizavimo. SourceMonitor esantis maksimalus sudėtingumas pasako ar tas programos kodas (metodas, klasė, kodo blokas) yra lengvai ištestuojamas, sunkiai testuojamas ar apskritai yra netestuojamas. Kuo mažesnis sudėtingumo skaičius tuo programa yra lengviau testuojama ir prižiūrima.

Pagal pateiktus 8 lentelėje duomenis galima pasakyti, kad kodas yra mažai komentuotas – tik 8,7% viso projekto. Mažai komentuotas kodas apsunkina programos priežiūrą. Ateičiai patartina labiau komentuoti kodą. Klasių kiekis yra didelis, tačiau metodų per klasę procentas yra mažas – 5,05%. Toks skaičius pasako, kad klasės yra paprastos ir nėra perkrautos daugybės metodų. Maksimalus sudėtingumo skaičius (ciklomatinis sudėtingumas) nurodo kodo bloko (metodo) sudėtingumo lygį. Didėjant ciklomatiniam sudėtingumo skaičiui, sudėtingėja kodo ištestavimas. Šiame modulyje yra vienas metodas, kuris turi pačią didžiausią ciklomatinę reikšmę – 138. Pagal 9 lentelėje pateiktoje reikšmių vertinimą, šis programos kodas yra „netestuojamas“.

9 lentelė. Ciklomatinio sudėtingumo reikšmės vertinimas.

CC reikšmė	Programos įvertinimas
1 - 10	paprasta, klaidų tikimybė maža
11 - 20	vidurinio sudėtingumo, klaidos tikimybė vidutinė
21 - 50	sudėtinga, didelė klaidos tikimybė
> 50	netestuojama

Remiantis 9 lentelės reikšmių vertinimais komunalinių įmokų modulyje yra tik 6 klasės kurios viršija 50 reikšmę ir yra netestuojamos. 90 klasių atitinka ribas [21-50], kurios yra sudėtingos ir galimos didelės klaidų tikimybės. Tokias klases būtina išskaidyti į mažesnes. 161 klasė atitinka ribas nuo [11-20]. Šios klasės pagal 9 lentelėje pateiktą įvertinimą yra vidutinio sudėtingumo ir klaidos tikimybė vidutinė. Visos kitos likusios klasės yra žemiau 10 ribos – tai paprastos klasės, klaidų tikimybė jose yra maža. Metrikų modulio duomenų vaizdą galima pamatyti 2 priede. Čia galima taip pat pamatyti ir „Kiviat“ grafikas. Tai išrinktų metrikų grafikas, kurie atvaizduojami apskritime.

SourceMonitor negali ištestuoti xBase++ programavimo kalba parašyto kodo, todėl nebuvo galima patikrinti „KUBAS“ kokybės ir įvertinti sudėtingumo lygio.

4.4 Siūlomi tolimesni programos tobulinimai

Nors modulis atitinka visus užsakovo reikalavimus, tačiau visada yra daug vietų kur galima patobulinti. Toliau pristatysiu mano nuomone kur būtų galima ką patobulinti.

4.4.1 Nauji funkcionalumai

Moduliu būtų galima efektyviau atlikti darbus jei būtų automatizuotas:

- įmokų failų siuntimas;
- pinigų pervedimas komunalininkams;
- komisinių paskirstymas kredito unijoms.

Būtų gerai jog būtų galima atsiskaityti už bet kurias komunalines paslaugas esančias Lietuvoje.

4.4.2 Grafinė sąsaja

Būtina patobulinti bankinės sistemos dizainą. Būtų gerai jog sąsaja būtų pritaikyta jungiantis prie bankinės sistemos tiek telefonu tiek planšetiniu kompiuteriu. Pildant įmokos duomenis bar kodu, automatiškai po duomenų užpildymo paspaustų mygtuką patvirtinantį duomenis.

4.4.3 Nefunkciniai reikalavimai

Būtinas restruktūrizavimas duomenų lentelių susijusių su komunalinėmis įmokomis. Paspartinti greitaveiką, pritaikant įvairiomis kodo optimizavimo taisyklėmis, programinę įranga, kompiuterine technika, naujomis informacinėmis technologijomis. Ten kur puslapis užkraunamas ilgiau nei per 4 sekundes, perspėti klientą, kad užklausa gali vykdoma ir teks palaukti, arba optimizuoti tą vietą, kad puslapis greičiau užsikrautų. Šiame projekte naudojama *hibernate* technologija dėl savo savybių kartais sumažina programos greitaveiką. Būtina šia vietą optimizuoti.

