ŠIAULIŲ UNIVERSITETAS

Informatikos, matematikos ir e. studijų institutas

Kompiuterių sistemų katedra

Ligita Norkutė

Programinės įrangos kainos skaičiavimo modelio sudarymas ir tyrimas

Magistro baigiamasis darbas

Vadovė doc. dr. A. Slotkienė

Šiauliai, 2015

ŠIAULIŲ UNIVERSITETAS

Informatikos, matematikos ir e. studijų institutas

Kompiuterių sistemų katedra

TVIRTINU
Kompiuterių sistemų katedros vedėjas
doc. dr. E. Paliulis
2015-06-01

Programinės įrangos kainos skaičiavimo modelio sudarymas ir tyrimas

Informatikos inžinerijos magistro baigiamasis darbas

Vadovė
Kompiuterių sistemų katedros docentė Doc.dr. A. Slotkienė
2015 m. birželio 1 d.

Recenzentė
Kompiuterių sistemų katedros lektorė Lekt.dr. A. Drukteinienė
2015 m. birželio ___ d.

Recenzentas
Kompiuterių sistemų katedros docentas Doc.dr. E. Paliulis
2015 m. birželio ___ d.

Autorė
ITM-13 gr. studentė L. Norkutė
2015 m. birželio 1 d.

Šiauliai, 2015

DARBO SANTRAUKA

Šio darbo tikslas sukurti programinės įrangos kainos skaičiavimo metodą. Tam kad tai atlikti buvo išanalizuoti programinės įrangos dydžio apimties vertinimo metodai. Analizuojami buvo funkcinių taškų metodas, panaudojimo atvejų taškų metodas ir objektų taškų metodas.

Darbe pateikta informacija apie programinės įrangos projektus, jų tipus, klasifikaciją. Taip pat pateikta informacija apie programinės įrangos kūrimo procesą.

Darbe buvo išanalizuoti ir ištirti nagrinėjami metodai. Buvo analizuoti kitų autorių darbai. Taip pat buvo atliktas praktinis metodų tyrimas. Buvo pasirinkta informacinė sistema ir ji buvo įvertinta tyrinėjamais metodais.

Atlikus tyrimus ir įvertinus metodų pliusus ir minusus buvo pasiūlytas naujas programinės įrangos kainos skaičiavimo metodas. Pasiūlytas programinės įrangos funkcionalumo įvertinimo metodas įvertina funkcijų sudėtingumą, žmogiškuosius faktorius ir sistemos chrakteristikas.

Raktiniai žodžiai: programinės įrangos kaina, panaudos atvejų taškų metodas, funkcinių taškų metodas, objektų taškų metodas.

SUMMARY

Model Development And Research Software Cost Calculation

This is a work about software cost calculation. Here you can find information on the projects, their types and classification. Here you can find information about software development process. This work examines the most popular software cost estimation methods.

In this paper describes the software development process peculiarities. Here it describes the projects requirements analysis, design, programming and testing. It also describes the life cycle models such as the waterfall model and spiral model.

This paper analyzes the size scale of assessment models. Analyzed use case method, function point method and object point method. It was carried out the analysis of these methods.

It was a practical method of investigation. This was achieved by analyzing e-shop system. Were analyzed peculiarities of the method and disadvantages.

The analysis developed new software cost estimation method.

Keywords: Software cost estimation, use case points method, function points method, object points method.

TURINYS

ĮVADAS

Yra labai svarbu žinoti, ar projektas vyksta pagal planą. Yra sakoma, kad negalima kontroliuoti to, ko negalima pamatuoti. Kadangi viso programinės įrangos gyvavimo ciklo metu yra labai sunku numatyti projekto būseną, reikia organizacijoje įgyvendinti ir visiems projektams taikyti esminių matų matavimo programą. Šie esminiai matai leis palyginti projektų esamą būseną su planuotąja ir įgalins pirmalaikį jų būsenų neatitikimų nustatymą. Matai pašalina netikėtumus informuodami mus, kai projektas artėja prie tam tikrų problemų dar prieš joms įvykstant [7].

Kad būtų efektyviai numatomas ir valdomas programinės įrangos kūrimas, reikia sugebėti matuoti programinę įrangą. Tokiu pat būdu, kaip centimetrai ir gramai suteikia mums informacijos apie fizinio, realaus pasaulio objektų dydį ir masę, programinės įrangos matai informuoja mus apie įvairias programinės įrangos charakteristikas [27]. Yra daug įvairių programinės įrangos ir programinės įrangos kūrimo projektų charakteristikų, kurias galime išmatuoti, kaip, pavyzdžiui, programų sistemos dydis, sudėtingumas, patikimumas ir pelningumas[13].

Sistemos pastoviai didėja ir tampa sudėtingesnės. Jas vis sunkiau kurti ir analizuoti. Kodavimo įrankių tobulėjimas leidžia programinės įrangos kūrėjams lengviau pagaminti didesnius programinės įrangos kiekius, kad patenkintų vis didėjančius vartotojų norus. Kadangi sistemos didėja, turi būti naudojamas kažkoks metodas, kuriuo būtų galima nustatyti sistemos dydį [18].

Darbo **tiksla**s – išanalizuoti programinės įrangos kainos skaičiavimo metodus ir sukurti naują modelį kainos skaičiavimui.

Darbo **uždaviniai**:

1. Sukurti modelį programinės įrangos kainos skaičiavimui.
2. Išnagrinėti PĮ kainos skaičiavimo metodų klasifikaciją ir taikymo ypatumus.
3. Išskirti kriterijus, nusakančius PĮ kainos skaičiavimo aspektus atsižvelgiant į sistemos tipą.
4. Atlikti palyginamąją analizę objektų dydžio įvertinimo matų.
5. Sudaryti PĮ kainos skaičiavimo modelį, atsižvelgiant pasirinktą sistemos tipą.
6. Ištirti sudarytą modelį įvertinant žmogiškuosius, techninius ir aplinkos parametrus.

1. PROGRAMINĖS ĮRANGOS PROJEKTAI

Nepaisant sparčios informacinių technologijų plėtros, programinės įrangos kūrimas yra palyginti nauja industrijos šaka, kurios praktikos nėra nusistovėjusios. Tiek kūrėjai, tiek užsakovai ieško sprendimų, kaip optimizuoti programinės įrangos kūrimo procesus, kad minimaliomis lėšomis būtų sukurtas kokybiškas, vartotojo poreikius atitinkantis produktas. [23]

1. 1 PI projektų klasifikacija

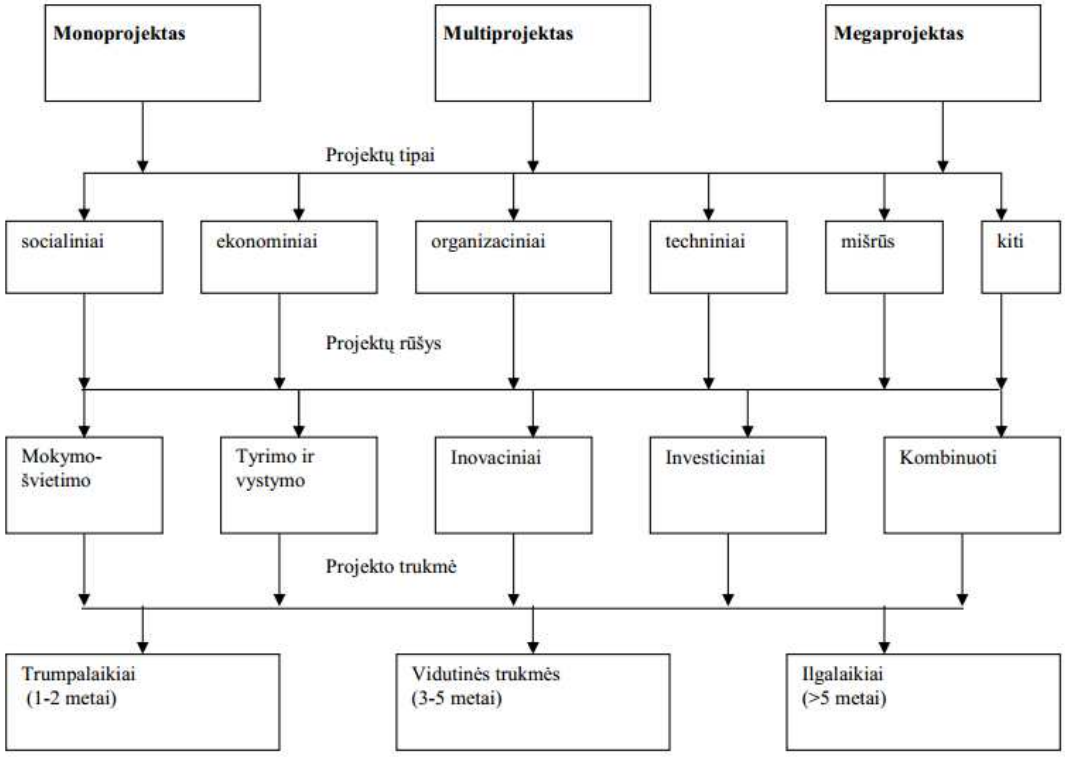
Projektas – kompleksinės, koordinuojamos, vienkartinės pastangos apribotos laiko, biudžeto, išteklių ir kryptingų atlikimo specifikacijų, skirtos patenkinti vartotojo poreikius.

Projektai gali skirtis svarbumu, dydžiu, naujumu, turiniu, trukme, dalyviais, sudėtingumu ir t. t. Pagal tikslą ir metodus projektai klasifikuojami: inžineriniai, produkto (paslaugos) kūrimo, sistemų kūrimo, tyrimų bei organizacinių transformacijų.

Pagal problemų turinį, nustatantį problemos aktualumą ir jų sprendimų naujumą, projektai gali būti tipiniai ir unikalūs. Tipiniai projektai gali būti atkuriami skirtingose situacijose, pakoregavus pagal vietos sąlygas. Unikalūs projektai negali būti kartojami, nes jų situacijos nesikartoja, neįmanoma sukurti kitos panašios projektavimo srities (pvz., unikalaus objekto rekonstrukcijos projektas).

Projektus siūloma skirstyti į klases, tipus, rūšis. Klasė nustatoma pagal projekto sudėtį ir jo veiklos srities struktūrą: monoprojektai (atskiri, nepriklausomi, skirtingo tipo ir dydžio projektai), multiprojektai (projektų kompleksas ar programa, susidedanti iš tarpusavyje susijusių monoprojektų), megaprojektai (tikslinės regionų vystymo programos, apimančios keletą tarpusavyje susijusių projektų, turinčių bendrą tikslą, bendrus išteklius bei jiems įgyvendinti nustatytą laiką, tokios programos gali būti tarptautinės, valstybinės, nacionalinės, regioninės).

Pagal trukmę projektus galima skirstyti į trumpalaikius (iki 2 metų ir trumpesni projektai – gali trukti ir vos keletą mėnesių), vidutinės trukmės ir ilgalaikius. Projekto sudėtingumą lemia jo aprėptis ir valdymo struktūra. Pagal projekto dydį, dalyvių skaičių ir įtaką aplinkai projektai gali būti skirstomi į smulkius, vidutinius, stambius ir labai stambius. Projektai taip pat gali būti tarpvalstybiniai, tarptautiniai, nacionaliniai, regioniniai ar pan (žr. 1 pav.) [2].



1 pav. *Projektų klasifikacija [2]*

Vieni iš sudėtingiausių projektų yra projektai susiję su informacinėmis technologijomis (IT). Daugeliu šių projektų siekiama pagerinti IT infrastruktūrą, palaikančią visą organizaciją.

Šiuolaikinės programinės įrangos negalima vertinti vien tik pagal produkto dydį t.y. kodo eilučių skaičių. Taip neįvertinamas tikrasis projekto sudėtingumas. Todėl yra siūloma IT projektų charakteristikas įvertinti taškais, pagal kuriuos nustatomas projekto sudėtingumas.

Yra pateikiamas metodas, kuriuo galima naudotis klasifikuojant informacinių technologijų projektus kaip: mažas, vidutinis, didelis projektai. Kai IT projektai skirstomi pagal dydį, įmonės gali taikyti atitinkamus projektų valdymo metodus ir procesus, kad atitiktų projekto tikslus. Pirmas žingsnis yra nuspręsti, kuri projektų valdymo praktika turėtų būti taikoma projektui klasifikuojant jį pagal dydį.

Projekto klasifikacija nustatoma remiantis vertinimo balų sistema, kuria atsižvelgiama į įvairias projekto charakteristikas [28].

2 lentelė. Projekto klasifikacijos nustatymo metodas

Charakteristika	0 taškų	1 taškas	2 taškai	Rezultatas
Projekto kaina	Mažiau nei $500,000	$500,000 – $1,000,000	Daugiau nei $1,000,000	...
Projekto komandos dydis	Mažiau nei 5	5-9 žmonės	Daugiau nei 9	...
Dalyvaujantys departamentai	1-2 departamentai	3-4 departamentai	Daugiau nei 4 departamentai	...

Dalyvaujantčios įmonės	Darbo grupė įmonėje	Įmonė mastu	Daugiau nei viena įmonė	...
Veiklos laikas	Mažiau nei 6 mėnesiai	6-12 mėnesių	Daugiau nei metai	...
Poveikis įmonei	Minimalūs pokyčiai arba prideda funkcijų esamoms sistemoms	Vidutiniai pokyčiai arba keičia naudojamą sistemą, bet nekeičia darbo procesų	Reikšmingi darbo metodų pakeitimai įmonės personalui ir/arba paslaugų teikimui įmonės klientams	...
Poveikis įmonės aplinkai	Dažniausiai pasireiškia įmons vidaus operacijos	Netiesioginis poveikis piliečiams ir/arba turi matomumą įstatymams	Tiesioginis poveikis piliečiams ir/arba turi aukštą matomumą įstatymams	...
Technologijos	Standartinės, įrodytos įmonės technologijos	Įrodytos pramonėje ar valstybiniu lygiu, bet nauja įmones ar programos srityse	Kylančios, neįrodytos	...
Buvusi patirtis su rangovu	Gera patirtis praeityje dirabant su rangovu	Kitos įmonės turėjo problemų su šiuo rangovu	Pirmą kartą užsakomos paslaugos su šiuo rangovu	...
Sistemos sudėtingumas	Autonominė sistema	Integracija su kita sistema	Nauja sistema, kuri turi būti integruota su kitomis ir/arba jos yra kritinės sistemos	...
Bendras rezultatas				...

3 lentelė. Projekto klasifikacijos bendro rezultato nustatymas

Klasifikacija	Mažas projektas	Vidutinis projektas	Didelis projektas
Bendras rezultatas	0-2 taškiai	3-10 taškų	Daugiau negu 10

1. 2 PI tipai

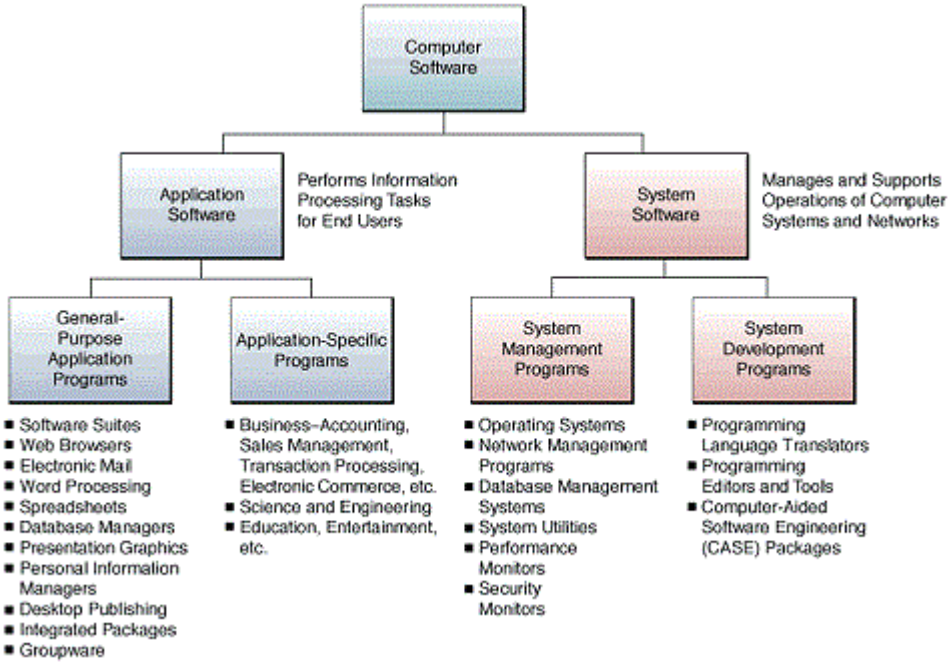
Klasifikuojant programinę įrangą, pirmiausia ją galima suskirstyti į dvi grupes: bendrąją (sisteminę) ir taikomąją. Bendrajai priklauso ta programinė įranga, be kurios kompiuteris apskritai negalėtų funkcionuoti. Tai operacinės sistemos, darbo kompiuterių tinkluose bei kitos programos, kurias turi visi šiuolaikiniai kompiuteriai, nesvarbu, kam jie naudojami. Sisteminė programinė įranga pradeda veikti kompiuterio įjungimo metu ir yra techninės jo įrangos komponentų bei taikomųjų programų darbo koordinatorė. Pagal atliekamas funkcijas ji skirstoma į grupes (žr. 2 pav.):

- valdančioji (operacinės sistemos);
- aptarnaujančioji (utilitos),
- programų kūrimo įrangą, ją sudaro programavimo kalbos ir kalbų procesoriai.

Taikomąją programinę įrangą sudaro programos ir jų sistemos, skirtos konkretiems vartotojo uždaviniams spręsti. Ji apima įmonių programinę įrangą, biuro programas, grafinę programinę įrangą, garso grotuvus. Dauguma taikomųjų programų susijusių su dokumentais.

Šią įrangą kuria ir profesionalūs programuotojai (dažniausiai nemaži jų kolektyvai), ir kvalifikuoti vartotojai. Eiliniam vartotojui svarbu suprasti, kuri taikomosios įrangos programų sistema (dažnai vadinama paketu) tinkamiausia jo uždaviniui atlikti.

Programinės įrangos skirstymas į sisteminę ir taikomąją yra labai sąlygiškas. Pavyzdžiui, darbo kompiuterių tinkluose programas dažnai galima priskirti ir prie vienos, ir prie kitos, ir dažnai labiausiai pavykusios taikomosios programos vėliau tampa sisteminėmis. Nors jau yra sukurta labai daug taikomųjų programų paketų, vis dėlto vartotojų poreikiai auga, ir nuolat tenka kurti naujas programas. Todėl ir derėtų išskirti svarbią programinės įrangos dalį - programų kūrimo įrangą. Ją sudaro programavimo kalbos ir kalbų procesoriai (transliatoriai). Ši dalis galėtų būti vadinama universaliąja, nes naudojama tiek sisteminėms, tiek taikomosioms programoms rašyti ir įdiegti. [31]



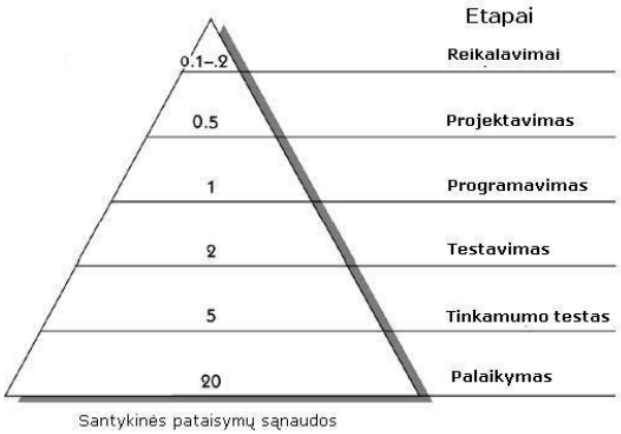
2 pav. *Programinės įrangos tipai [29]*

1. 3 Programinės įrangos kūrimas

1. 3. 1 Reikalavimų analizė

Mūsų laikais informacinės sistemos nebėra pagalbinė verslo priemonė – jos jau seniai tapo neatsiejama daugumos verslo organizacijų integruota dalimi, neatsiejamai dalyvaujančia verslo procesuose. Tačiau IS vystymo tempai, apimtys tik nesustojamai toliau didėja. Ypač daug dėmesio skiriama Internetui orientuotų sistemų kūrimui bei įvairių informacinių sistemų integracijai. Augant kuriamos programinės įrangos apimtims ir sudėtingumui, projektavimas tampa vis svarbesne IS inžinerijos dalimi. IS kūrimo ir vystymo projektams tampant sudėtingesniems ir jų apimtims augant, nemažai projektų užbaigiami su viršytomis laiko ar biudžeto sąnaudomis arba nebaigiami išvis.

Todėl IS projektavime, reikalavimų valdymas yra kritinis procesas, galintis užtikrinti sėkmingą viso IS projekto vykdymą ir užbaigimą. Netinkamas reikalavimų valdymas yra labiausiai pasitaikančios IS kūrimo klaidos, kurios kainuoja brangiausiai. Pateiktame paveikslėlyje (žr. 3 pav.) matome, kad klaidų pataisymas reikalavimų apibrėžimo stadijoje užima 5-10 kartų mažiau laiko negu jas taisant programavimo metu ir 100-200 kartų mažiau laiko, negu palaikant jau įdiegtą sistemą [22]:



3 pav. *Pataisymų kaštai IS kūrimo stadijose [22]*

Programinės įrangos kūrimas prasideda nuo reikalavimų analizės ir aprašymo. Daugiausia problemų, kuriant programinę įrangą, kyla būtent iš prastai dokumentuotų, nepilnai aprašytų, skirtingai suprantamų bei projekto eigoje besikeičiančių reikalavimų. Labai svarbu suprasti, kad programinės įrangos projektų reikalavimai turi būti nuosekliai analizuojami keliuose lygiuose – nuo vizijos ir verslo tikslų iki galutinio vartotojo scenarijų bei realizacijos funkcijų aprašymo (žr. 4 pav.) [22].

4 pav. *Cockburn reikalavimų laivo modelis [22]*

Apie reikalavimų analizės kokybę dažnai galima spręsti iš vėliau parengtų vartotojo instrukcijų. Jeigu aprašinėjami vien vartotojo sąsajos langai ir mygtukai, tačiau neaišku, kaip atlikti veiksmų seką, leidžiančią pasiekti vartotojui reikalingą rezultatą, tai yra indikatorius, kad reikalavimų analizė buvo atlikta tiktai realizacijos lygmenyje, neatsižvelgiant į vartojimo scenarijus.

Reikalavimų kokybė tiesiogiai priklauso nuo užsakovų ir galutinių vartotojų įtraukimo į projektą. Siekiant kuo labiau įtraukti vartotoją į projektą ir išgauti iš jo tikslius reikalavimus, naudojami įvairūs metodai – nuo "brainstorming" sesijų ir klausymų iki interaktyvių sistemos prototipų kūrimo. Net ir kruopščiai surinkti reikalavimai gali keistis projekto eigoje dėl įvairių priežasčių, todėl užsakomuosiuose projektuose labai svarbu įvesti formalią reikalavimų pakeitimų procedūrą. Chaotiškai keičiant reikalavimus, išauga projekto apimtis, nesilaikoma tvarkaraščių, krenta produkto kokybė [23].

Reikalavimų pasikeitimai yra potencialus projekto problemų šaltinis. Netinkamai juos valdant, projektas gali užsitęsti, kainuoti daugiau nei buvo numatyta ar iš viso nepavykti. Todėl yra būtina kiek galima anksčiau ir tiksliau juos apibrėžti.

Viena svarbiausių ir dažniausiai pasitaikančių problemų, trukdanti tinkamai apibrėžti reikalavimus yra užsakovo neapsisprendimas. Kartais gali net susidaryti įspūdis, kad klientas nežino ko nori – tai nutinka dėl tokių priežasčių:

- Užsakovas neturi patirties IT srityje ir tiesiog nežino, kokius sprendimus ji gali pateikti.
- Užsakovams sunku tiksliai įsivaizduoti naujos sistemos galutinį variantą, jei jiems tenka pereiti nuo kažkokios kitos egzistuojančios sistemos.
- Dažniausiai tik pabandę galutinį produktą ar bent jo prototipą, užsakovai gali tiksliai pasakyti, ar šis jiems tinka.

Net iš pirmo žvilgsnio nežymūs pakeitimai reikalauja papildomo laiko ar išlaidų. Kuo vėliau projekte šie pakeitimai yra inicijuojami, tuo daugiau jie kainuoja. Svarbu yra tiksliai numatyti, kokią įtaką projektui turės reikalavimų pasikeitimas – ir suderinti tai su komanda bei užsakovu. Taip pat svarbu yra atskirti reikalavimų pasikeitimą nuo reikalavimų pridėjimo – tai padaryti padės tik tiksliai parengta sistemos reikalavimų specifikacija [24].

1. 3. 2 Projektavimas

Kaip ir pastatus prieš statybas ar automobilius prieš pradedant jų gamybą, taip ir programinės įrangos sistemas reikia suprojektuoti. Tik turint pakankamai detalų projektą galima pereiti į jų gamybos – programavimo – etapą. Programinės įrangos projektavimas galimas daugybe pjūvių. Šiuo metu populiariausias yra 4+1 pjūvių architektūros modelis, išskiriantis konceptualų, realizacijos, procesų ir išdėstymo pjūvius bei juos apjungiantį funkcionalumo (panaudojimo atvejų) pjūvį (žr. 5 pav.). [23]

5 pav. *4+1 architektūrinių pjūvių modelis [23]*

Kuriais pjūviais analizuoti ir projektuoti labai priklauso nuo projekto specifikos. Nors daugiausiai dėmesio įprasta skirti realizacijos pjūviui, kuris aprašo programinio kodo struktūrą ir veikimą, tačiau to neužtenka.

Kuriant paskirstytas sistemas, projektuojamas išdėstymo pjūvis, nurodantis, kaip programinės įrangos komponentai bus paskirstyti techninėje įrangoje ir kokiais protokolais bendraus tarpusavyje. Procesų pjūvis svarbus realaus laiko ir paskirstytų procesų sistemose, kur reikia užtikrinti greičio, našumo, pralaidumo charakteristikas. Visi pjūviai projektuojami, naudojant UML modeliavimo kalbą, kuri šiuo metu yra visuotinai pripažinta kaip programinės įrangos sistemų analizės ir projektavimo kalba.

Programinės įrangos projektai daugiau ar mažiau keičiasi pradėjus programuoti, todėl reikia tinkamai pasirinkti, kuriuos projektavimo artefaktus atnaujinti ir išlaikyti projekto eigoje, o kuriuos panaudoti pradinei analizei ir komunikacijai, o vėliau tiesiog išmesti. Per daug detalaus techninio projekto palaikymas gali tapti nereikalinga našta ir įvesti sinchronizacijos chaosą. Detalų projektavimą tikslingiau atlikti programavimo metu, o naudojant atstatomosios inžinerijos įrankius, galima kai kurias modelio diagramas nubraižyti automatizuotai pagal programinį kodą. Dažniausiai modeliuojami, dokumentuojami ir palaikomi šie programinės įrangos sistemų projektavimo aspektai:

- Duomenų struktūros;
- Programinių modulių išskaidymas ir integracija;
- Vartotojo sąsajos elementai ir navigacijos schemos. [23]

1. 3. 3 Programavimas

Programavimas yra kertinė programinės įrangos kūrimo veikla, kurios metu sukuriamos veikiančios programinės įrangos versijos. Profesionaliam programavimui reikalingas ne tik geras

naudojamų programavimo kalbų ir technologijų išmanymas, bet ir analitinis mąstymas, komandinio darbo įgūdžiai, bendro stiliaus susitarimai ir kūrimo įrankių rinkinys.

Labai svarbu vykdyti programinio kodo peržiūras – jos leidžia pastebėti ir ištaisyti daugumą kodo defektų, pagerinti programinio kodo struktūrą, skaitomumą, pakelti sistemos veikimo greitį. [23]

1. 3. 4 Testavimas

Testavimas apima tiek galutinio programinės įrangos produkto, tiek tarpinių artefaktų tikrinimą, siekiant įvertinti, ar pasiekti rezultatai tenkina lūkesčius. Nemažai programinę įrangą kuriančių kompanijų atlieka tiktai galutinio produkto testavimą, prieš perduodant jį vartotojams. Kartais testavimas netgi perkeliamas ant užsakovo pečių – nustatomas programinės įrangos perėmimo periodas, kurio metu vartotojai turi ją ištestuoti ir pranešti apie defektus.

Svarbu, kad programinę įrangą ar kitus artefaktus (reikalavimų specifikacijas, programinį kodą ir kt.) tikrintų ne jų autorius, o kitas žmogus. Lietuvių patarlė byloja, kad “kito akyje ir krislą pastebėsi, o savoje ir rąsto nematai”. Jeigu programinės įrangos modulį testuoja tas pats programuotojas, daug defektų lieka nepastebėti.

Visų pastebėtų defektų iškart ištaisyti neįmanoma, todėl juos visus reikia užregistruoti. Defektai grupuojami pagal prioritetą, kuris nurodo, kaip dažnai defektas pasitaiko vartotojui, ir žalingumą, kuris nusako defekto poveikį sistemai. Defekto aprašymas turi būti aiškus, kad defektą būtų galima pakartoti ir ištaisyti. Neužtenka tik aprašyti ir pataisyti defektus – reikalingas nuoseklus defektų valdymas. Kiekvienas defektas turi būti išsprendžiamas, tačiau tai nebūtinai turi būti ištaisymas – kai kurie defektai atidedami, kiti yra nepataisomi, dar kiti atmetami, jeigu jie užregistruojami dėl klaidingo testuojančio žmogaus supratimo apie sistemą. Ištaisius defektą, reikia patikrinti, ar jis buvo tinkamai ištaisytas ir ar nesąlygojo naujų defektų atsiradimo. Todėl efektyviam testavimui užtikrinti būtina naudoti defektų registravimo ir valdymo sistemą bei nustatyti defektų užregistravimo, taisymo, patikrinimo ir užbaigimo procedūras.

Prieš perduodant programinės įrangos produktą vartotojams, reikia užtikrinti, kad pasiektas tinkamas produkto kokybės lygis, pavyzdžiui, turi būti ištaisyti visi aukščiausio prioriteto ir didžiausio žalingumo defektai ir savaitę laiko naujų tokių defektų neberandama [23].

1. 4 PĮ projekto dalyviai

Projekto žmogiškųjų resursų valdymas apima projekto komandos organizavimo ir valdymo procesus. Projekto komanda sudaryta iš asmenų, kuriems priskirtos rolės ir atsakomybės projekte. Komandos nariai turėtų būti įtraukiami į projekto planavimą ir sprendimų priėmimą.

Projekto žmogiškųjų resursų valdymo procesai:

Žmogiškųjų resursų planavimas (angl. Human Resource Planning): identifikavimas ir dokumentavimas projekto rolių, atsakomybių ir atsiskaitomybių bei kadrų komplektavimo valdymo plano sudarymas.

Projekto komandos surinkimas (angl. Acquire Project Team): gavimas žmogiškųjų resursų, reikalingų projektui vykdyti.

Projekto komandos suformavimas (angl. Develop Project Team): pagerinimas komandos narių kompetencijos ir tarpusavio bendravimo, kad padidinti darbo efektyvumą.

Projekto komandos valdymas (angl. Manage Project Team): komandos narių darbo efektyvumo stebėjimas, grįžtamojo ryšio užtikrinimas, problemų sprendimas ir pakeitimų koordinavimas, siekiant padidinti darbo efektyvumą.

Pagrindinis IT projekto vadovo tikslas yra užtikrinti suformuotos projekto komandos darbą ir įtraukti į projekto vykdymą užsakovą, vartotojus ir kitus susijusius asmenis. Šio tikslo yra siekiama visose pagrindinėse projekto valdymo veiklose.

Pradiniame projekto apibrėžimo etape projekto vadovas sukuria projekto vykdymo aplinką, palankią visiems projekto dalyviams dirbti vienoje komandoje:

- sudaromas projekto dalyvių sąrašas ir įvertinamos jų rolės projekte, apibrėžiami sutartiniai įsipareigojimai,
- išsiaiškinami projekto dalyvių lūkesčiai, turintys įtakos formuluojant projekto tikslus,
- įvertinama projekto apimtis: apibrėžiamas galutinis laukiamas rezultatas bei tarpiniai darbo produktai,
- pateikiami pradiniai biudžeto ir tvarkaraščio pasiūlymai,
- pasirašoma sutartis [14].

Projekte dalyvaujantys žmonės yra pats svarbiausias faktorius. Nuo jų profesionalumo, patirties ir motyvavimo priklauso projekto sėkmė. Projektų vadovai teigia, kad be darnaus ir profesionalaus kolektyvo joks projektas nebus sėkmingas.

Komandinis darbas grindžiamas bendradarbiavimu ir tarpusavio pagalba, bet jei komandoje dirba labai panašūs žmonės, jų darbo efektyvumas nebūna aukštas. Tik iš panašių siekių, bet kartu skirtingų ir vienas kitą papildančių žmonių grupės gali susiformuoti efektyvi komanda.

Projekto įgyvendinime dalyvauja kelios suinteresuotų žmonių grupės, kurias galima dalinti į penkias grupes:

- Aukščiausias pareigas užimantys vadovai – nusako verslo tikslus ir daro didelę įtaką projektui.
- Projektų (techniniai) vadovai – sudaro projekto planus, organizuoja, motyvuoja ir valdo komandą. Jis atsako už priimamus projektinius sprendimus, atliekamų darbų kokybę ir savalaikiškumą, informacijos apdorojimo technologijas, projektavimo darbų valdymą ir detalų planavimą.
- Specialistai (programuotojai, analitikai, testuotojai ir t.t.) – žmonės, kurie kuria produktą.
- Užsakovas – nustato kuriamos programinės įrangos reikalavimus. Jis atsako už darbų finansavimą, techninių bei sisteminių programinių priemonių įsigijimą, darbuotojų parengimą,organizacinius pertvarkymus, reikiamos informacijos pateikimą.

- Vartotojas – programinės įrangos naudotojas.

Kiekvienas projektas susideda iš šių žmonių grupių. Projektų vadovas turi taip organizuoti komandą, kad maksimaliai išnaudoti kiekvieno žmogaus sugebėjimus ir įgūdžius [1].

1. 5 Pį kainos skaičiavimas

Tradicinis programinės įrangos dydžio matas yra kodo eilučių skaičius. Kodo eilučių skaičius matuoja fizinį programinės įrangos dydį. Matuojant produktyvumą, kodo eilučių skaičiaus ar kokio kito fizinio dydžio matas nėra tinkamas. Fizinio dydžio matai turi mažai bendro su programinės įrangos teikiama nauda, o daugiau su pačia jos kūrimo technologija. Klientams rūpi funkcionalumas, o ne kiek kodo eilučių reikėjo jam sukoduoti ir derinti. Todėl papildomai reikia kažkokių kitų programinės įrangos dydžio skaičiavimo matų.

Projektai yra vertinami atliekant pirminį spėjimą, kuris naudojamas kaip priemonė pasiūlymų atrankai, o preliminarus vertinimas ir tikslus įvertinimas yra naudojami apimties įvertinimui projekto eigoje. Preliminarus įvertinimas pagrįstas vartotojo poreikių analize ir statistine informacija apie panašių projektų apimtį. Lietuvoje nėra išsamios IT projektų statistikos, todėl IT kompanijos priverstos vadovautis tik savo patirtimi.

Praktikoje dažniausiai taikomi du pagrindiniai netikslaus apimties įvertinimo rizikos valdymo būdai: valandinio įkainio kontraktai ir fazinis apimties įvertinimas fiksuotos kainos kontraktuose. Valandiniai įkainio kontraktai pasižymi tuo, kad esant neaiškiems reikalavimams, projekto biudžetas ir trukmė apibrėžiami labai abstrakčiai ir užsakovas moka vykdytojui už darbo laiką pagal sutartą valandinį įkainį. Faziniam apimties įvertinimui fiksuotos kainos kontraktuose yra būdingas preliminarus apimties įvertinimas visam projektui ir tikslų įvertinimą artimiausiai fazei, po kiekvienos fazės patikslinamas preliminarus įvertinimas likusiai projekto daliai ir pateikiamas tikslus būsimos fazės įvertinimas [14].

Viena iš opiausių IT projektų problemų yra neplanuotai didėjantys projektų biudžetai ir vėlavimas. Užsakovai bando apsisaugoti fiksuotos kainos sutartimis bei baudomis už vėlavimą. Vykdytojai bando išvengti apimties didėjimo tobulindami reikalavimų pasikeitimų valdymą, reikalaudami papildomo laiko ir apmokėjimo pakeitimams atlikti. Deja, šios priemonės dažniausiai problemos neišsprendžia, bet padidina užsakovo ir vykdytojo konfrontaciją. Norint kovoti su projektų trukmės ir biudžeto didėjimo problema reikalinga suprasti ir pašalinti jos priežastis.

Kadangi projekto inicijavimo fazėje visos projekto apimties detalės nėra žinomos, esant tokioms sąlygoms atsiranda tikimybė per klaidą ką nors užmiršti. Nepaisant šio bendro projekto apimties nustatymo, egzistuoja kiti pavojai, tokie kaip pradžioje apskaičiuota projekto įgyvendinimo data paprastai yra pernelyg optimistinė, projekto pabaigos data gali būti apskaičiuota neįvertinus projekto rizikos, taip pat egzistuoja užsakovų spaudimas.

Pradinis projekto biudžetas gali neleisti atsitiktinių nukrypimų nuo apimties, suklystų elementų pakartotinio atlikimo ar projekto vėlavimo. Nustatyta projekto apimtis dėl techninių problemų, tvarkaraščio terminų ar biudžeto ribojimo gali būti keičiama, tačiau tai remiasi į kainą. Projektų vadovų darbo pobūdžio vienas iš aspektų yra projekto darbų apimties nustatymas, kuris

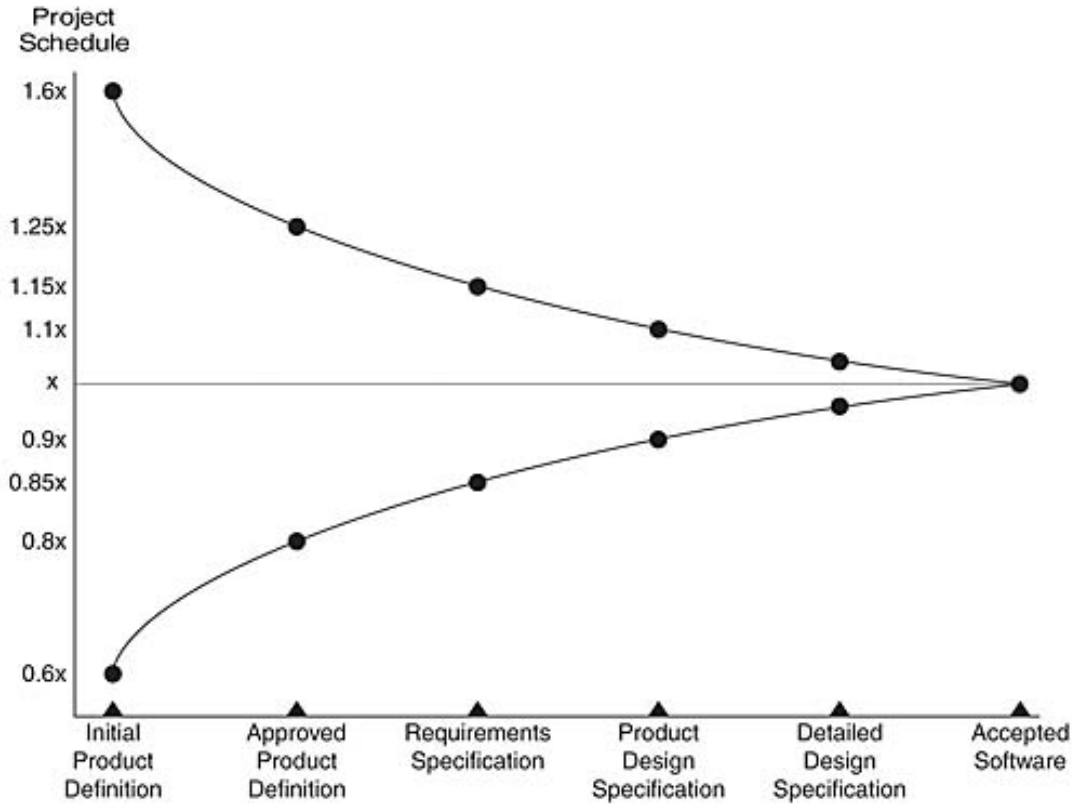
padeda planuojant projektą, rengiant biudžetą, organizuojant darbus, vėliau ruošiant specifikacijas, instrukcijas ir susijusius dokumentus.