5 IŠVADOS

- Sukurtas bankinės sistemos komunalinių įmokų modulis bankinei sistemai „i-Unija“/„i-Kubas“. Modulis sėkmingai integruotas į sistemą. Komunalinių įmokų moduliu naudojasi visi kredito unijų nariai, darbuotojai.
- Užsakovo reikalavimams surinkti ir jų kokybei užtikrinti buvo pasirinktas *Volere* šablonas.
- Visi užsakovo reikalavimai buvo įgyvendinti. Norint prisijungti prie bankinės sistemos ją galima rasti internetiniu adresu < https://www.i-unija.lt/login.htm >
- Atlikus bankinės sistemos analizę buvo prisitaikyta prie N-sluoksnių architektūros, panaudota *hibernate* technologija ir *spring* karkasas leidžiant greitai suprogramuoti norimą funkcionalumą. Modulis buvo sukurtas laiku.
- Tyrimo metu buvo pastebėta jog *hibernate* technologija dėl savo specifikos kartais sulėtina sistemos greitaveiką, todėl būtina tas vietas optimizuoti pakeičiant *SQL* užklausomis.
- Tiriant lyginamuoju principu liktinės sistemos veikimą ir pasiekiamumą su nauja bankine sistema pastebėta, kad liktinės sistemos stipriosios savybė patapo silpnosiomis naujos sistemos savybėmis. Panašus efektas atsitiko ir su silpnosiomis liktinės sistemos savybėmis. Silpnosios liktinės sistemos savybės atitiko stipriąsias naujos sistemos savybes.
- Užsakovams buvo pateikta programos ir proceso kūrimo kokybės vertinimo apklausa. Rezultatas buvo labai geras, nes užsakovas buvo praktiškai viskuo patenkintas.

6 LITERATŪRA

[1] LIETUVOS RESPUBLIKOS FINANSŲ ĮSTAIGŲ ĮSTATYMAS 2002 m. rugsėjo10 d. Nr. IX-1068 PIRMASIS SKIRSNIS BENDROSIOS NUOSTATOS. Prieiga per internetą: <http://www3.lrs.lt/pls/inter3/dokpaieska.showdoc_l?p_id=245489>. [žiūrėta 2009-10-24]

[2] Tong Ka Iok. *Essential Skills for Agile Development* [interaktyvus], 2004 birželis. [žiūrėta 2009-09-10]

[3] Deepak Alur, John Crupi, Dan Malks. *Core J2EE™ Patterns: Best Practices and Design Strategies*. Sun Microsystems Press A Prentice Hall Title. Prentice Hall, ISBN 0-13-064884-1, 2001. [žiūrėta 2009-11-01]

[4] *How PGP works*. Prieaiga per internetą:
< http://www.pgpi.org/doc/pgpintro/ >. [žiūrėta 2009-11-02]

[5] *An Introduction to Cryptography*. PGP*, Version 6.5.1, Copyright © 1990-1999 Network Associates, Inc. and its Affiliated Companies. Prieaiga per internetą:
< ftp://ftp.pgpi.org/pub/pgp/6.5/docs/english/IntroToCrypto.pdf>. [žiūrėta 2009-11-02]

[6] Tim Lindholm,Frank Yellin. The *JavaTM Virtual Machine Specification: Second Edition*. Prieaiga per internet:
< http://java.sun.com/docs/books/jvms/second_edition/html/VMSpecTOC.doc.html >.[žiūrėta 2009-10-17]

[7] *.NET Framework Conceptual Overview*. Prieaiga per internetą:
< http://msdn.microsoft.com/en-us/library/zw4w595w.aspx >. [žiūrėta 2009-10-19]

[8] Christian Bauer, Gavin King. *Java Persistence with Hibernate Second Edition of Hibernate in Actio*. ISBN: 1-932394-88-5, November, 2006. [žiūrėta 2009-10-01]

[9] Craig Walls, Ryan Breidenbach. *Spring in Action*. ISBN: 1932394354, 2005. [žiūrėta 2009-09 ir 2009-11]

[10] Henry Dauderis. *Finansų apskaita, Kaip pasirinkti sprendimą*. Pirmas tomas, Pasaulio lietuvių kultūros, mokslo ir švietimo centras, 1993. [žiūrėta 2009-09-10]

[11] *iBATIS Data Mapper*. Prieaiga per internet:
< http://ibatis.apache.org/ >. [žiūrėta 2009-10-25]

[12] *Hibernate* technologija (projektas) ir visa dokumentacija bei bibliotekos susiją su ja. Prieaiga per internet:
< https://www.hibernate.org/ >. [žiūrėta 2009-09-15]

[13] James Gosling, Bill Joy, Guy Steele, Gilad Bracha. *The Java Language Specification, Third Edition*. Prentice Hall, June 14, 2005. Prieaiga per internetą:

< http://java.sun.com/docs/books/jls/third_edition/html/j3TOC.html >. [žiūrėta nuo 2009-09-10]

[14] *C++ Reference*. Prieaiga per internet:

< http://www.cppreference.com/wiki/start >. [žiūrėta 2009-09-10]

[15] Robert Chartier. *Application Architecture: An N-Tier Approach - Part 1*. Prieaiga per internetą:

< http://www.15seconds.com/issue/011023.htm >. [žiūrėta 2009-09-06]

[16] Channu Kambalyal. *3-Tier Architecture*. Prieaiga pe rinternetą:

< http://channukambalyal.tripod.com/NTierArchitecture.pdf >

[17] Ben Collins-Sussman, Brian W. Fitzpatrick, C. Michael Pilato. *Version Control with Subversion. For Subversion 1.5 (Compiled from r3305)*, 2004. Prieaiga per internet:

< http://svnbook.red-bean.com/en/1.5/svn-book.pdf >. [žiūrėta 2009-10-01]

[18] *lambdaj:manipulate collections in a pseudo-functional and statically typed way*. Prieaiga per internet:

< http://code.google.com/p/lambdaj/ >. [žiūrėta 2009-11-21]

[19] *jQuery*. Prieaiga per internetą:

< http://api.jquery.com/category/ajax/ >. [žiūrėta 2009-11-30]

[20] *JavaServer Pages Technology*. Prieaiga per internetą:

< http://java.sun.com/products/jsp/ >. [žiūrėta 2009-12-02]

[21] *Java Servlet Technology*. Prieaiga per internet:

< http://java.sun.com/products/servlet/ >. [žiūrėta 2009-12-05]

[22] *Applets*. Prieaiga per internet:

< http://java.sun.com/applets/ >. [žiūrėta 2009-12-05]

7 TERMINŲ IR SANTRUMPŲ ŽODYNAS

MVC (angl. Model View Controller) – modelis-vaizdas-valdiklis šablonas

PGP (angl. Pretty Good Privacy) – gana geras privatumas.

JVM (Java Virtual Machine) – Java virtuali mašina.

API (angl. Application Programming Interface) – taikomųjų programų programavimo sąsaja.

JDBC (angl. Java Database Connectivity) – tai Java programavimo kalbos taikomųjų programų programavimo sąsaja, kuri apibrėžia kaip klientas gali prisijungti (prieiti) prie duomenų bazės.

IoC (angl. Inversion of Control) – valdiklio inversija.

Front Controller – išorinis valdiklis.

EJB (angl. Enterprise JavaBeans) – EJB technologija yra serverio pusės komponentės architektūra Java platformai.

Framework – platforma, pagrindas, skeletas.

HTML (angl. Hyper Text Markup Language) - hiperteksto žymėjimo kalba.