Apimtis – svarbiausia projektų vadybos grandis. Apimtis yra suprantama kaip kas turės būti padaryta. Į apimtį (plačiąja prasme) įeina ir kokybė. Projektų vadovas, IBM Lietuva (Kęstutis Lašinskas) savo pristatyme „Geros ir blogos projektų valdymo praktikos“ išskiria vieną iš svarbiausių aspektų dirbant projektuose - darbo apimtį. Iš to galima teigti, kad laiku ir tinkamai įvertinta darbo apimtis gali padėti pasiekti pageidaujamą kokybę [15].

PĮ kaštų įvertinimas yra nuolatinis, nenutrūkstamas procesas. Nagrinėjant tam tikros PĮ idėją, galima susidaryti bendrą vaizdą apie jos apimtis ir reikalingas pastangas. Nėra paprasto būdo tiksliai įvertinti programų sistemos reikalingas kūrimo pastangas, nes:

- pradiniai vertinimai remiasi neadekvačia reikalavimų apibrėžimo informacija;
- programos gali naudoti naujas technologijas;
- projekte dirbantys žmonės nepakankamai pažįstami.
- projekto kainos vertinimas gali būti pats apsprendžiantis, t.y. vertinimas apsprendžia biudžetą ir produktas yra priderinamas prie biudžeto.

Kaštų įvertinimas yra nuolat tikslinamas, atliekant reikalavimų specifikavimą, projektuojant, koduojant ar testuojant. Kuo daugiau žinoma apie kuriamą sistemą ir kuo vėlesnis sistemos kūrimo etapas – tuo tikslesnis yra įvertinimas. Pvaeiksle (žr. 6 pav.) pateiktas klasikinis PĮ kainos prognozės kitimas pagal Barry W. Boehm priklausomai nuo PĮ kūrimo etapo [25].



6 pav. *Projekto vertinimo proceso modelis [30]*

1. 5. 1 Funkcinių taškų analizės metodas

Duomenų funkcijų ir jų sudėtingumų nustatymas:

Yra dviejų tipų duomenų funkcijos:

- **vidinis loginis failas** (VLF). Tai programos ribose palaikomų logiškai susijusių duomenų ar valdymo informacijos grupė. Pagrindinis VLF tikslas yra saugoti matuojamos programos duomenis, naudojamus vieno ar kelių elementarių procesų metu. Tai savotiškas esybių identifikavimas (nebūtinai duomenų bazės lentelės. Tai gali būti failų tipai, kiti operacinės sistemos objektai);
- **išorinis interfeiso failas** (IIF). Tai logiškai susijusių duomenų ar valdymo informacijos grupė, į kurią kreipiasi programa, bet yra palaikoma kitos programos ribose. Pagrindinis IIF tikslas yra saugoti duomenis, į kuriuos kreipiamasi vieno ar kelių matuojamos programos ribose vykdomų elementarių procesų metu. VLF apskaičiuotas vienoje programoje yra IIF kitos programos, naudojančios tą VLF, ribose. Tai informacija arba failai, kuriais vyksta mainai su kitomis sistemomis. Pvz. duomenys apie mokesčių mokėtojo darytus pavedimus eksportuojami į Excel programą.

Nustačius matuojamos programinės įrangos vidinių loginių failų ir išorinių interfeiso failų kiekį, reikia nustatyti kiekvieno jų sudėtingumą. VLF ir IIF sudėtingumas priklauso nuo dviejų faktorių:

- **duomenų elementų tipai** (DET). Tai unikalūs naudotojam suprantami, nesikartojantys laukai ar atributai;
- **įrašų elementų tipai** (ĮET). Tai naudotojam suprantami duomenų elementų, esančių VLF ir IIF, pogrupiai.

Kiekvienam nustatytam VLF ir IIF turi būti priskirtas funkcinis sudėtingumas, priklausantis nuo duomenų elementų tipų ir įrašų elementų tipų, susijusių su VLF ar IIF, skaičiaus. Galimos sudėtingumo reikšmės: žemas, vidutinis arba aukštas. Jos priskiriamos pagal DET ir ĮET kiekį naudojant VLF ir IIF sudėtingumo matricos 1 lentelę.

4 lentelė. VLF ir IIF sudėtingumo matrica

ĮET	DET		
	1-19	20-50	>50
1	Žemas	Žemas	Vidutinis
2-5	Žemas	Vidutinis	Aukštas
>5	Vidutinis	Aukštas	Aukštas

Transakcinių funkcijų ir jų sudėtingumų nustatymas:

Transakcinės funkcijos yra skirstomos į tris grupes:

- **išorinis įvedimas** (IĮ). Tai elementarus procesas, kuris apdoroja duomenis ar valdymo informaciją, kuri atvyksta iš programos išorės. Pagrindinis IĮ tikslas yra palaikyti vieną ar daugiau VLF arba keisti sistemos elgseną. Tai rašymo, informacijos keitimo transakcijos. Pvz. tam tikro duomenų rinkinio registravimas;

- **išorinė užklausa** (IU). Tai elementarus procesas, kuris siunčia duomenis ar valdymo informaciją į programos išorę. Pagrindinis IU tikslas yra pateikti informaciją naudotojui per duomenų išrinkimą ar valdymo informaciją iš VLF ar IIF. Vykdymo logika neturi jokių matematinių formulių ar skaičiavimų, nesukuria jokių išvedamų duomenų. Joks VLF nėra palaikomas bei nekeičiama sistemos elgsena. tai vartotojo transakcijos, kurios teikia informaciją, bet nekeičia jos. Tačiau informacija neišeina už sistemos ribų. Pvz. peržiūros formos;
- **išorinis išvedimas** (II). Tai elementarus procesas, kuris siunčia duomenis ar valdymo informaciją į programos išorę. Pagrindinis II tikslas yra pateikti informaciją naudotojui per vykdymo logiką. Vykdymo logika turi turėti bent vieną matematinę formulę ar skaičiavimą, sukurti išvedamus duomenis, palaikyti vieną ar daugiau VLF arba keisti sistemos elgseną. Tai informacijos išvedimo, ataskaitų ar dokumentų formavimo operacijos. Pvz. ataskaitos apie mokesčių mokėtoją formavimas ir spausdinimas.

Nustačius matuojamos programinės įrangos išorinių įvedimų, išorinių užklausų ir išorinių išvedimų kiekį, reikia nustatyti kiekvieno jų sudėtingumą. IĮ, IU ir II sudėtingumas priklauso nuo programos ribas kertančių DET ir:

- **kreipimosi failų tipai** (KFT). Tai bendras skaitomų ar palaikomų VLF ir skaitomų IIF, vykdant IĮ ir II transakcijas, skaičius. Vykdant IU transakcijas skaičiuojamas tik skaitomų VLF ir IIF skaičius.

Kiekvienam nustatytam IĮ, IU ir II turi būti priskirtas funkcinis sudėtingumas, priklausantis nuo duomenų elementų tipų ir kreipimosi failų tipų, susijusių su IĮ, IU ir II, skaičiaus. Galimos sudėtingumo reikšmės: žemas, vidutinis arba aukštas. Jos priskiriamos pagal DET ir KFT kiekį naudojant IĮ sudėtingumo matricos 2 lentelę, bei IU ir II sudėtingumo matricos 3 lentelę.

5 lentelė. IĮ sudėtingumo matrica

KFT	DET		
	1-4	5-15	>15
0-1	Žemas	Žemas	Vidutinis
2	Žemas	Vidutinis	Aukštas
>2	Vidutinis	Aukštas	Aukštas

6 lentelė. IU ir II sudėtingumo matrica

KFT	DET		
	1-5	6-19	>19
0-1	Žemas	Žemas	Vidutinis
2-3	Žemas	Vidutinis	Aukštas
>3	Vidutinis	Aukštas	Aukštas

Nekoreguotų funkcinių taškų skaičiavimas:

Nustačius duomenų ir transakcinių funkcijų kiekį ir jų sudėtingumus, skaičiuojami nekoreguoti funkciniai taškai (NFT). NFT skaičiuojami naudojant IFPUG nekoreguotų funkcinių taškų 4 lentelės koeficientais, susumuojant visų funkcijų sudėtingumų lygių įverčius:

$$NFT = \sum (VLF \times S) + \sum (IIF \times S) + \sum (IĮ \times S) + \sum (IU \times S) + \sum (II \times S) \quad (1)$$

NFT – nekoreguoti funkciniai taškai;

VLF – vidinis loginis failas;

IIF – išorinis interfeiso failas;

IĮ – išorinis įvedimas;

IU – išorinė užklausa;

II – išorinis išvedimas;

S – duomenų arba transakcinės funkcijos sudėtingumas iš atitinkamos sudėtingumo matricos;

7 lentelė. IFPUG nekoreguotų funkcinių taškų lentelė

Funkcijos	Funkcijų sudėtingumo lygiai		
	Žemas	Vidutinis	Aukštas
Vidinis loginis failas	×7	×10	×15
Išorinis interfeiso failas	×5	×7	×10
Išorinis įvedimas	×3	×4	×6
Išorinė užklausa	×3	×4	×6
Išorinis išvedimas	×4	×5	×7

Reikšmių koregavimo faktoriaus skaičiavimas:

Kad galėtume apskaičiuoti koreguotus funkcinius taškus, pirma reikia nustatyti reikšmių koregavimo faktorių (RKF). Šiuo žingsniu į programos vertinimą įtraukiami ir nefunkciniai (kokybiniai) reikalavimai. Skaičiuojant RKF yra atsižvelgiama į 14 bendrųjų sistemos charakteristikų (BSC).

8 lentelė. Bendrosios sistemos charakteristikos

Bendrosios sistemos charakteristikos	Trumpas aprašas
1. Duomenų perdavimas	Programos tiesioginio bendravimo su procesoriumi laipsnis
2. Paskirstytų duomenų apdorojimas	Programos duomenų persiuntimas tarp savo komponentų laipsnis
3. Našumas	Programos atsako laiko ir našumo laipsnis
4. Dažnai naudojama konfigūracija	Sistemos naudojamų kompiuterinių resursų laipsnis
5. Tranzakcijų greitis	Verslo transakcijų greičio laipsnis
6. Tiesioginis duomenų įvedimas	Duomenų įvedimo per interaktyvias transakcijas laipsnis
7. Galutinio naudotojo efektyvumas	Programos naudojimo paprastumo ir suprantamumo naudotojui laipsnis
8. Tiesioginiai pakeitimai	Programos vidinių loginių failų tiesioginio atnaujinimo laipsnis
9. Sudėtingas apdorojimas	Programos vykdymo logikos sudėtingumo laipsnis
10. Pakartotinis panaudojamumas	Programos ir jos kodo pakartotinio panaudojimo kitose sistemose laipsnis
11. Diegimo paprastumas	Perėjimo iš ankstesnių aplinkų ir diegimo paprastumo laipsnis
12. Naudojimo paprastumas	Programos rūpinimosi paleidimo, dubliavimo ir atstatymo procesais laipsnis
13. Daugkartinis panaudojimas	programos tinkamumas skirtingoms techninėms ir programinėms aplinkoms laipsnis
14. Pakeitimų lengvumas	Programos vykdymo logikos ar duomenų struktūrų lengvo pakeitimo laipsnis

Kiekvienos BSC įtaka kuriamai sistemai yra įvertinama nuo 0 (jokios įtakos) iki 5 (didelė įtaka) balų.

9 lentelė. Sudėtingumo laipsniai ir jų reikšmės

Reikšmė	Įtakos laipsnis
0	Nėra/neturi
1	Netikėta
2	Nedidelė
3	Vidutinė
4	Reikšminė
5	Didelė

Bendras įtakos laipsnis (BĮL) apskaičiuojamas susumuojant visų 14 BSC įtakų vertes. Reikšmių koregavimo faktorius tada apskaičiuojamas taip:

$$RKF = (BĮL \times 0{,}01) + 0{,}65 \quad (2)$$

RKF – reikšmių koregavimo faktorius;

BĮL – bendras įtakos laipsnis.

RKF maksimali reikšmė gali pasiekti 1,35, o minimali 0,65, todėl ji gali koreguoti nekoreguotus funkcinius taškus +/- 35% .

Sudaroma skaičiavimo lentelė, kurioje registruojami visi nepritaikyti funkciniai taškai, dauginami iš jų sudėtingumo koeficientų, sumuojami ir dauginami iš pritaikymo koeficiento.

10 lentelė. Skaičiavimo šablonas

Tipas	Sudėtingumas									
	Žemas			Vidutinis			Aukštas			Viso
	Kiek	Koef.	Viso	Kiek	Koef.	Viso	Kiek	Koef.	Viso	
IĮ	X	3	X×3	X	4	X×4	X	6	X×6	...
II	X	4	X×4	X	5	X×5	X	7	X×7	...
IU	X	3	X×3	X	4	X×4	X	6	X×6	...
VLF	X	7	X×7	X	10	X×10	X	15	X×15	...
IIF	X	5	X×5	X	7	X×7	X	10	X×10	...
Nepritaikytų taškų reikšmių suma										...
Pritaikymo daugiklis										...
Pritaikytų taškų reikšmių suma										...

Koreguotų funkcinių taškų skaičiavimas:

Šiame etape skaičiuojami koreguoti funkciniai taškai (KFTa). Kaip jau minėta, yra trys funkcinių taškų skaičiavimo tipai ir kiekvienam jų koreguoti funkciniai taškai apskaičiuojami naudojant skirtingas formules. [8]

Programinės įrangos funkcinių taškų kiekis apskaičiuojamas pagal tokią formulę:

$$KFTa = NFT \times RKF \quad (3)$$

KFTa - programos koreguoti funkciniai taškai;

NFT - programos nekoreguoti funkciniai taškai;

RKF - reikšmių koregavimo faktorius.

Tam tikri laikinių įverčių skaičiavimo metodai, skaičiuodami sistemos apimtį pagal funkcinių taškų metodą, skaičiuoja nepritaikytus funkcinius taškus nenaudojant pritaikymo koeficiento.

Galimas ryšys tarp funkcinių taškų ir fizinių kodo eilučių. Žemiau pateikta funkcinių taškų konvertavimo lentelė pagal programavimo kalbą:

11 lentelė. Funkcinių taškų konvertavimas

Programavimo kalba	Kodo eilučių skaičius per funkcinį tašką	Programavimo kalba	Kodo eilučių skaičius per funkcinį tašką
Assembler	320	Smalltalk	21
Assembler (Macro)	213	Query Languages	16
Algol	106	Spreadsheet Languages	6
Cobol	106	C	150
Fortran	106	C++ default	53
Jovial	106	Delphi	18
Pascal	91	Html4	14
RPG	80	Java 2 default	46
PL/I	80	Visual basic 6	24
Ada	71	ANSI/Quick/Turbo Basic	64
Lisp	64	SQL default	13
Basic	64	Report Generator	80
4th Generation Database	40	PL/Sql	45
APL	32		

Funkcinius taškus konvertuojame į kodo eilučių skaičių, apsiskaičiavę pastangos koregavimo faktorių apskaičiuojame pastangas.

$$SLOC = 16 \times SLOC/KFTa \quad (4)$$

$$EFFORT = EAF \times A \times (SLOC)^{EX} \quad (5)$$

$$EAF = CPLX \times TOOL \quad (6)$$

A = 3,2

EX = 0,38

CPLX = 1,3

TOOL = 1,1

Viską apskaičiavę apskaičiuojame kūrimo laiką mėnesiais.

$$TDEV = 2,5 \times (EFFORT)^{EX} \quad (7)$$

SLOC – kodo eilučių skaičius

EAF – pastangos koregavimo koeficientas

A – konstanta

EX – konstanta

CPLX – techninio sudėtingumo faktoriaus konstanta

TDEV – kūrimo laikas mėnesiais

EFFORT – pastangos

1. 5. 2 Panaudojimo atvejų taškų metodas

Panaudojimo atvejų taškų analizė (angl. *Use Case Point Analysis*, UCP) ‑ šį metodą sukūrė Gustavas Karneris 1993 m. Švedijos įmonėje Objective Systems AB (Karner, 1993), kuri nuo 2003 m. kartu su metodu priklauso įmonę įsigijusiai kompanijai IBM.

Pagal panaudojimo atvejų taškų (UCP) metodiką vietoj funkcinio dydžio PĮ sistemos dydis matuojamas pagal funkcinius PĮ reikalavimus, išreikštus PĮ sistemos aktorių ir panaudojimo atvejų skaičiumi ir jiems suteiktomis svorinėmis vertėmis. UCP modelyje PĮ funkcinis dydis koreguojamas papildomais veiksniais (daugikliais) –techninių veiksnių ir aplinkos parametrais, kurie nėra įtraukti į FPA modelį. Nors pradiniame FPA modelyje dalis techninių veiksnių, kuriuos apima UCP vertinimas, vertinimas buvo įtrauktas, tačiau jis nebuvo standartizuotas. Todėl tolesnis šių veiksnių tyrimas vyksta naudojant UCP metodiką, kuri taikoma ir papildant FPA metodikas.

Panaudojimo atvejų taškų skaičiavimas:

UCP (Use Case Points) skaičiuojamas sudauginant nepakoreguotus panaudojimo atvejų taškus (UUCP), techninio sudėtingumo faktorių (TCF) ir aplinkos sudėtingumo faktorių (ECF).

$$UCP = UUCP \cdot TCF \cdot ECF \tag{8}$$

UUCP susideda iš panaudojimo atvejų svorio (UUCW) ir aktorių svorio (UAW).

12 lentelė. Panaudojimo atvejų svoris

Tipas	Aprašymas	Svoris	Panaudos atvejų skaičius	Rezultatas
Paprastas	Paprasta vartotojo sąsaja, naudojamas tik vienas duomenų bazės subjektas, jo scenarijus turi =< 3 žingsnių	5	...	...
Vidutinis	Didesnis sąsajos projektavimas, naudojami 2 ar daugiau duomnų bazės subjektai, jo scenarijus turi 4-7 žingsnius	10	...	...
Sudėtingas	Apima sudėtingą vartootjo sąsają, naudojami 3 ar daugiau duomenų bazės subjektų, jo scenarijus apima >7 žingsnių	15	...	...
			UUCW = svoris * skaičius:	...

13 lentelė. Aktorių svoris

Tipas	Aprašymas	Svoris	Aktorių skaičius	Rezultatas
Paprastas	Sisteminės sąsajos: ktorius atstovauja kitą sistemą su nustatyta taikomųjų programų sąsaja	1	...	...
Vidutinis	Protokolo valdomos sąsajos: aktorius atstovauja kitą sistemą sąveikaujant per protokolus tokius kaip TCP/IP	2	...	...
Sudėtingas	Vartotojo sąsajos: aktorius yra asmuo bendraujantis per	3	...	...

	vertotojo sąsają.			
			UAW = svoris * skaičius:	...

Skaičiuojant techninio sudėtingumo faktorių (TCF) dalyvauja projekto komanda. Kiekvienam faktoriui priskiriama jo įtaką atspindinti reikšmė, nuo 0 iki 5 (3 - vidurkis). Pirmiausia skaičiuojamas bendras techninis faktorius (TTF).

14 lentelė. Bendras techninis faktorius

TF	Aprašas	Svoris	Įtaka	Rezultatas
T1	Sistemos pasiskirstymas	2	...	...
T2	Eksploatacinės savybės	1	...	...
T3	Galutinio vartotojo efektyvumas	1	...	...
T4	Vidinio veikimo kompleksiškumas	1	...	...
T5	Perpanaudojamumas	1	...	...
T6	Lengvas instaliavimas	0,5	...	...
T7	Lengvas naudojimas	0,5	...	...
T8	Portatyvumas	2	...	...
T9	Keitimo lengvumas	1	...	...
T10	Konkurencingumas	1	...	...
T11	Saugumas	1	...	...
T12	Tiesioginis perėjimas trečiosioms šalims	1	...	...
T13	Vartotojų apmokymo reikalingumas	1	...	...
T-lygis 0-5 kur 0 = nesvarbus 5 = esminis				
			TTF = svoris * įtaka	...

$$TCF = 0{,}6 + (0{,}01 \cdot TTF) \tag{9}$$

Konstantos apriboja TCF įtaką skaičiuojant UCP: nuo 0,6 (kai įtaka visur 0), iki 1,3 (kai įtaka visur 5).

Aplinkos sudėtingumo faktorius (ECF) skaičiuojamas įvertinant projekto komandos patirtį. Kiekvieno faktoriaus poveikis vertinamas 0 (jokio poveikio), 1 (stiprus neigiamas poveikis) iki 5 (stiprus teigiamas). Pirmiausia skaičiuojamas bendras aplinkos faktorius (ETF).