8 PRIEDAI

1 priedas. Tyrimui naudojamos programinės įrangos aprašymas

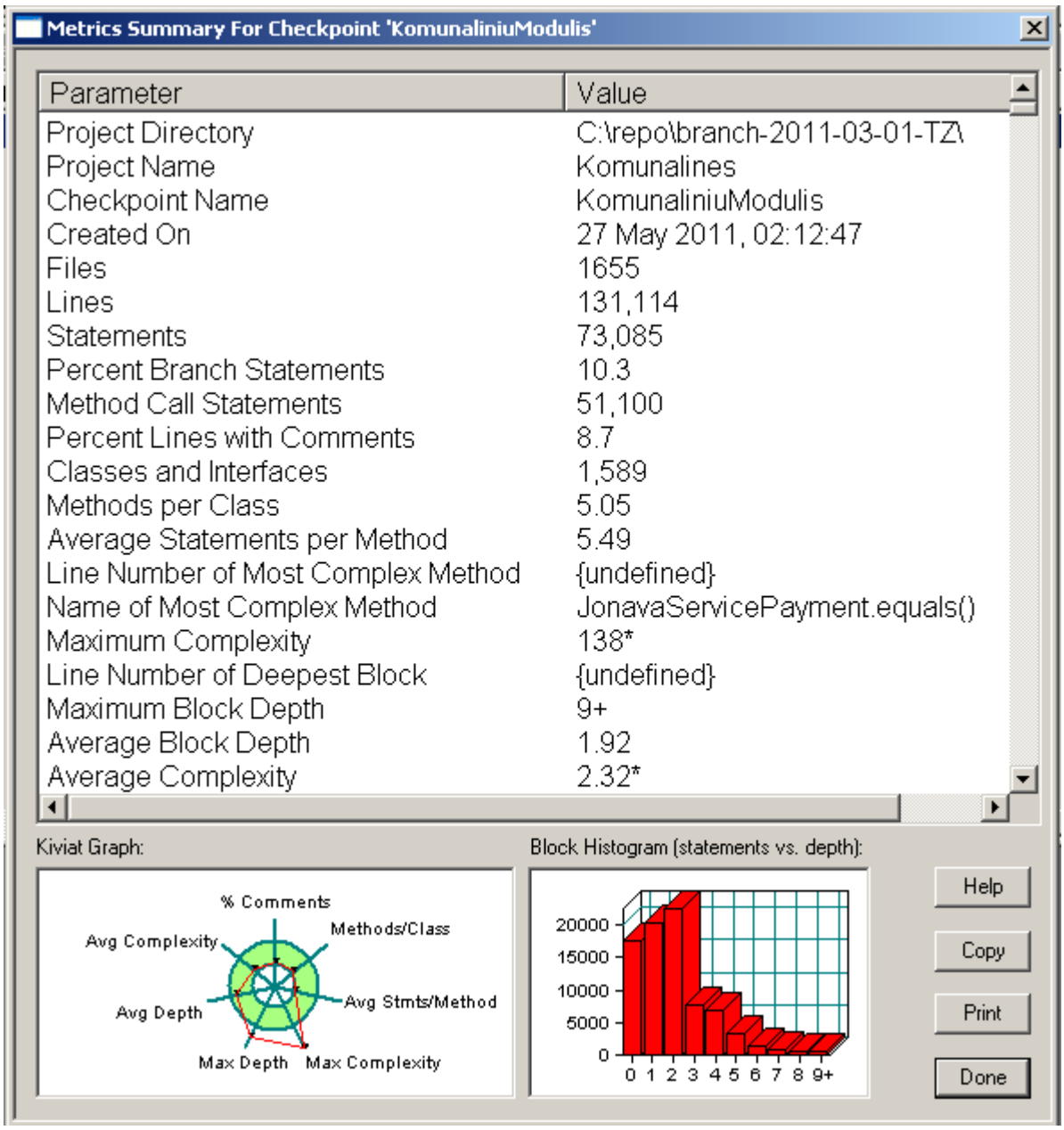
Programos kodo metrikų tyrimui buvo pasirinkta SourceMonitor programa. Tai nemokama programa, kuri pažvelgia į jūsų programinės įrangos išeities kodą ir sužino kiek kodo yra joje, bei nustato santykinį jūsų modulio sudėtingumą. SourceMonitor programą galima naudoti patikrinti jūsų kodui, kuris yra labiausiai tikėtina gali turėti defektų, ir todėl pateisina/papildo oficialią peržiūrą. SourceMonitor yra parašytas C++ programavimo kalba, todėl kodo nuskaitymas vyksta labai greitai (bent 10.000 eilučių kodo per sekundę).

SourceMonitor turi tokia funkcijas:

- Greitai surenka analizuojamo kodo metrikus.
- Galimybė išmatuoti programos kodą parašytą C++, C, C#, VB.NET, Java, Delphi, Visual Basic (VB6) arba HTML kalbomis.
- Apima metodo ir funkcijos lygio metrikus programoms parašytomis C++, C, C#, VB.NET, Java ir Delphi programavimo kalbomis.
- Siūlymas Modifikuoto sudėtingumo metrikų variantas.
- Išsaugoja surinktus metrikus, kad vėliau būtų galima juos sulyginti projekto kūrimo metu.
- Rodo ir spausdina metrikus kaip lenteles ir diagramas, įskaitant Kiviat diagramas.
- Veikia per standartinėje *Windows* grafinėje naudotojo sąsajoje (angl. GUI) arba naudotojo nurodytuose skriptuose naudojant XML komandos failus.
- Eksportuoja metrikus į XML arba CSV (kableliu atskirtos reikšmės) failus tolimesniam procesui su kitais įrankiais.

Programą galima rasti šiuo adresu <http://www.campwoodsw.com/sourcemonitor.html>.

2 priedas. SourceMonitor programos langas ir komunalinių įmokų modulio metrikų santrumpa.



27 pav. Komunalinių įmokų modulio SourceMonitor metrikų santrumpos langas.