15 lentelė. Bendras aplinkos faktorius

EF	Aprašas	Svoris	Poveikis	Rezultatas
E1	Pažintis su UML	1,5	...	...
E2	Taikymų patirtis	0,5	...	...
E3	Objektais orientuota patirtis	1	...	...
E4	Pagrindinio analitiko gebėjimai	0,5	...	...
E5	Motyvacija	1	...	...
E6	Stabilūs reikalavimai	2	...	...
E7	Darbuotojai antraeilėse pareigose	-1	...	...
E8	Sudėtinga programavimo kalba	-1	...	...
E9	Žinios apie verslo procesą	0,5	...	...
T-lygis 0-5 kur 0 = nėra žinių 5 = ekspertas				
			ETF = svoris * poveikis:	...

$$ECF = 1{,}4 + (-0{,}03 \cdot ETF) \tag{10}$$

Konstantos apriboja ECF įtaką skaičiuojant UCP: nuo 0,425, iki 1,4 (kai poveikis visur 5).

Gautus rezultatus sudėję į pagrindinę formulę gausime panaudojimo atvejų taškus:

$$UCP = UUCP \cdot TCF \cdot ECF \quad (11)$$

Žmogaus darbo valandų skaičius, reikalingas vienam UCP:

- PF turėtų būti skaičiuojamas remiantis ankstesnių projetų statistika.
- Pvz.: įvykdytas projektas truko 2200 valandų , jo UCP =120. Tai PF =18 žmogaus darbo valdandų.
- Jei istorinių duomenų nėra naudojama reikšmė tarp 15-30 (dažniausiai 20). [6]

$$T = UCP * PF \quad (12)$$

1. 5. 3 Objektinių taškų metodas

Šis metodas, vertindamas informacinę sistema, charakterizuoja tai objektiniais taškais ir suskaido iki konkrečių programinių modulių, vertinant jų sudėtingumą. Sudėtingumas nustatomas pagal parametrų lenteles (kiek lange vaizdų, kiek naudoja DB lentelių).

16 lentelė. Programinių modulių tipai bei sudėtingumas

Forma				Ataskaita			
Virtualių lentelių skaičius (wiew'ai)	Duomenų lentelių skaičius ir šaltinis			Virtualių lentelių skaičius	Duomenų lentelių skaičius ir šaltinis		
	<4 (<2 srvr <3 clnt)	<8 (2/3 srvr 3-5 clnt)	8+ (>3 srvr <5 clnt)		<4 (<2 srvr <3 clnt)	<8 (2/3 srvr 3-5 clnt)	8+ (>3 srvr <5 clnt)
<3	paprastas	paprastas	vidutinis	0 arba 1	paprastas	paprastas	vidutinis
3-7	paprastas	vidutinis	sudėtingas	2 arba 3	paprastas	vidutinis	sudėtingas
>8	vidutinis	sudėtingas	sudėtingas	4 ir daugiau	vidutinis	sudėtingas	sudėtingas

Srvr – sisteminės lentelės, naudojamos ekraninėje formoje arba ataskaitoje.

Clnt – vartotojiškos arba veiklos lentelės, naudojamos ekraninėje formoje arba ataskaitoje.

Po to, kai yra identifikuojama kiekvieno programinio modulio (formos arba ataskaitos) sudėtingumas reikia priskirti svorį. Žemiau esančioje lentelėje pateiktos galimos svorių reikšmės:

17 lentelė. Programinio modulio sudėtingumo klasifikatorius

Objekto tipas	Sudėtingumo koeficientas		
	Paprastas	Vidutinis	Sudėtingas
Forma	1	2	3
Ataskaita	2	5	8
Komponentas	10	10	10

Visi svoriniai įverčiai sudedami ir gaunamas vienas dydis - OP (ObjectPoints).

$$Size = \sum_{i}^{t} OP_i \tag{13}$$

Objektų taškų produktyvumas: PROD

- Matuojami 4-50 objektų taškai/žmogui mėnesių.
- Priklauso nuo kūrėjo gebėjimų. [19]

18 lentelė. Objektų taškų produktyvumo nustatymas

Kūrėjų patirtis ir gebėjimai	Labai žemas	Žemas	Vidutis	Aukštas	Labai aukštas
PROD: objektų taškų vienam asmeniui per mėnesį našumas	4	7	13	25	50

$$Effort = OP/PROD \tag{14}$$

2. PROGRAMINĖS ĮRANGOS KAINOS SKAIČIAVIMO METODŲ ANALIZĖ

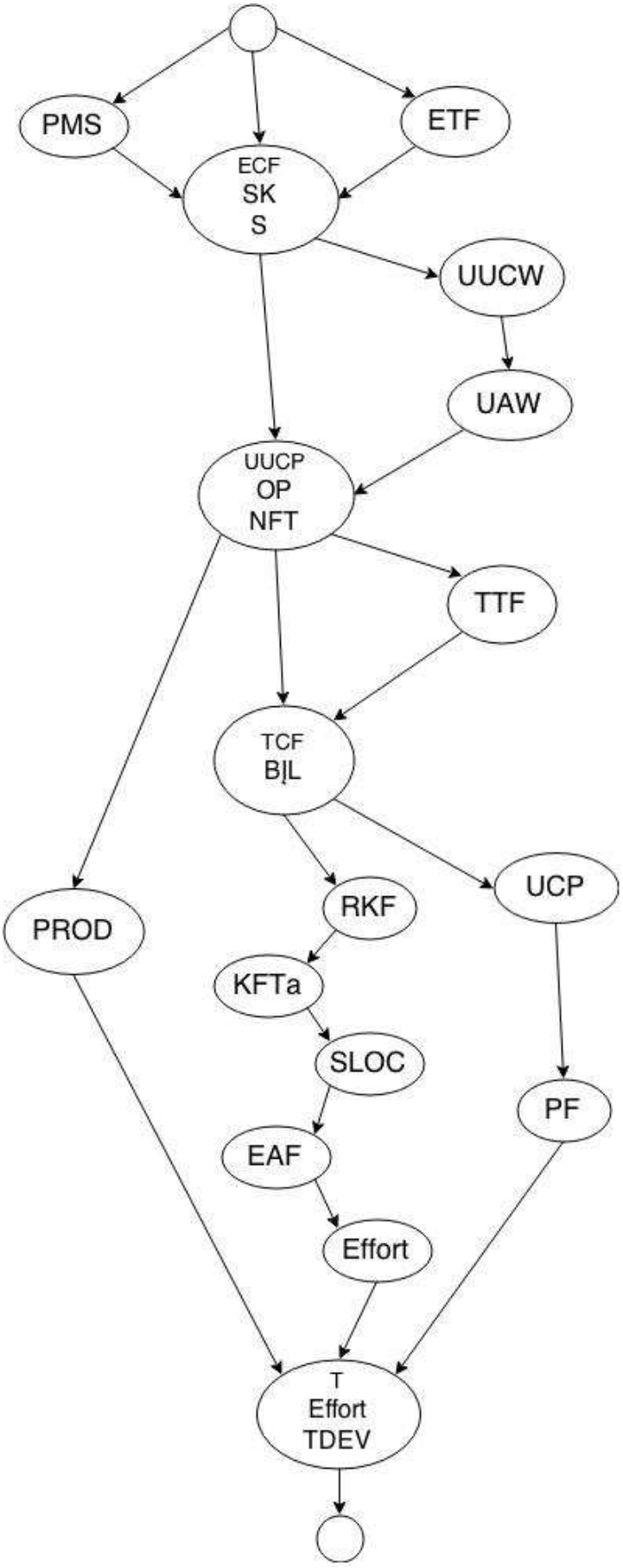
2. 1 Teorinė metodų analizė

2. 1. 1 Euristiniai metodai

Analizuojant dydžio apimties matavimo metodus buvo pasitelktas euristinis metodas. Jis parodo analizuojamų metodų kriterijus. Iš grafo galima matyti metodų skaičiavimo eigą. Sujungus visų analizuojamų metodų skaičiavimo eigas į bendrą grafą, galime pamatyti, kurie kriterijai kartojasi. Tai parodo, kas yra būtina vertinant ir kas įtakoja programinės įrangos kainą.

Dydžio matavimo grafe nagrinėjami funkcinių taškų, panaudos atvejų taškų ir objektų taškų metodai. Funkcinių taškų metodą sudaro tokie kintamieji: duomenų ir tranzakcinis funkcijos sudėtingumas (S), nekoreguoti funkciniai taškai (NFT), bendras įtakos laipsnis (BĮL), reikšmių koregavimo faktorius (RKF), koreguoti funkciniai taškai (KFTa), konvertavimas į kodo eilučių skaičių (SLOC), pastangos koregavimo faktorius (EAF), pastangos (Effort), kūrimo laikas mėnesiais (TDEV). Panaudos atvejų taškų metodą sudaro kintamieji: bendras aplinkos faktorius (ETF), aplinkos sudėtingumo faktorius (ECF), panaudos atvejų svoris (UUCW), aktorių svoris (UAW), nepakoreguoti panaudos atvejų taškai (UUCP), bendras techninis faktorius (TTF), techninio sudėtingumo faktorius (TCF), panaudos atvejų taškai (UCP), žmogaus darbo valandų skaičius vienam panaudos atvejui (PF), žmogaus darbo valandų skaičius (T). Objektų taškų metodą sudaro kintamieji: programinių modulių sudėtingumas (PMS), sudėtingumo klasifikatorius (SK), objektų taškai (OP), objektų taškų produktyvumas (PROD), kūrimo laikas mėnesiais (Effort) (žr. 7 pav.).

Iš šio grafo galima teigti, kad visi metodai vertina aplinkos faktorius, taip pat visi metodai vertina programinės įrangos apimtį analogiškais taškais. Funkcinių taškų metodas ir panaudos atvejų taškų metodas vertina ir techninį faktorių.



7 pav. *Dydžio apimties matavimo metodų grafas*

2. 1. 2 Kitų autorių alikti tyrimai

Vytis Dedonis ir Lina Dovidonienė savo darbe „Programinės įrangos kūrimo kompleksiškumo modelis: eksperimentinio tyrimo rezultatai“ analizuoja funkcinių taškų metodą ir panaudos atvejų taškų metodą. Juos išanalizavę ir modifikavę pasiūlė naują kompleksiškumo modelį.

Sukurtas PĮ kompleksiškumo vertinimo modelis remiasi patobulintu panaudos atvejų taškų analizės metodu, plačiai naudojamu modeliuojant PĮ sistemas, panaudojant universalią modeliavimo kalbą (UML). Sukurtas modelis nuo esamų UCP metodikų skiriasi dviem tyrimo metu papildomai nustatytais PĮ vertinimo kriterijais. Pirma, jame sistemos funkcinis dydis matuojamas ir pagal duomenų ir duomenų bazės struktūros dydį, o tai leidžia „debesijos“ taikymų kūrėjams įsivertinti duomenų kompleksiškumą ir panaudoti jį modeliuojant perkeliamos į „debesiją“ PĮ sistemos architektūrą kartu su duomenu baze. Antra, matuojamos sistemos kompleksiškumo įsivertinimui naudojamas technologijų kompleksiškumo parametras (daugiklis), kuris vertina programavimo įrankių sudėtingumą ir suteikia galimybes PĮ kūrėjams įsivertinti pasirinktų technologijų poveikį bendram sistemos kompleksiškumui.

Funkcinio dydžio matavimas imtas naudoti daug anksčiau, nei atsirado objektinė programavimo kalba UML, todėl jis nėra specialiai pritaikytas PĮ sistemoms, sukurtoms objektus programuojančiomis Java ir kitoms programavimo kalbomis.

FPA matuoja PĮ sistemos funkcinį dydį tik pagal funkcinių transakcijų ir loginių duomenų failų skaičių. Tai yra labai laikui imli metodika (vienai sistemai įvertinti gali prireikti net 12-18 mėn. analitinio darbo), todėl FPA naudojama tik didžiausių, standartizuotų sistemų analizei.

Panaudojimo atvejų taškų (UCP) metodika buvo sukurta kaip alternatyva FPA metodikai ir, priešingai nei FPA metodikos, skirta PĮ analizei pradiniame PĮ kūrimo etape.

Pagal UCP metodiką vietoj funkcinio dydžio PĮ sistemos dydis matuojamas pagal funkcinius PĮ reikalavimus, išreikštus PĮ sistemos aktorių ir panaudojimo atvejų skaičiumi ir jiems suteiktomis svorinėmis vertėmis (žr. UCP vertinimo lentelėje Priede). UCP modelyje PĮ funkcinis dydis koreguojamas papildomais veiksniais (daugikliais) – techninių veiksnių ir aplinkos parametrais, kurie nėra įtraukti į FPA modelį. Nors pradiniame FPA modelyje dalis techninių veiksnių, kuriuos apima UCP vertinimas, vertinimas buvo įtrauktas, tačiau jis nebuvo standartizuotas. Todėl tolesnis šių veiksnių tyrimas vyksta naudojant UCP metodiką, kuri taikoma ir papildant FPA metodikas. [6]

Šarūnas Gervė savo darbe „Funkcinių taškų analizės metodų tyrimas“ išsako tokius pastebėjimus apie funkcinių taškų metodą.

IFPUG metodas kritikuojamas dėl komponentų sudėtingumo įvertinimo skalės nepakankamumo dabartiniams vertinimo poreikiams, nekoreguotų funkcinių taškų lentelės reikšmių tinkamumo, vidinių loginių failų nustatymo taisyklių tikslumo, bendrųjų sistemos charakteristikų suprantamumo ir jų vertinimo dydžio vienodumo, nepakankamo atsižvelgimo į vidinį transakcijų vykdymo sudėtingumą, netinkamo pavienių ir kompleksinių sistemų vertinimo skirtumo, bei blogesnio didelių sistemų vertinimo.

Pirmos kartos funkcinio dydžio matavimo metodas IFPUG yra tinkamesnis matuoti valdymo informacines sistemas.

Pavyzdžiui, naudojant tokias kalbas, kaip UML, sunku pritaikyti dydžio matavimo sąvokas. Taip pat pirmos kartos metodai netinka realaus laiko sistemų vertinimui.

Funkcinio dydžio matavimas yra svarbus programinės įrangos matavime ir vertinime. Tai yra esminis matavimo ir vertinimo patobulinimas. Funkciniai matavimai: nepriklauso nuo programavimo technologijos, yra efektyvūs ankstyvose programinės įrangos gyvavimo ciklo fazėse (reikalavimų formulavimo fazėje), yra gerai dokumentuoti ir apibrėžti, juos palaiko standartai, tarptautinės naudotojų grupės, yra pakankamai patikimi ir tikslūs, yra naudingi derybose su naudotojais ir valdymui. Bet funkciniai matavimai turi ir trūkumų: jie yra semantiškai sudėtingi, ne visi terminai lengvai suprantami, turi daug žingsnių, reikalingos didelės pastangos, norint tapti sertifikuotu vertintoju, nėra priemonių, automatizuojančių funkcinių taškų skaičiavimą, koregavimo faktoriuose galimas ženklus subjektyvumas. Funkcinių taškų skaičiavimas gali reikalauti daug laiko ir būti brangus. Tiksliam skaičiavimui reikalingas atestuotas funkcinių taškų specialistas, bet tarp organizacijų funkcinių taškų metrikos laikomos patikimesnėmis negu grįstos kodo eilučių skaičiumi, ypač kai naudojamos įvairios programavimo kalbos arba vertinamas produktyvumas. [8]

2. 2 Praktinė metodų analizė

Praktinei metodų analizei atlikti buvo pasirinkta informacinė sistema ir paskaičiuota jos kaina analizuotais metodais. Pasirinkta elektroninės parduotuvės informacinė sistema. Jos kaina buvo skaičiuojama panaudojimo atvejų taškų, funkcinių taškų ir objektų taškų metodais.

2. 2. 1 Naudojamos informacinės sistemos apibūdinimas

Praktinei metodų analizei bus naudojama elektroninės parduotuvės www.kavaverslui.lt informacinė sistema. Sistemoje yra galimybė įsigyti prekes ar užsisakyti paslaugas. Prie kiekvienos prekės pateikiamas aprašymas, taip pat siūlomos rekomenduojamos prekės. Prekės ir paslaugos surūšiuotos pagal kategorijas. Prekes taip pat galima rūšiuoti pagal jų naujumą bei kainą. Pateikiami specialūs pasiūlymai bei informacija apie įmonę. Taip pat sistemoje pateikiama informacija apie atsiskaitymą, prekių pristatymą. Taip pat sistemoje galima rasti prekių katalogą .pdf formatu. Informacinėje sistemoje taip pat yra informacija apie taisykles, dažniausiai užduodamus klausimus (D.U.K) bei kontaktai.

Sistemoje galimos vartotojų rolės: svečias, klientas, administratorius. Svečias gali peržiūrėti informaciją ir atlikti registraciją. Klientas gali prisijungti. Prisijungęs klientas gali užsakyti prekes, redaguoti savo prekių krepšelį bei savo profilį. Administratorius gali administruoti prekes, jas kurti, šalinti ir redaguoti. Taip pat gali administruoti vartotojus, keisti jų roles, juos šalinti. Administratorius gali keisti informacinės sistemos turinį, jį šalinti, redaguoti bei kurti naują. Taip pat jis gali keisti sistemos nustatymus. Administratorius administruoja reklamas esančias sistemoje.

Klientai gali būti fiziniai ir juridiniai. Nuo to jų galimos funkcijos nesikeičia. Tik juridiniams asmenims yra išrašoma išankstinė sąskaita.

Klientų užsakymai saugomi ne tik duomenų bazėje, bet taip pat ir failuose .xml formatu. Taip pat sistemoje yra integracijos su kitomis sistemomis. Sistemoje integruota Paysera mokėjimų sistema. Taip pat informacinėje sistemoje yra integruota Google Maps sistema, kurioje nurodoma įmonės buvimo vieta.

Informacinė sistema patogi ir lengva vartotojams. Patogus ir aiškus prekių išdėstymas. Sistemoje pateikta visa reikalinga vartotojui informacija.

Sistemos funkcijos paprastos ir aiškios. Vartotojui norint prisiregistruoti reikia nurodyti ar jis fizinis ar juridinis asmuo. Jei tai fizinis asmuo jis turi įvesti savo vardą, pavardę, el. paštą, savo adresą, telefono numerį bei nurodyti slaptažodį. Jei vartotojas juridinis asmuo jis turi nurodyti įmonės pavadinimą, atsakingo asmens vardą ir pavardę, įmonės kodą, įmonės PVM kodą, adresą, el.paštą, telefono numerį ir slaptažodį. Jei registruotas vartotojas nepamena savo prisijungimo slaptažodžio yra galimybė priminti slaptažodį įvedus el.paštą nurodytą registracijos metu. Registruotas vartotojas prisijungęs su savo duomenimis gali redaguoti savo profilio informaciją. Tam reikia tik suvesti naują informaciją ir ją išsaugoti. Taip pat prisijungęs vartotojas gali matyti savo pirkinių krepšelį. Jis gali šalinti prekes iš krepšelio. Registruotas vartotojas prisijungęs gali prekes įsidėti į krepšelį ir jas nusipirkti. Taip pat peržiūrint konkrečią prekę apačioje yra rodomos rekomenduojamos prekės. Informacinėje sistemoje vartotojai gali atlikti prekių paiešką pagal pageidaujamą žodį. Paieška palengvina ir pagreitina norimų prekių paiešką.

Administratorius prisijungęs prie savo paskyros gali administruoti visą sistemą. Jam tereikia pasirinkti iš kategorijų sąrašo ką nori administruoti. Pasirinkus jis gali viska keisti, redaguoti, šalinti bei kurti.

Sistemos vidinė struktūra ir panaudos atvejų diagrama pateikta prieduose (žr. 1 priedas).

2. 2. 2 Kaina apskaičiuota panaudos atvejų metodu

Skaičiuojant kainą panaudos atvejų taškų metodu primiausia reikia apskaičiuoti nepakoreguotus panaudojimo atvejų taškus (UUCP). Juos gausime įvertinę panaudojimo atvejų (UUCW) ir aktorių (UAW) svorius. Nustatę svorius juos sudauginame su kiekiu ir sudedame gautus rezultatus. Panaudojimo atvejų svorius (UUCW) gauname 110, nes sistemoje yra 20 paprastų panaudos atvejų ir vienas vidutinis panaudos atvejis. Panaudojimo atvejų svorių įvertinimo lentelė prieduose (žr. 2 priedas).

Tada taip pat įvertiname aktorių svorius ir taip apskaičiuojame aktorių svorius (UAW). Aktorių svorius (UAW) gauname 9, nes sistemoje yra trys sudėtingi aktoriai. Aktorių svorių įvertinimo lentelė prieduose (žr. 2 priedas).

Norint apskaičiuoti nepakoreguotus panaudojimo atvejų taškus sudedame panudojimo atvejų svorius su aktorių svoriais, kuriuos apskaičiavome anksčiau. Sudėję gauname rezultatą 119.

$$UUCP = UUCW + UAW = 110 + 9 = 119$$

Toliau skaičiuojamas bendras techninis faktorius. Jam apskaičiuoti reikia įvertinti 13 techninių faktorių. Juos įvertiname nustatydami jų įtaką nuo 0 iki 5. Bendrojo techninio faktoriaus įvertinimo lentelė prieduose (žr. 2 priedas).

Įvertinus bendruosius techninius faktorius ir juos apskaičiavus gautą rezultatą dedame į 9 formulę ir gauname techninio sudėtingumo faktorių. Gautas rezultatas 0,76.

$$TCF = 0,6 + (0,01 * 16) = 0,76$$

Tam kad gautume aplinkos faktoriaus sudėtingumą reikia įvertinti bendrojo aplinkos faktoriaus poveikį, jų yra 9. Bendrojo aplinkos faktoriaus įvertinimo lentelė prieduose (žr. 2 priedas).

Apskaičiavus bendrojo aplinkos faktoriaus poveikį galime apskaičiuoti aplinkos faktoriaus sudėtingumą pagal 10 formulę. Gauname rezultatą 0,59.

$$ECF = 1,4 + (-0,03 * 27) = 0,59$$

Norint apskaičiuoti panaudos atvejų taškus reikia sudauginti nepakoreguotus panaudojimo atvejų taškus, techninio sudėtingumo faktorių ir aplikos sudėtingumo faktorių. Viską apskaičiavę pagal 11 formulę gauname rezultatą 53,3.

$$UCP= 119*0,76*0,59=53,3$$

Toliau apskaičiuojame žmogaus darbo valandų skaičių pagal 12 formulę. Gauname rezultatą 1066 žmogaus darbo valandų.

$$T=53,3*20=1066$$

Panaudos atvejų teškų metodas įvertina panaudos atvejus, techninius bei aplinkos faktorius. Šis metodas neįvertina sistemos funkcionalumo, funkcijų sudėtingumo.

Vertinant techninius ir aplinkos faktorius, nagrinėjant informacines sistemas, kai kurie faktoriai nereikalingi. Metodas taip pat galėtų labiau analizuoti sistemos charakteristikas.

Panaudos atvejų taškų metodas neatskleidžia sistemos vidinės struktūros. Metodas nenagrinėja integracijos su kitomis sistemomis.

Šis metodas negali būti naudojamas pradiniame kūrimo etape.

Aktorių identifikavimui reikia techninių detalių, pavyzdžiui: kokius protokolus aktorius naudoja. Vertinant šio metodo aktorių bei panaudojimo atvejų sudėtingumą, jie nėra tinkamai įvertinami. Reiktų labiau atsižvelgti į jų sudėtingumus.

Problema panaudos atvejų taškų metode su techniniais faktoriais ta, kad kai kurie iš jų neturi įtakos bendram projektui.

Nagrinėjamas metodas neįvertina šiuolaikinių kūrimo technologijų. Todėl sudėtinga tinkamai įvertinti naujas kuriamas sistemas.

Panaudojimo atvejų taškų metodo problema ta, kad norint jį skaičiuoti turi būti aprašyti visi panaudojimo atvejai.

2. 2. 3 Kainos apskaičivimas funkcinių taškų metodu

Suskaičiavus išorinių įvedimų, išorinių išvesčių, išorinių užklausų, vidinių loginių failų, išorinių interfeisų failų kiekį ir nustačius jų sudėtingumą apskaičiuojame nekoreguotus funkcinius taškus. Gauname rezultatą 396. Funkcinių taškų skaičiavimo lentelė prieduose (žr. 2 priedas).

Tada reikia įvertinti 14 bendųjų sistemos charakteristikų sudėtingumo laipsnius ir juos sudėti. Gauname bendrą įtakos laipsnį ir jo rezultatą 40. Bendrųjų sistemos charakteristikų sudėtingumo įvertinimo lentelė prieduose (žr. 2 priedas).

$$BĮL = 3+5+4+2+5+4+4+3+2+1+0+3+1+3=40$$

Tada skaičiuojams reikšmių koregavimo faktorius pagal 2 formulę ir jį gauname 1,05.

$$RKF=40*0,01+0,65=1,05$$

Tada skaičiuojami koreguoti funkciniai taškai pagal 3 formulę ir gauname rezultatą 415,8

$$KFTa=396*1,05=415,8$$

Toliau norint skaičiuoti reikia funkcinius taškus konvertuoti į kodo eilučių skaičių (žr. 11 lentelė). Pagal 4 formulę apskaičiuojame kodo eilučių skaičių, gauname 0,62. Pagal 6 formulę apskaičiuojame pastangos koregavimo koeficientą, gauname 1,43. Norint apskaičiuoti pastangas naudojame 5 formulę ir gauname rezultatą 3,8. Tada skaičiuojame žmogaus darbą mėnesiais pagal 7 formulę. Gauname galutinį rezultatą 4,15 žmogaus darbo mėnesių.

$$SLOC = 16 * 16/415,8=0,62$$

$$EAF = 1,3*1,1= 1,43$$

$$EFFORT = 1,43*3,2*(0,62)^{0,38}=3,8$$

$$TDEV = 2,5 * (3,8)^{0,38}=4,15$$

Funkcinių taškų metodas vertina sistemos vidinius veiksmus. Taip pat gana detaliai vertina sistemos charakteristikas. Metodas vertina duomenų bazės lentelių kiekį, bet nevertina lentelių naudojimo ar jų sudėtingumo.

Šis metodas kaip ir panaudos atvejų taškų metodas neįvertina sistemos funkcijų sudėtingumo. Šiame metode nėra vertinama aplinkos faktorių.

Šis metodas neįvertina šiuolaikinių technologijų.

Funkcinių taškų metodas nėra taikomas visų tipų programinei įrangai, pavyzdžiui: jis nefiksuoja visų funkcinių savybių realaus laiko programinei įrangai.

Šiam metodui reikalingi subjektyvūs vertinimai su daug dalyvaujančių sprendimų.

Dauguma pastangų ir išlaidų skaičiavimo modeliai yra grindžiami kodo eilučių skaičiumi, todėl funkciniai taškai yra konvertuojami į kodo eilučių skaičių. Vertinant šiuolaikines sistemas kodo eilučių skaičius neatspindi sistemos galimybių, bei funkcionalumo.

Kad metodas būtų tinkamai įvertintas tam reikia kvalifikuotų specialistų, kurie sugebėtų tinkamai įvertinti programinę įrangą.

Funkcinių taškų modelis tinkamas dideliems projektams, o mažiems jis nenaudingas.

2. 2. 4 Kainos skaičiavimas objektų taškų metodu

Įvertinus sistemos formas, ataskaitas ir komponentus ir įvertinus jų sudėtingumus gauname: 9 paprastas formas, 6 paprastas ataskaitas ir 3 komponentus.

Iš lentelės (žr. 17 lentelė) suskaičiuojame sudėtingumo koeficientus ir juos sudedame sudauginę su jų kiekiais pagal 13 formulę.

$$Size = 9*1 + 6*2 + 3*10 = 51$$

Tada įvertiname objektų taškų produktyvumą (žr. 18 lentelė) ir apskaičiuojame projekto kūrimo pastangas pagal 14 formulę. Gauname rezultatą 3,15 žmogaus darbo mėnesių.

$$Effort = 51/13=3,92$$

Šitas metodas silpniausiai vertina sistemą iš nagrinėtų metodų. Jis nevertina sistemos funkcijų. Taip pat nevertina nei sistemos charakteristikų, nei aplinkos faktorių, kurie įtakoja programinės įrangos kainą.

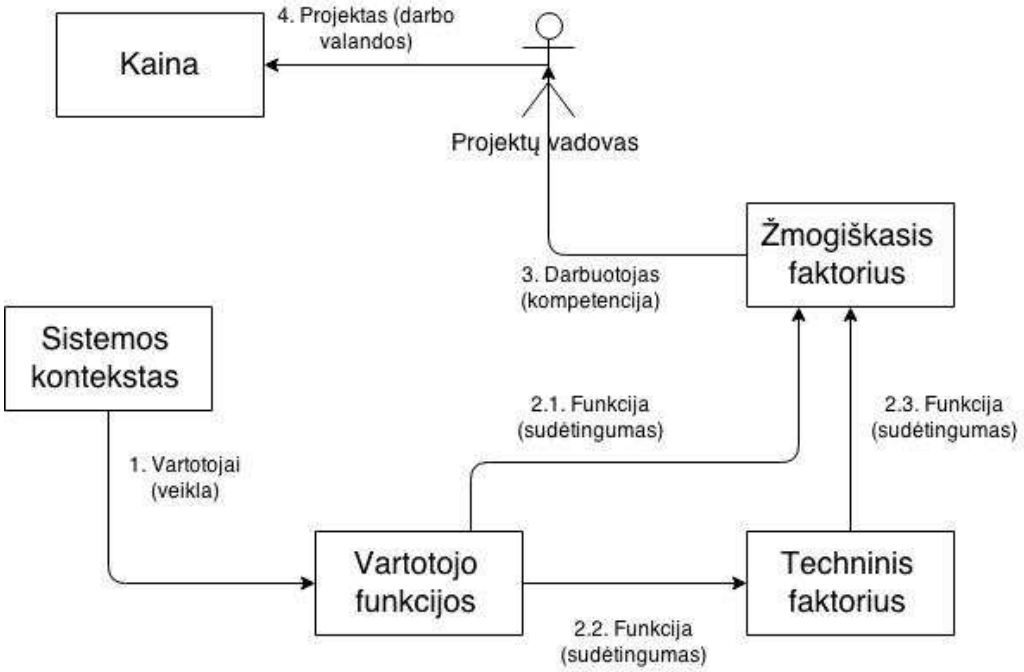
Šis metodas nėra tinkamas visų tipų programinei įrangai. Objektų taškų metodas neįvertina šiuolaikinių projektų, jis nevertina naujų naudojamų technologijų, todėl jis nėra tinkamas apskaičiuoti tikslią kainą.

Nagrinėjamas metodas nevertina vidinės duomenų struktūros. Metode nėra skaičiuojami aplinkos bei sistemos faktoriai, nuo kurių priklauso sistemos sudėtingumas, funkcionalumas ir galimybės. Todėl metodas neįvertina kainą įtakojančių svarbių savybių.

3. KAINOS SKAIČIAVIMO METODAS FUNKCIONALUMO ĮVERTINIMUI

3. 1 Programinės įrangos funkcionalumo įvertinimo metodo struktūra

Bendradarbiavimo diagrama parodo sąveiką tarp sistemos objektų, išskirdama objektus, ryšius tarp jų ir ryšio metu perduodamus pranešimus.



8 pav. *Bendradarbiavimo diagrama*

Paveiksle (žr. 8 pav.) pateikiama metodo bendradarbiavimo diagrama. Joje parodomi objektai ir ryšiai tarp jų. Bendradarbiavimo diagramoje yra tokie objektai: sistemos kontekstas, vartotojo funkcijos, techninis faktorius, žmogiškasis faktorius ir kaina. Objektus jungia tokie ryšiai: vartotojai, funkcija, darbuotojas ir projektas.

3. 2 Programinės įrangos funkcionalumo įvertinimo metodo aprašymas

Atlikus praktinį sistemos kainos skaičiavimo metodų tyrimą, paaiškėjo, kad nė vienas tyrinėjamas metodas nesigilina į sistemos funkcijų sudėtingumą. Todėl naujame kuriamame metode bus nustatomas panaudos atvejų aktorių sudėtingumas, bei kiekvienos funkcijos sudėtingumas. Tam kad nustatytume funkcijų sudėtingumą bus reikalinga funkcijos veiklos diagrama, kurioje bus įvertinti diagramos elementai ir jų sudėtingumas.

Atlikus tyrimą buvo pastebėta, kad universaliausias ir lengviausiai taikomas metodas yra panaudojimo atvejų taškų. Jis vertina techninius bei aplinkos faktorius, ko nedaro kiti metodai. Jis lengvai suprantamas, kad jį būtų galima panaudoti nereikia būti tos srities specialistu.

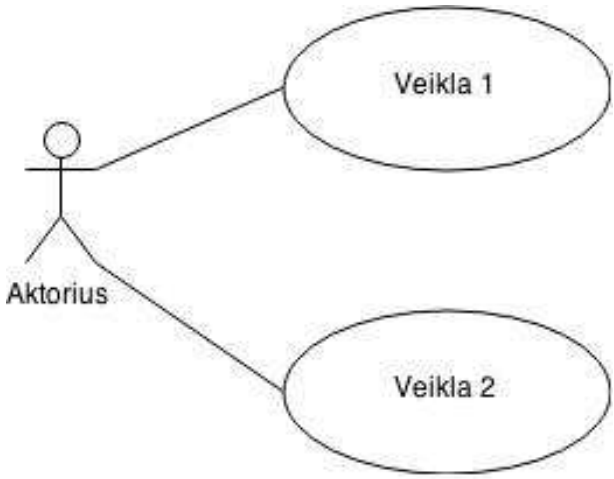
Lentelėje (žr. 19 lentelė) pateikiama panaudos atvejų aktorių sudėtingumo nustatymo aprašymai ir pavyzdžiai. Aktoriai vertinami keturiais sudėtingumo lygiais.

19 lentelė. Panaudos atvejų aktorių sudėtingumo nustatymas

Elementas	Sudėtingumas	Aprašymas	Pavyzdys
Aktorius (žr. 9 pav.)	Paprastas	Nepaveldi kitų aktorių savybių, ir gali atlikti tik paprastus veiksmus.	Informacinės sistemos lankytojas. Jis gali tik peržiūrėti informaciją esančią sistemoje ir atlikti registraciją. Atlikęs registraciją jis tampa sistemos klientu ir tada aktorius jau laikomas vidutiniu.
	Vidutinis	Gali atlikti daugiau nei tris veiksmus, tarp kurių yra sudėtingų veiksmų. Jis gali paveldėti kitų paprastų aktorių savybes.	Pavyzdžiui sistemos klientas, kuris prisijungęs gali peržiūrėti daugiau sistemos infomacijos nei paprastas aktorius.
	Sudėtingas	Jis paveldi kitų aktorių (gali būti paprastų ar vidutinių) savybes. Jo atliekamos veiklos gali būti sudėtingos, kurios gali keisti sistemos įrašus, pavyzdžiui administruoti kitas veiklas.	Pavyzdžiui sistemos klientas, kuris gali administruoti savo profilį ir atlikti kitas sudėtingas veiklas. Taip pat kurios nors vienos srities administratorius.
	Labai sudėtingas	Jis gali atlikti visas įmanomas arba beveik visas veiklas. Gali administruoti visos sistemos funkcijas. Paveldi visų kitų aktorių savybes.	Pavyzdžiui sistemos administratorius.

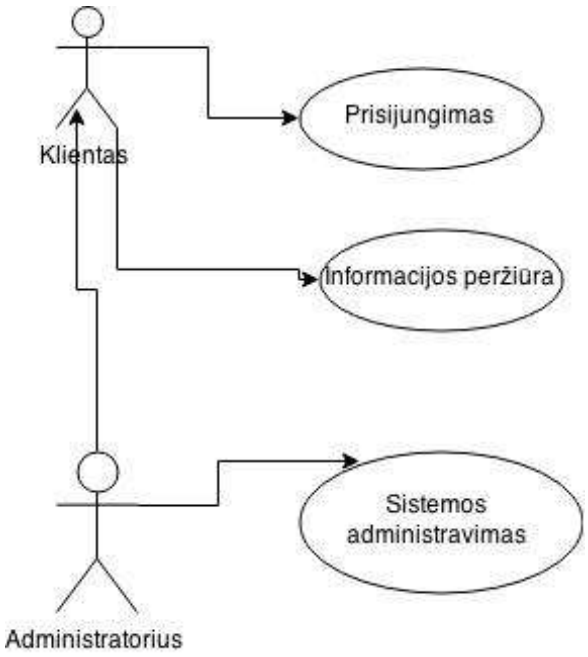
Aktorius (angl. actors) - simbolizuoja bet kokią esybę (ar esybes), atliekančią konkrečius vaidmenis turimoje sistemoje. Skirtingi aktoriaus simbolizuojami vaidmenys yra tikrieji sistemoje esančių vartotojų veiklos vaidmenys. Aktorius panaudos atvejų diagramoje sąveikauja su panaudos atvejais (žr. 9 pav.).

Panaudos atvejis (angl. use case) - atvaizduoja veiksmus atliekamus vieno ar daugiau aktorių siekiant konkretaus tikslo (žr. 9 pav.).



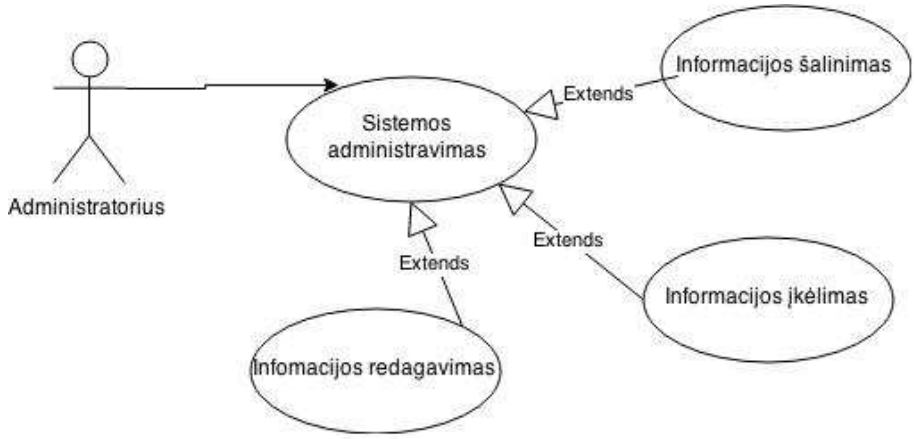
9 pav. *Aktorius ir veikla*

Aktorių paveldimumas (angl. inheritance) – aktorius paveldi kito aktoriaus savybes (žr. 10 pav.).



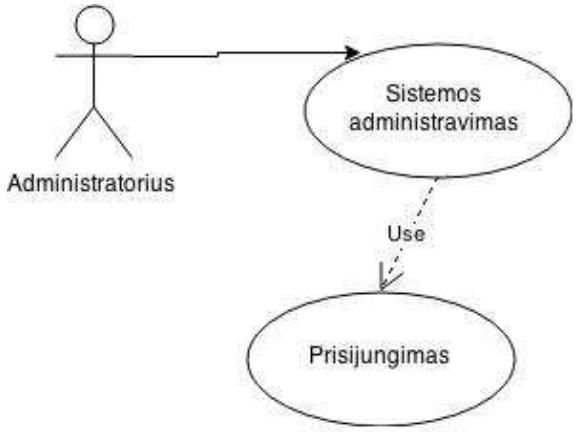
10 pav. *Aktorių paveldėjimas*

Išplėtimo ryšys (angl. extend) - Sąryšis, nurodantis, kaip išplėsto panaudos atvejo elgsena priklauso nuo bazinio panaudos atvejo elgsenos. Bazinis panaudos atvejis nepriklauso nuo išplėtimo panaudos atvejo. Esant išplėtimo sąryšiui tarp dviejų panaudos atvejų, žemesnysis panaudos atvejis prijungiamas prie jau esamo viršesniojo funkcionalumo ir charakteristikų (žr. 11 pav.).



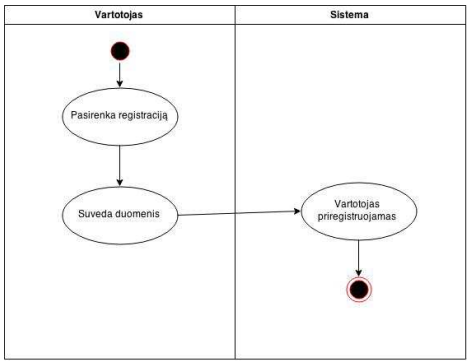
11 pav. *Išplėtimo ryšys*

Įtraukimo ryšys (angl. include) - Sąryšis, nurodantis, kaip bazinio panaudos atvejo elgsena priklauso nuo įtraukiamo panaudos atvejo elgsenos. Bazinis panaudos atvejis priklauso nuo įtraukiamo panaudos atvejo. Esant įtraukimo sąryšiui, panaudos atvejis kaip savo veiklos proceso srauto dalį „įtraukia“ funkcionalumą, aprašytą kitame panaudos atvejyje (žr. 12 pav.).



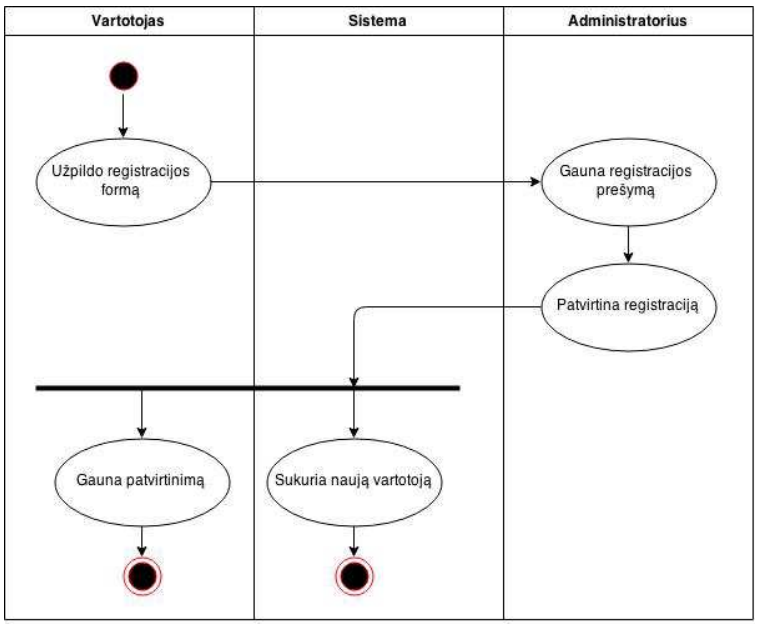
12 pav. *Įtraukimo ryšys*

Juosta (angl. swimlane) - tai būdas, kaip vienoje juostoje sugrupuoti to pačio aktoriaus atliekamas veiklas (žr. 13 pav.).



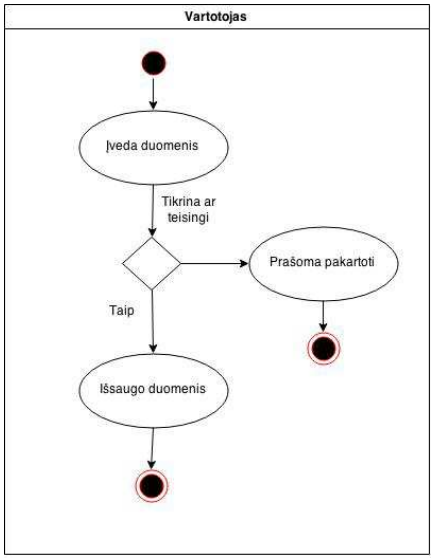
13 pav. *Juosta*

Lygiagretumas (angl. fork) - išsišakojimas parodo konkurencinio srauto pradžią. Išskirsto srautą į kelis lygiagrečius srautus (žr. 14 pav.).



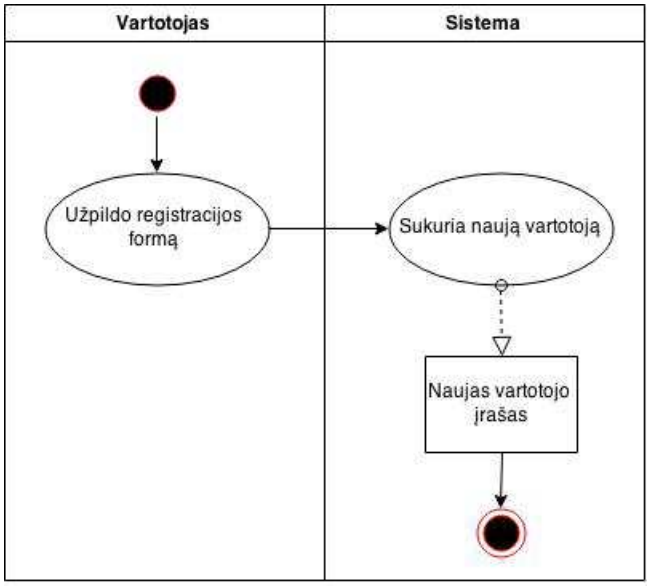
14 pav. *Lygiagretumas*

Salyga (angl. decision) - sprendimas, remiantis nurodyta sąlyga, pasirenka tarp išeinančių srautų, kas leidžia kontroliuoti srautą, jei reikalingos apsauginės sąlygos (žr. 15 pav.).



15 pav. *Salyga*

Objektas (angl. object) - objekto būsena apibūdina duomenų srautą veikloje ir žymi objektų judejimą veiklos diagramoje (žr. 16 pav.).



16 pav. *Objektas*

Lentelėje (žr. 20 lentelė) pateikiami veiklos diagramų elementų sudėtingumo nustatymo aprašymai ir pavyzdžiai. Diagramos elementai vertinami keturiais sudėtingumo lygiais. Veiklos diagramose vertinami tokie elementai: veikla, salyga, juosta, lygiagretumas, objektas.

20 lentelė. Veiklos diagramų elementų sudėtingumo nustatymas

Elementas	Sudėtingumas	Aprašymas	Pavyzdys
Veikla (žr. 9 pav.)	Paprastas	Veiklos, kuriose peržiūrima informacija arba patvirtinamas veiksmas.	Pavyzdžiui prisijungiama, peržiūrima prekės informacija.
	Vidutinis	Veiklos, kuriose įvedami duomenys arba atliekamas duomenų tikrinimas.	Pavyzdžiui registracijos duomenų suvedimas.
	Sudėtingas	Veiklos, kuriose yra atliekami sistemos pakeitimai, sukuriami ar šalinami įrašai.	Pavyzdžiui registracijos patvirtinimas, duomenų keitimas.

	Labai sudėtingas	-	-
Salyga (žr. 15 pav.)	Paprastas	-	-
	Vidutinis	Salyga, kuri tikrina patvirtinimą ar įrašo egzistavimą.	Pavyzdžiui ar sandėlyje yra prekė. Ar tikrai norima atlikti veiksmą.
	Sudėtingas	Salyga tikrinanti duomenis, failus, įrašus laikoma sudėtinga.	Pavyzdžiui tikrina ar suvesti leidžiami simboliai.
	Labai sudėtingas	Salyga tikrinanti saugumą laikoma labai sudėtinga.	Pavyzdžiui tikrina įvestą saugos kodą.
Juosta (žr. 13 pav.)	Paprastas	1 juosta	Veikla vykdoma tik pas vartotoją.
	Vidutinis	2 juostos	Veikla vykdoma pas vartotoją ir sistemą.
	Sudėtingas	3 juostos	Veikla vykdoma pas vartotoją, sistemą ir administratorių.
	Labai sudėtingas	Daugiau kaip 3 juostos	Veikla vykdoma pas vartotoją, sistemą, administratorių ir duomenų bazėje.
Lygiagretumai (žr. 14 pav.)	Paprastas	-	-
	Vidutinis	Jei lygiagrečiai vykdomos dvi veiklos vienoje juostoje.	Priimant užsakymą patikrinama ar yra prekė ir patikrinama ar vartotojas registruotas.
	Sudėtingas	Jei lygiaigrečiai vykdomos dvi veiklos skirtingose juostose. Jei lygiagrečiai vykdomos trys veiklos.	Po registracijos patvirtinimo vartotojas gauna patvirtinimą ir taip pat duomenų bazėje sukuriamas naujas vartotojo įrašas.
	Labai sudėtingas	Jei vykdomos daugiau kaip 3 veiklos lygiagrečiai.	Registruojantis sistema tikrina ar neegzistuoja įvestas el.paštas, tikrina ar įvestas slaptažodis sutampa su pakartotinai įvestu slaptažodziu, tikrina ar el.paštas atitinka savo struktūrą.
Objektas (žr. 16 pav.)	Paprastas	-	-
	Vidutinis	Jei funkcijoje yra sukuriami objektai.	Sukuriamas naujos prekės įrašas duomenų bazėje.
	Sudėtingas	-	-
	Labai sudėtingas	-	-

21 lentelė. Diagramos elementų sudėtingumas

Kriterijai	Paprastas	Vidutinis	Sudėtingas	Labai sudėtingas
Aktorių paveldimumas (žr. 10 pav.)	-	+	+	+
Juosta	1	2	3	>3
Lygiagretumas	-	2 veiklos	2 veiklos skirtingose juostose	>3 veiklos
Salyga	-	Tikrina egzistavimą	Tikrina duomenis	Tikrina saugumą
Objektas	-	-	+	+

Įvertinę elementų sudėtingumus priskiriame jiems svorius.

22 lentelė. Sudėtingumo svoriai

Sudėtingumas	Paprastas	Vidutinis	Sudėtingas	Labai sudėtingas
Svoris	1	2	3	4

Norint įvertinti panaudojimo atvejų sudėtingumą (PAS) turime įvertinti panaudos atvejų aktorius. Įvertinus jų sudėtingumus priskiriame svorius ir juos sudedame, taip gauname panaudojimo atvejų sudėtingumo svorį (PASS).

Vertinant sistemos funkcionalumą reikalinga visų funkcijų veiklos diagramos. Taikant šį metodą galima vertinti tik pagrindines funkcijas, bet įvertinus kuo daugiau funkcijų tuo kaina bus tikslesnė. Turėdami veiklos diagramas įvertiname joje esančių elementų sudėtingumus, taip gausime veiklos diagramos sudėtingumą (VDS). Įvertinus visų elementų sudėtingumumus priskiriame jiems svorius, taip gauname veiklos diagramos sudėtingumo svorį (VDSS). Tada nustatome pačios vienos funkcijos sudėtingumą (VFS). Jei VDS gauname <11 - paprastas, 11-14 – vidutinis, 15-19 – sudėtingas, >19 – labai sudėtingas ir prisikiriame svorius.

23 lentelė. Vienos funkcijos sudėtingumo nustatymas

Sudėtingumas	Paprastas	Vidutis	Sudėtingas	Labai sudėtingas
Veiklos diagramos sudėtingumas	<11	11-14	15-19	>19

Norėdami gauti funkcionalumo sudėtingumą (FS) turime sudėti visus gautus veiklų diagramų sudėtingumo svorius.

$$PAS = \sum PASS \quad (15)$$

$$VDS = \sum VDSS \quad (16)$$

$$FS = \sum VFS \quad (17)$$

Norint gerai įvertinti programinės įrangos kainą reikia atsižvelgti į komandos faktorius. Komandos narių patirits, kompetencija, sugebėjimas greitai orientuotis projekto kūrime turi įtakos kainai. Norint apskaičiuoti komandos faktorių (KF) turime juos įvertinti ir įvertinti komandos faktoriaus įtaką. Įtaka vertinama nuo 0 iki 5. 0 – neturi jokios įtakos, 5 – labai stipri įtaka. Įvertinę įtaką, jas sudedame ir gauname komandos faktoriaus įtaką (KFĮ). Komandos faktorius apskaičiuojamas pagal 18 formulę. Formulė naudojama remiantis panaudos atvejų taškų metodo aplinkos faktoriaus skaičiavimu (žr 1.6.2 skyrius).

$$KF = KFĮ *(- 0,03) + 1,4 \quad (18)$$

24 lentelė. Komandos faktorius

Faktorius	Įtakos aprašymas	Įtaka
Taikymų patirtis	0-neturi įtakos, 5-turi svarbią įtaką	...
Stabilūs reikalavimai	1- reikalavimai nekinta nuo projekto, 2 - kinta neesminiai reikalavimai, 3 - reikalavimai kinta dalinai, 4 - reikalavimai kinta didžiają projekto dalį, 5 - pradžios kinta esminiai reikalavimai viso projekto metu	...
Žinios apie verslo procesą	0-neturi įtakos, 5-turi svarbią įtaką	...
Aiškūs reikalavimai	0 - tikslūs reikalavimai ir funkcijos, 3-aiškios funkcijos, bet neapibrėžti reikalavimai, 4 - nėra tikslių reikalavimų, 5 - nėra reikalavimų	...
Komandos bendradarbiavimas	0-neturi įtakos, 5-turi svarbią įtaką	...
Greita reakcija	0-neturi įtakos, 5-turi svarbią įtaką	...

Išankstinis numanymas	0-neturi įtakos, 5-turi svarbią įtaką	...
Problemos supratimas	0-neturi įtakos, 5-turi svarbią įtaką	...
Patirtis	0-nėra patirties, 3-5metai patirties, 5-10metų patirties	...
Naujos žinios	0-nesuprantama nauja tema, 3-vidutiniškai suprantama, 5-puikiai suprantama nauja projekto tema	...
Profesinės žinios	0-neišmano savo srities, 1-prastos profesinės žinios, 3-vidutinės žinios, 5- puikios savo srities žinios	...
Kompetencija	0-ji nedaro įtakos, 5-ji daro didelę įtaką projektui	...
Dizainerio poreikis	0-nereikia, 3-reikia dalinai, 5-reikia nuolatos	...
Projektuotojo poreikis	0-nereikia, 3-reikia dalinai, 5-reikia nuolatos	...
Konsultacijų poreikis	0-nereikia, 3-reikia dalinai, 5-reikia nuolatos	...
Apmokymų poreikis	0-nereikia, 3-reikia dalinai, 5-reikia nuolatos	...

Programinės įrangos kainai taip pat daug įtakos turi sistemos charakteristikos (SC). Todėl skaičiuojant kainą turime įvertinti sistemos charakteristikų įtaką. Įtaka vertinama nuo 0 iki 5. 0 – neturi jokios įtakos, 5 – labai stipri įtaka. Įtakas taip pat sudedame ir gauname sistemos charakteristikų įtaką (SCĮ). Sistemos charakteistikos apskaičiuojamos pagal 19 fomulę. Formulė naudojama remiantis panaudos atvejų taškų metodo techninio faktoriaus skaičiavimu (žr 1.6.2 skyrius).

$$SC = SCĮ * 0,01 + 0,6 \quad (19)$$

25 lentelė. Sistemos charakteristikos

Sistemos charakteristika	Įtakos aprašymas	Įtaka
Patikimumas	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Patogumas (vartotojimo savybės)	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Priežiūros ir modifikavimo savybės	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Našumas	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Perkeliamumas	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Galutinis naudotojo efektyvumas	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Naudojimo paprastumas	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Pakeitimų lengvumas	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Saugumas	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Tranzakcijų greitis	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Integracija su kitomis sistemomis	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...
Daugiaplatformiškumas	0-nėra įtakos, 3-vidutinė įtaka, 5-stipri įtaka	...

Apskaičiavę visus faktorių sudėtingumus galime skaičiuoti šio metodo taškų skaičių (T). Formulė naudojama remiantis panaudos atvejų taškų metodo panaudojimo atvejų taškų skaičiavimu (žr 1.6.2 skyrius).

$$T = SC * KF * (PAS + FS) \quad (20)$$

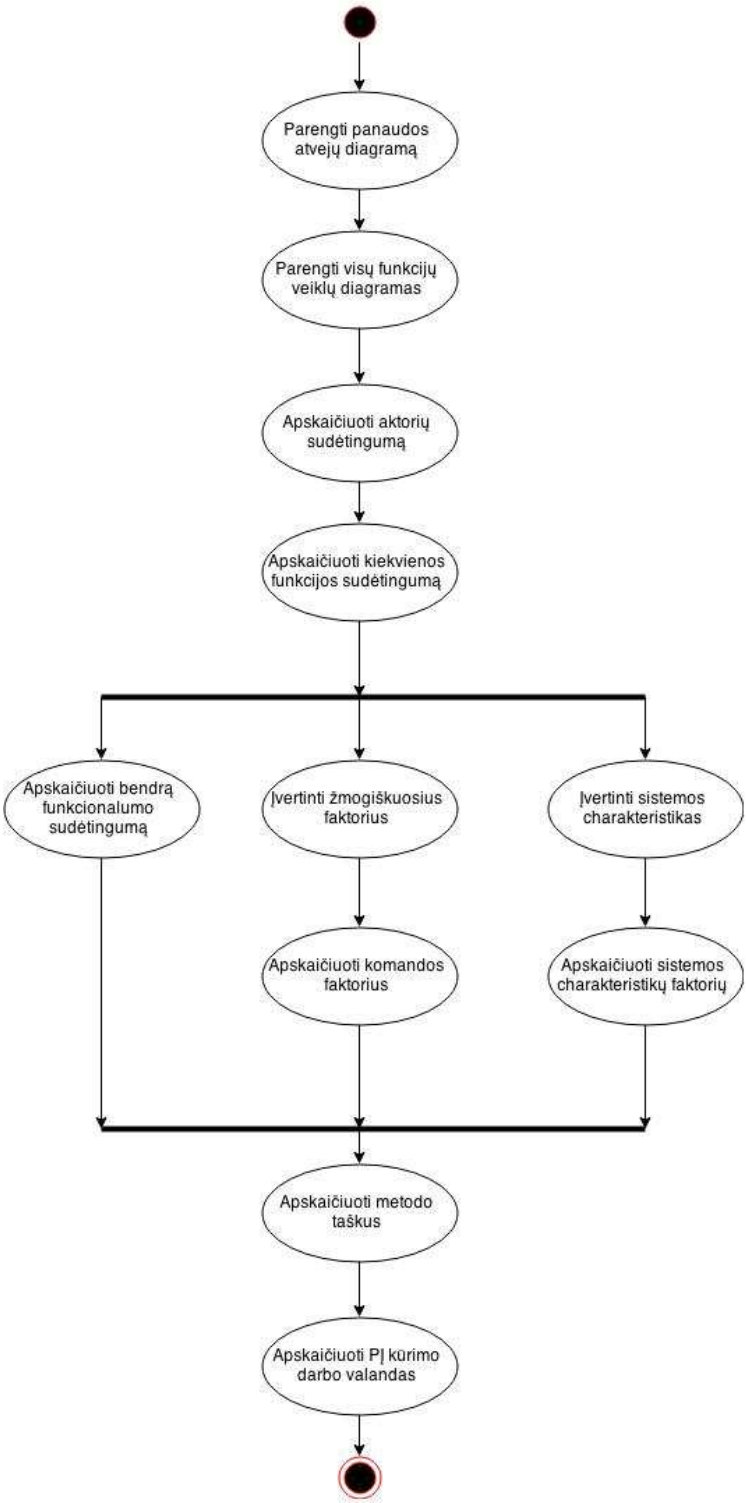
Gavę taškus galime skaičiuoti kūrimo laiką (L). Formulė naudojama remiantis panaudos atvejų taškų metodo žmogaus darbo valandų skaičiavimu (žr 1.6.2 skyrius).

$$L = T * 20 \quad (21)$$

Taip gauname valandų skaičių reikalingą sistemos kūrimui.

3. 3 Veiklos tyrimo diagrama

Veiklos tyrimui atvaizduoti pasinaudota veiklos diagrama. Veiklos diagrama puikiai parodo atliekamus veiksmus, jų eiliškumą, bei galimus pasirinkimus.



17 pav. *Veiklos tyrimo diagrama*

Veiklos tyrimo diagramoje (žr. 17 pav.) matome kaip atliekamas kainos skaičiavimas. Pirmiausia reikia parengti panaudos atvejų diagramą. Tada pagal visas sistemos funkcijas yra parengiamos funkcijų veiklos diagramos. Pagal panaudos atvejų diagramą įvertinami aktorių sudėtingumai. Iš veiklos diagramų įvertinami visi diagramos elementai ir jų sudėtingumas. Tada įvertinamas visos funkcijos sudėtingumas.

Sekantis skaičiavimo žingsnis yra apskaičiuoti bendrą funkcionalumo sudėtingumą. Taip pat reikia įvertinti žmogiškuosius faktorius ir sistemos charakteristikas. Apskaičiuojama komandos faktoriaus įtaka ir sistemos charakteristikų įtaka. Viską apskaičiavus galima skaičiuoti metodo taškus, o po to galima įvertinti sistemos kūrimo laiką darbo valandomis.

3. 4 Metodo praktinis patikrinimas

Pirmiausia įvertinamas panaudos atvejų aktorių sudėtingumas. Svečias – paprastas (svoris 1), klientas – sudėtingas (svoris 3), administratorius – labai sudėtingas (svoris 4). Skaičiuojame panaudos atvejų sudėtingumą (PAS) pagal 15 formulę ir gauname rezultatą 8.

$$PAS = 1+3+4=8$$

Skaičiuojama visų funkcijų veiklų diagramų sudėtingumas. Tam reikalingos funkcijų veiklos diagramos. Įvertinami veiklos diagramos elementų sudėtingumai, jie susumuojami. Pagal gautą rezultatą nustatome funkcijos sudėtingumą pagal vienos funkcijos sudėtingumo nustatymo lentelę (žr. 23 lentelė).

Visos funkcijų veiklos diagramos pateiktos prieduose (žr. 3 priedas).

Prekių peržiūros funkcijos veiklos diagramą (žr. 20 pav.) sudaro trys paprastos veiklos: prekių kategorijos pasirinkimas, norimos prekės pasirinkimas ir prekės informacijos peržiūra. Taip pat viena paprasta juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 4. Vadinasi šios funkcijos sudėtingumas paprastas (svoris - 1).

26 lentelė. Prekių peržiūros funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veiklos	Paprastas	3	1	3
Juosta	Paprastas	1	1	1
			Bendras funkcijos rezultatas:	4
Funkcijos sudėtingumas:	paprastas		Svoris:	1

Informacijos peržiūros funkcijos veiklos diagramą (žr. 21 pav.) sudaro dvi paprastos veiklos: norimos informacijos pasirinkimas ir informacijos peržiūra. Diagramoje yra viena paprasta juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 3. Vadinasi šios funkcijos sudėtingumas paprastas (svoris - 1).

27 lentelė. Informacijos peržiūros funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veiklos	Paprastas	2	1	2
Juosta	Paprastas	1	1	1
			Bendras funkcijos rezultatas	3
Funkcijos sudėtingumas:	Paprastas		Svoris:	1

Registracijos funkcijos veiklos diagramą (žr. 22 pav.) sudaro dvi paprastos, trys vidutinės ir dvi sudėtingos veiklos. Diagramoje juostų sudėtingumas vidutinis. Taip pat yra sudėtinga salyga bei vidutinis objektas. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 21. Vadinasi šios funkcijos sudėtingumas labai sudėtingas (svoris - 4).

28 lentelė. Registracijos funkcijos sudėtingumo nutatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	2	1	2
Veikla	Vidutinis	3	2	6
Veikla	Sudėtingas	2	3	6
Juosta	Vidutinis	1	2	2
Salyga	Sudėtingas	1	3	3
Objektas	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	21
Funkcijos sudėtingumas:	Labai sudėtingas		Svoris:	4

Prisijungimo funkcijos veiklos diagramą (žr. 23 pav.) sudaro trys paprastos ir trys vidutinės veiklos. Diagramoje yra vidutinio sudėtingumo juosta ir salyga. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 13. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

29 lentelė. Prisijngimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	3	1	3
Veikla	Vidutinis	3	2	6
Juosta	Vidutinis	1	2	2
Salyga	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	13
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Prekės užsakymo funkcijos veiklos diagramą (žr. 24 pav.) sudaro trys paprastos ir trys vidutinės veiklos. Taip pat diagramoje yra vidutinio sudėtingumo juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 11. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

30 lentelė. Prekės užsakymo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	3	1	3
Veikla	Vidutinis	3	2	6
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	11
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Krepšelio redagavimo funkcijos veiklos diagramą (žr. 25 pav.) sudaro dvi paprastos, viena vidutinė ir dvi sudėtingos veiklos. Diagramoje yra vidutinio sudėtingumo juosta ir lygiagretumas. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 14. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

31 lentelė. Krepšelio redagavimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	2	1	2
Veikla	Vidutinis	1	2	2
Veikla	Sudėtingas	2	3	6
Juosta	Vidutinis	1	2	2
Lygiagretumas	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	14
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Profilio keitimo funkcijos veiklos diagramą (žr. 26 pav.) sudaro dvi paprastos ir po vieną vidutinės ir sudėtingos veiklos. Taip pat diagramoje yra vidutinio sudėtingumo juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 9. Vadinasi šios funkcijos sudėtingumas paprastas (svoris - 1).

32 lentelė. Profilio keitimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	2	1	2
Veikla	Vidutinis	1	2	2
Veikla	Sudėtingas	1	3	3
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	9
Funkcijos sudėtingumas:	Paprastas		Svoris:	1

Sistemos nustatymų keitimo funkcijos veiklos diagramą (žr. 27 pav.) sudaro trys paprastos ir viena sudėtinga veiklos. Diagramoje yra juosta, kurios sudėtingumas vidutinis. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 8. Vadinasi šios funkcijos sudėtingumas paprastas (svoris - 1).

33 lentelė. Sistemos nustatymų keitimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	3	1	3
Veikla	Sudėtingas	1	3	3
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	8
Funkcijos sudėtingumas:	Paprastas		Svoris:	1

Prekių administravimo funkcijos veiklos diagramą (žr. 28 pav.) sudaro dvi paprastos ir trys sudėtingos veiklos. Diagramoje yra paprasto sudėtingumo juosta ir sudėtingo sudėtingumo lygiagretumas. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 15. Vadinasi šios funkcijos sudėtingumas sudėtingas (svoris - 3).

34 lentelė. Prekių administravimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	2	1	2
Veikla	Sudėtingas	3	3	9
Juosta	Paprastas	1	1	1
Lygiagretumas	Sudėtingas	1	3	3
			Bendras funkcijos rezultatas	15
Funkcijos sudėtingumas:	Sudėtingas		Svoris:	3

Naujos prekės kūrimo funkcijos veiklos diagramą (žr. 29 pav.) sudaro keturios paprastos, trys vidutinės ir viena sudėtinga veiklos. Diagramoje yra vidutinio sudėtingumo juosta ir objektas. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 17. Vadinasi šios funkcijos sudėtingumas sudėtingas (svoris - 3).

35 lentelė. Naujos prekės kūrimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	4	1	4
Veikla	Vidutinis	3	2	6
Veikla	Sudėtingas	1	3	3
Juosta	Vidutinis	1	2	2
Objektas	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	17
Funkcijos sudėtingumas:	Sudėtingas		Svoris:	3

Prekės šalinimo funkcijos veiklos diagramą (žr. 30 pav.) sudaro trys paprastos ir dvi sudėtingos veiklos. Diagramoje yra vidutinio sudėtingumo juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 11. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

36 lentelė. Prekės šalinimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	3	1	3
Veikla	Sudėtingas	2	3	6
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	11
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Prekės redagavimo funkcijos veiklos diagramą (žr. 31 pav.) sudaro keturios paprastos ir po vieną vidutinę ir sudėtingą veiklą. Diagramoje yra vidutinė juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 11. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

37 lentelė. Prekės redagavimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	4	1	4
Veikla	Vidutinis	1	2	2
Veikla	Sudėtingas	1	3	3
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	11
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Vartotojų administravimo funkcijos veiklos diagramą (žr. 32 pav.) sudaro dvi paprastos ir trys sudėtingos veiklos. Diagramoje yra paprasta juosta ir sudėtingas lygiagretumas. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 14. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

38 lentelė. Vartotojų administravimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	2	1	2
Veikla	Sudėtingas	3	3	9
Juosta	Paprastas	1	1	1
Lygiagretumas	Sudėtingas	1	3	3
			Bendras funkcijos rezultatas	14
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Naujo vartotojo kūrimo funkcijos veiklos diagramą (žr. 33 pav.) sudaro trys paprastos, viena vidutinė ir viena sudėtinga veiklos. Diagramoje yra po vieną vidutinio sudėtingumo juostą ir objektą.

Įvertinę diagramos elementų sudėtingumą gauname rezultatą 12. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

39 lentelė. Naujo vartotojo kūrimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	3	1	3
Veikla	Vidutinis	1	2	2
Veikla	Sudėtingas	1	3	3
Juosta	Vidutinis	1	2	2
Objektas	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	12
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Vartotojo šalinimo funkcijos veiklos diagramą (žr. 34 pav.) sudaro dvi paprastos, viena vidutinė ir viena sudėtinga veiklos. Diagramoje yra viena vidutinė juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 9. Vadinasi šios funkcijos sudėtingumas paprastas (svoris - 1).

40 lentelė. Vartotojo šalinimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	2	1	2
Veikla	Vidutinis	1	2	2
Veikla	Sudėtingas	1	3	3
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	9
Funkcijos sudėtingumas:	Paprastas		Svoris:	1

Vartotojo rolės keitimo funkcijos veiklos diagramą (žr. 35 pav.) sudaro keturios paprastos ir viena sudėtinga veiklos. Taip pat viena vidutinė juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 9. Vadinasi šios funkcijos sudėtingumas paprastas (svoris - 1).

41 lentelė. Vartotojo rolės keitimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	4	1	4
Veikla	Sudėtingas	1	3	3
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	9
Funkcijos sudėtingumas:	Paprastas		Svoris:	1

Informacijos administravimo funkcijos veiklos diagramą (žr. 36 pav.) sudaro dvi paprastos, trys sudėtingos veiklos. Taip pat viena paprasta juosta ir vienas sudėtingas lygiagretumas. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 16. Vadinasi šios funkcijos sudėtingumas sudėtingas (svoris - 3).

42 lentelė. Informacijos administravimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	2	1	2
Veikla	Sudėtingas	3	3	9
Juosta	Paprastas	1	2	2
Lygiagretumas	Sudėtingas	1	3	3
			Bendras funkcijos rezultatas	16
Funkcijos sudėtingumas:	Sudėtingas		Svoris:	3

Naujos informacijos kūrimo funkcijos veiklos diagramą (žr. 37 pav.) sudaro keturios paprastos, viena vidutinė ir viena sudėtinga veiklos. Diagramoje yra vidutinio sudėtingumo juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 11. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

43 lentelė. Naujos iformacijos kūrimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	4	1	4
Veikla	Vidutinis	1	2	2
Veikla	Sudėtingas	1	3	3
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	11
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Informacijos redagavimo funkcijos veiklos diagramą (žr. 38 pav.) sudaro keturios paprastos, viena vidutinė ir viena sudėtinga veiklos. Taip pat vidutinė juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 11. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

44 lentelė. Informacijos redagavimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	4	1	4
Veikla	Vidutinis	1	2	2
Veikla	Sudėtingas	1	3	3
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	11
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Informacijos šalinimo funkcijos veiklos diagramą (žr. 39 pav.) sudaro keturios paprastos ir dvi sudėtingos veiklos. Tai pat vidutinė juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 12. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

45 lentelė. Informacijos šalinimo funkcjos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	4	1	4
Veikla	Sudėtingas	2	3	6
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	12
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Užsakymų administravimo funkcijos veiklos diagramą (žr. 40 pav.) sudaro trys paprastos ir viena sudėtinga veiklos. Taip pat diagramoje yra paprasta juosta ir vidutinis lygiagretumas. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 9. Vadinasi šios funkcijos sudėtingumas paprastas (svoris - 1).

46 lentelė. Užsakymų administravimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	3	1	3
Veikla	Sudėtingas	1	3	3
Juosta	Paprastas	1	1	1
Lygiagretumas	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	9
Funkcijos sudėtingumas:	Paprastas		Svoris:	1

Užsakymo atšaukimo funkcijos veiklos diagramą (žr. 41 pav.) sudaro trys paprastos ir dvi sudėtingos veiklos. Taip pat diagramoje yra vidutinio sudėtingumo juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 11. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

47 lentelė. Užsakymo atšaukimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	3	1	3
Veikla	Sudėtingas	2	3	6
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	11
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Užsakymo peržiūros funkcijos veiklos diagramą (žr. 42 pav.) sudaro keturios paprastos veiklos ir paprasta juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 5. Vadinasi šios funkcijos sudėtingumas paprastas (svoris - 1).

48 lentelė. Užsakymo peržiūros funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	4	1	4
Juosta	Paprastas	1	1	1
			Bendras funkcijos rezultatas	5
Funkcijos sudėtingumas:	Paprastas		Svoris:	1

Reklamos administravimo funkcijos veiklos diagramą (žr. 43 pav.) sudaro dvi paprastos ir dvi sudėtingos veiklos. Diagramoje taip pat yra paprasta juosta ir vidutinis lygiagretumas. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 11. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

49 lentelė. Reklamos administravimo funkcijos sudėtigumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	2	1	2
Veikla	Sudėtingas	2	3	6
Juosta	Paprastas	1	1	1
Lygiagretumas	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	11
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Reklamos kūrimo funkcijos veiklos diagramą (žr. 44 pav.) sudaro keturios paprastos ir po vieną vidutinę ir sudėtingą veiklą. Taip pat yra vidutinė juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 11. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

50 lentelė. Reklamos kūrimo funkcijos sudėtingumo nustatymas

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	4	1	4
Veikla	Vidutinis	1	2	2
Veikla	Sudėtingas	1	3	3
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	11
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Reklamos šalinimo funkcijos veiklos diagramą (žr. 45 pav.) sudaro keturios paprastos ir dvi sudėtingos veiklos. Diagramoje taip pat yra vidutinė juosta. Įvertinę diagramos elementų sudėtingumą gauname rezultatą 12. Vadinasi šios funkcijos sudėtingumas vidutinis (svoris - 2).

Elementas	Sudėtingumas	Kiekis	Svoris	Rezultatas
Veikla	Paprastas	4	1	4
Veikla	Sudėtingas	2	3	6
Juosta	Vidutinis	1	2	2
			Bendras funkcijos rezultatas	12
Funkcijos sudėtingumas:	Vidutinis		Svoris:	2

Įvertinus visų funkcijų sudėtingumus juos sudedame pagal 17 formulę. Apskaičiavę gauname rezultatą lygų 50.

$$FS = 1+1+4+2+2+2+1+1+3+3+2+2+3+2+1+1+3+2+2+2+1+2+1+2+2+2=50$$

Toliau įvertiname komandos faktorių įtaką nuo 0 iki 5.

51 lentelė. Komandos faktorius

Faktorius	Įtaka
Taikymų patirtis	3
Stabilūs reikalavimai	3
Žinios apie verslo procesą	2
Aiškūs reikalavimai	2
Komandos bendradarbiavimas	1
Greita reakcija	1
Išankstinis numanymas	2
Problemos supratimas	3
Patirtis	3
Naujos žinios	2
Profesinės žinios	4
Kompetencija	3
Dizainerio poreikis	2
Projektuotojo poreikis	0
Konsultacijų poreikis	0
Apmokymų poreikis	0
Bendras rezultatas	31

Įvertinus visas komandos faktorių įtakas jas susumuojame. Tada skaičiuojame komandos faktoriaus sudėtingumą pagal 18 formulę ir gauname rezultatą 0,47.

$$KF = 31*(-0,03)+1,4=0,47$$

Analogiškai kaip komandos faktorius įvertiname ir sistemos charakteristikas. Jos vertinamos taip pat nuo 0 iki 5.

52 lentelė. Sistemos charakteristikos

Sistemos charakteristika	Įtaka
Patikimumas	3
Patogumas (vartotojimo savybės)	2
Priežiūros ir modifikavimo savybės	2
Našumas	4
Perkeliamumas	0
Galutinis naudotojo efektyvumas	3

Naudojimo paprastumas	2
Pakeitimų lengvumas	2
Saugumas	3
Tranzakcijų greitis	2
Integracija su kitomis sistemomis	4
Daugiaplatformiškumas	0
Bendras rezultatas	27

Įvertinus visas įtakas, jos susumuojamos. Ir tada pagal 19 formulę skaičiuojama sistemos charakteristikų sudėtingumas. Taip gaunamas rezultatas 0,87.

$$SC = 27*0,01+0,6=0,87$$

Atlikus panaudos atvejų aktorių sudėtingumo, funkcijų sudėtingumų įvertinimus, taip pat komandos faktorių bei sistemos charakteristikų įtakas galima apskaičiuoti metodo taškus. Jie skaičiuojami pagal 20 formulę. Viską apskaičiavus gaunamas rezultatas 23,72.

$$T = 0,87*0,47*(8+50)=23,72$$

Gavus metodo taškų skaičių galima skaičiuoti sistemos kūrimo laiką darbo valandomis. Tai atliekama pasinaudojus 21 formule. Gaunamas rezultatas lygus 474,4 žmogaus darbo valandoms.

$$L = 42,891*20=474,4$$

3. 5 Metodų apibendrinimas

Visi analizuojami metodai apskaičiuoja laiką darbo valandomis, reikalingą programinės įrangos kūrimui. Įvertinus laiką galima jį padauginti iš atlikto darbo įkainio. Taip gaunama programinės įrangos kaina. Tai patogu, nes kūrimo įkainiai gali skirtis pagal šalį, miestą ar įmonę. Taip pat įkainiai gali priklausyti nuo kūrėjų kvalifikacijos, kompetencijos, patirties ir kitų gebėjimų.

Šiuo atveju kaina paskaičiuota 15 eurų valandiniu įkainiu.

53 lentelė. Kainų palyginimas

Metodas	Panaudos atvejų taškų metodas	Funkcinių taškų metodas	Objektinių taškų metodas	PĮ funkcionalumo įvertinimo metodas	Reali kaina
Kūrimo laikas	1066 valandų	720 valandų	627,2 valandų	474,4 valandų	352 valandų
PĮ kaina	15990 eurų	10800 eurų	9408 eurų	7116 eurų	600 eurų

Apskaičiavus sistemos kūrimo laiką visais metodais galima teigti, kad visais atvejais laikas gavosi panašus. Naujai pasiūlytas programinės įrangos funkcionalumo įvertinimo metodas savo rezultatu yra arčiausiai realaus sistemos kūrimo laiko. Realaus sistemos kūrimo laiką įvertinus tokiu pat įkainiu, gautusi, kad jam artimiausias pasiūlytas metodas. Todėl galima teigti, kad pasiūlytas metodas sistemą įvertina gana tiksliai.

IŠVADOS

1. Analitinėje dalyje buvo pateikta informacija apie projektus, jų tipus bei kvalifikaciją. Taip pat buvo aprašytas programinės įrangos kūrimas ir projekto dalyviai. Analitinėje dalyje susipažįstama su darbo dalykine sritimi.
2. Darbe buvo analizuojami pagrindiniai programinės įrangos apimties dydžio vertinimo metodai. Ištirti ir išanalizuoti buvo panaudojimo atvejų taškų, funkcinių taškų ir objektų taškų metodai. Darbe buvo analizuota metodų skaičiavimo eiga ir nustatyta, kurie kriterijai labiausiai įtakoja programinės įrangos kainą. Tam, kad tinkamai būtų įvertinta programinė įranga, reikia įvertinti sistemos techninius faktorius ir aplinkos žmogiškuosius faktorius. Taip pat kaina priklauso nuo sistemos galimų funkcijų, todėl jas taip pat reikia įvertinti.
3. Darbe buvo atliktas tyrimas pasirinkus realią 2015 metais realizuotą elektroninę parduotuvę ir įvertinta jos kaina darbe analizuotais metodais. Išanalizavus tyrimo rezultatus buvo nustatyta, kad metodai nevertina sistemos funkcijų sudėtingumo atsižvelgiant tiek iš vartotojo, tiek iš sistemos įgyvendinimo pozicijų. Taip pat darbe nagrinėti metodai neįvertina šiuolaikinių kūrimo technologijų, sąlygojančių sistemos projektavimo ir kūrimo metodus. Teorinė ir praktinė metodų analizė parodė, kad reikalaingas programinės įrangos įvertinimo metodas įvertinantis sietmos funklciolumo sudėtingumą ir atsižvlegiantis iš šiuolaikinės sistemos kūrimo procesą.
4. Pasiūlytame metode esminis naujumas yra funkcijų sudėtingumo įvertinimas, skaičiuojant programinės įrangos kainą. Taip pat metode vertinama žmogiškųjų faktorių įtaka, bei sistemos charakteristikų įtaka. Tai padeda tiksliau įvertinti sistemą iš jos vidinės pusės ir iš vartotojo sąsajos.
5. Atlikus praktinį naujo pasiūlyto metodo tyrimą, buvo apskaičiuota sistemos kaina. Palyginus visų metodų kainą galima teigti, kad didžiausia kaina paskaičiuota panaudos atvejų taškų metodu, o mažiausia - naujai pasiūlytu metodu. Pasiūlytame metode buvo apjungti analizuojamų dydžio apimties vertinamų metodų kriterijai. Todėl pasiūlytas metodas apima daugiau vertinamų kriterijų ir tiksliau įvertina programinės įrangos kainą.

LITERATŪRA

1. Adomaitis E. *Projektų valdymas*. [žiūrėta 2014-11-20] Prieiga internete: https://www.google.lt/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCIQFjAA&url=http%3A%2F%2Fwww.mif.vu.lt%2F~ragaisis%2FPSI_mag2008%2FStudMedziaga%2FProjektu_valdymas-7gr_ErnestasAdomaitis.doc&ei=w5k7Ve7CIaahyAOA_ICYAg&usg=AFQjCNGMoECOxSZkJv3JVPRRpFDQU_HBtw&sig2=kPpyiXKkUQaRAU6-sx8unQ
2. Adamonytė I., Gudas M., Vaičiukynas V. *Projektų valdymas ir vandens politika.* 2008.
3. Afanasjeva V. *Programinės įrangos kūrimo projektų kokybės valdymas.* 2007. [žiūrėta 2014-11-13] Prieiga internete: http://vddb.library.lt/fedora/get/LT-eLABa-0001:E.02~2007~D_20090908_193946-96583/DS.005.1.01.ETD
4. Baužaitė D. *Programinės įrangos ir informacinių sistemų priežiūra didelėje organizacijoje*. 2007. [žiūrėta 2014-11-15] Prieiga internete: http://vddb.library.lt/fedora/get/LT-eLABa-0001:E.02~2007~D_20070816_144736-36406/DS.005.0.02.ETD
5. Dapkūnas S. *Programų sistemų kokybė*. 2007. [žiūrėta 2014-11-25] Prieiga internete: http://www.mif.vu.lt/~sigitas/Kokybe/PS_Kokybe.pdf
6. Dedonis V., Dovidonienė L. *Programinės įrangos kūrimo kompleksiškumo modelis: eksperimentinio tyrimo rezultatai*, 2013m. [žiūrėta 2013-12-10]. Prieiga internete: http://www.zara.lt/e-knygos/e-KoDi-2013-MD.pdf
7. Ebert C., Dumke R., Bundschuh M., Schmietindorf A. *Best Practices in Software Measurement: How to use metrics to improve project and process performance*. Springer-Verlag Berlin Heidelberg, 2005.
8. Gervė Š. *Funkcinių taškų analizės metodų tyrimas*. 2010. [žiūrėta 2014-10-20] Prieiga internete: http://vddb.library.lt/fedora/get/LT-eLABa-0001:E.02~2010~D_20110709_152454-40358/DS.005.1.01.ETD
9. HeydarNoori A. *Function point analysis*. [žiūrėta 2014-10-20] Prieiga internete: https://cs.uwaterloo.ca/~apidduck/CS846/Seminars/abbas.pdf
10. International Function Point Users Group. *Function Point Counting Practices Manual*. 2000. [žiūrėta 2013-12-09]. Prieiga internete: http://perun.pmf.uns.ac.rs/old/repository/research/se/functionpoints.pdf
11. Jociūtė D. *Užsiėmimų tvarkaraščių projektavimo sistema*. 2005 [žiūrėta 2014-12-22] Prieiga internete: http://www.elibrary.lt/resursai/Mokslai/KTU/Magistrai/D.Jociute_magistro%20darbas_2005.pdf
12. Kasparaitis P. *Žmogaus-kompiuterio sąveika. Programinės įrangos gyvavimo ciklas*. 2011. [žiūrėta 2014-09-20] Prieiga internete: http://www.mif.vu.lt/~pijus/ZKS/Gyv_ciklas.pdf
13. Laird L.M., Brennan M.C. *Software Measurement and Estimation: A Practical Approach*. A John Wiley & Sons, Inc., 2006.

14. Lapienė A. *IT projekto apimties įvetinimas*. [žiūrėta 2014-09-20] Prieiga internete: http://www.nomagic.lt/straipsniai/it-projekto-apimties-ivertinimas.html
15. Lašinskas K. *Geros ir blogos projektų valdymo praktikos*, 2008, [žiūrėta 2014-10-21]. Prieiga internete: http://www.slideserve.com/felton/geros-ir-blogos-projekt-valdymo-praktikos
16. Leščinskaitė D. *Informacinių sistemų projektų darbų apimties nustatymo metodika*. 2010. [žiūrėta 2013-11-20] Prieiga intrnete: http://vddb.library.lt/fedora/get/LT-eLABa-0001:E.02~2010~D_20100826_144308-11418/DS.005.0.01.ETD
17. Liming W. *The Comparison of the Software Cost Estimating Methods*. [žiūrėta 2014-01-02] Prieiga per internetą : http://www.compapp.dcu.ie/~renaat/ca421/LWu1.html
18. Longstreet D. *Fundamentals of Function Point Analysis*. Longstreet Consulting Inc., 2005.
19. Mikulėnas G. *Informacinės sistemos programinės įrangos kūrimo darbų apimties įvertinimo modelis*. 2006. [žiūrėta 2014-10-17] Prieiga internete: http://vddb.library.lt/fedora/get/LT-eLABa-0001:E.02~2006~D_20060111_134117-29486/DS.005.0.01.ETD
20. Norbert Heidbuchel„s Software. NH„s Function Point Analyzer. [žiūrėta 2013-12-22]. Prieiga internete: http://www.nh-software.com/#go3
21. Suresh Joseph. K, Ravichandran. T. ARPN Journal of Science and Technology. *A Novel Imputation Technique for Software Effort Estimation.* 2012 [žiūrėta 2013-12-22] Prieiga internete: http://www.ejournalofscience.org/archive/vol2no7/vol2no7_10.pdf
22. Šafranauskas P. *Veiklos modelio ir vartotojo reikalvimų (Use-Case) modelio sąsajos tyrimas*. 2007. [žiūrėta 2014-11-23] Prieiga internete: http://vddb.library.lt/fedora/get/LT-eLABa-0001:E.02~2007~D_20070816_143800-45511/DS.005.0.02.ETD
23. Šilingas D. *Programinės įrangos kūrimas*. [žiūrėta 2014-10-12] Prieiga internete: http://www.nomagic.lt/straipsniai/programines-%C4%AFrangos-kurimas.html
24. Ungurys A. *Reikalavimų formulavimas*. [žiūrėta 2014-12-05] Priega internete: https://www.google.lt/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCAQFjAA&url=http%3A%2F%2Fwww.mif.vu.lt%2F~ragaisis%2FTSPi%2F6-Reikalavimu_formulavimas.doc&ei=m5g7VZrrE-PZyAO19ICQDQ&usg=AFQjCNGvj4KANT_C1CT9nfdhzNMkYI7tzg&sig2=Gpz8gQT3j-WUz3uyUHs6tQ
25. Uzdanavičiūtė V. *Programų kaštų vertinimas*. 2010. [žiūrėta 2014-11-23] Prieiga internete: http://www.elen.ktu.lt/studentai/lib/exe/fetch.php?media=programu_kastu_vertinimas.pdf
26. Vaira Ž. *Programinės įrangos kūrimo technologijos*. 2013. [žiūrėta 2014-10-12] Prieiga internete: http://www.esparama.lt/es_parama_pletra/failai/ESFproduktai/2013_Programines_irangos_kurimo_technologijos.pdf.pdf
27. Vickers P. An Introduction to Function Point Analysis. Northumbria University,2003.

28. [žiūrėta 2014-12-14] Prieiga internete: http://www2.dir.state.tx.us/SiteCollectionDocuments/IT%20Leadership/Framework/ProjClassMethod.pdf
29. [žiūrėta 2014-12-14] Prieiga internete: http://rfka.tripod.com/Image76.gif
30. Chon M. *The purpose of Agile Planning*. 2009 [žiūrėta 2014-12-14] Prieiga internete: http://www.informit.com/articles/article.aspx?p=1374899
31. Petrylaitė G. *Programinės įrangos įvadas*. 2008 [žiūrėta 2015-03-17] Prieiga internete: http://www.softconsulting.lt/Straipsniai/Programines_irangos_ivadas
32. [žiūrėta 2014-12-14] Prieiga internete: http://geekswithblogs.net/Prabhats/archive/2007/03/01/107632.aspx

LENTELIŲ SĄRAŠAS

PAVEIKSLĖLIŲ SĄRAŠAS

TERMINŲ IR SANTRUMPŲ ŽODYNĖLIS

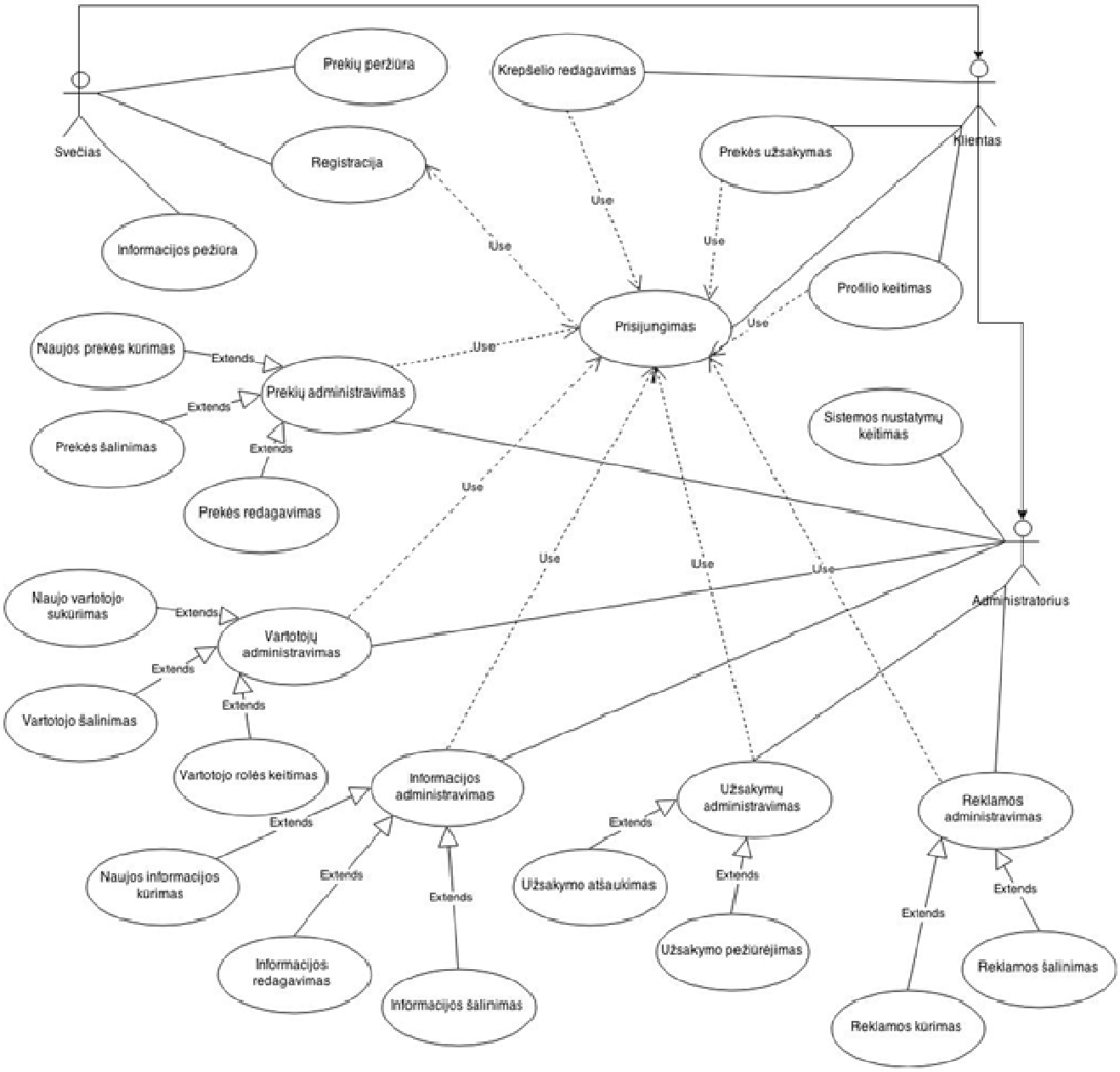
algoritminis modelis	modelis grindžiamas istoriniais programinės įrangos kainos skaičiavimų rezultatais
alternatyva	būtinybė ar galimybė rinktis vieną iš dviejų
BĮL	bendras įtakos laipsnis
charakteristika	apibūdinimas, skiriamųjų ypatybių nusakymas
DB	duomenų bazė
D.U.K	Dažniausiai užduodami klausimai
EAF	pastangos koregavimo koeficientas
ECF	aplinkos sudėtingumo faktorius
Effort	pastangos
ETF	bendras aplinkos faktorius
eksploatacija	naudojimas
fundamentalus	objektyvus veiksnys, darantis įtaką nagrinėjamam reiškiniui
funkcionalumas	gerai atliekantis taikomąją funkciją, atitinkantis poreikius; naudingas, pravartus
kaštai	sąnaudos tikslui pasiekti
KF	komandos faktoriaus sudėtingumas
KFĮ	komandos faktoriaus įtaka
KFTa	koreguoti funkciniai taškai
konstanta	fiksuotas, nekintantis dydis
komponentas	sudedamoji dalis
L	laikas
LF	lygio faktorius
metodas	žingsnių seka, padedanti pasiekti tikslą
NFT	nekoreguoti funkciniai taškai
OP	objektų taškai
PAS	panaudojimo atvejų sudėtingumas
PASS	panaudojimo atvejų sudėtingumo svoris
PF	darbo valandų skaičius reikalingas vienam panaudos atvejui

PI	programinė įranga
PMS	programinio modulio sudėtingumas
PROD	produktyvumas
produktyvumas	našumas
RKF	reikšmių koregavimo faktorius
RL	realizavimo laikas
S	duomenų arba transakcinės funkcijos sudėtingumas
sąnaudos	kuriam nors tikslui naudojamos lėšos, medžiagos, energija, darbas ir pan.
SC	sistemos charakteristikų sudėtingumas
SCĮ	sistemos charakteristikų įtaka
SK	sudėtingumo klasifikatorius
SLOC	kodo eilučių skaičius
T	žmogaus darbo valandų skaičius
TCF	techninio sudėtingumo faktorius
TDEV	kūrimo laikas mėnesiais
TTF	bendras techninis faktorius
UAW	aktorių svoris
UC	panaudos atvejis
UCP	panaudojimo atveju taškai
UUCP	nepakoreguoti panaudojimo atveju taškai
UUCW	panaudojimo atvejų svoris
VDS	veiklos diagramos sudėtingumas
VDSS	veiklos diagramos sudėtingumo svoris
FS	funkcionalumo sudėtingumas
VFS	vienos funkcijos sudėtingumas
ŽM	sąnaudos

PRIEDAI

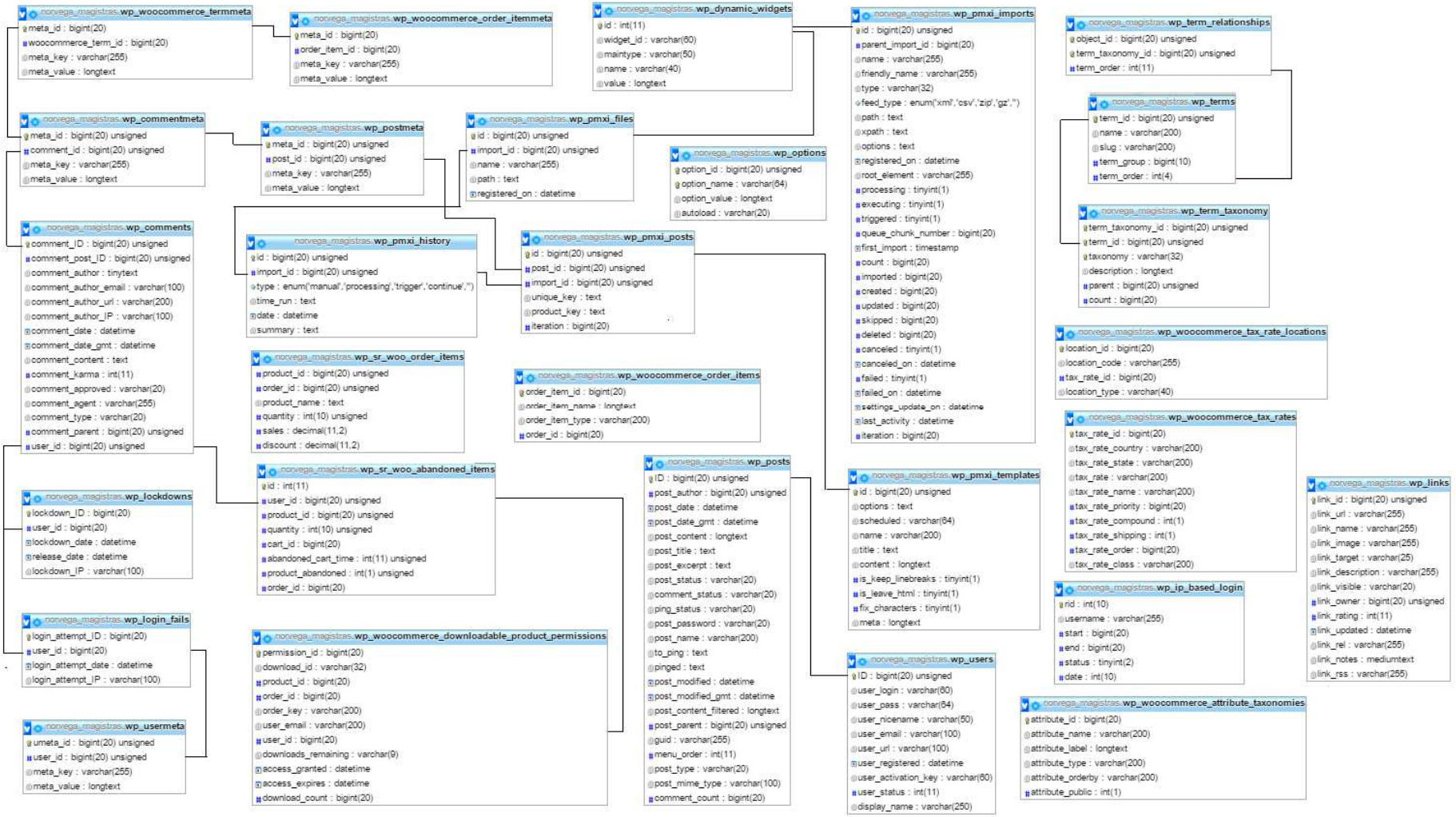
1 priedas. Sistemos struktūra

1.1. Sistemos panaudos atvejų diagrama



18 pav. *Sistemos panaudos atvejų diagrama*

1.2. Informacinės sistemos vidinė struktūra



19 pav. *Sistemos vidinė struktūra*

2 priedas. Sistemos kainos skaičiavimas

2.1. Kainos skaičiavimas panaudos atvejų metodu

54 lentelė. Panaudojimo atvejų svorių įvertinimas

Tipas	Aprašymas	Svoris	UC skaičius	Rezultatas
Paprastas	=< 3 opercijų	5	20	100
Vidutinis	4-7 operacijų	10	1	10
Sudėtingas	>7 operacijų	15	0	0
			UUCW = svoris * skaičius:	110

55 lentelė. Aktorių svorių įvertinimas

Tipas	Aprašymas	Svoris	Aktorių skaičius	Rezultatas
Paprastas	Sisteminės sąsajos	1	0	0
Vidutinis	Protokolo valdomos sąsajos	2	0	0
Sudėtingas	Vartotojo sąsajos	3	3	9
			UAW = svoris * skaičius:	9

56 lentelė. Bendrojo techninio faktoriaus įvertinimas

TF	Aprašas	Svoris	Įtaka	Rezultatas
T1	Sistemos pasiskirstymas	2	1	2
T2	Eksploatacinės savybės	1	2	2
T3	Galutinio vartotojo efektyvumas	1	3	3
T4	Vidinio veikimo kompleksiškumas	1	2	2
T5	Perpanaudojamumas	1	0	0
T6	Lengvas instaliavimas	0,5	0	0
T7	Lengvas naudojimas	0,5	2	1
T8	Portatyvumas	2	1	2
T9	Keitimo lengvumas	1	1	1
T10	Konkurencingumas	1	0	0
T11	Saugumas	1	3	3
T12	Tiesioginis perėjimas trečiosioms šalims	1	0	0
T13	Vartotojų apmokymo reikalingumas	1	0	0
T-lygis 0-5 kur 0 = nesvarbus 5 = esminis				
			TTF = svoris * įtaka	16

57 lentelė. Bendrojo aplinkos faktoriaus įvertinimas

EF	Aprašas	Svoris	Poveikis	Rezultatas
E1	Pažintis su UML	1,5	3	4,5
E2	Taikymų patirtis	0,5	4	2
E3	Objektais orientuota patirtis	1	3	3
E4	Pagrindinio analitiko gebėjimai	0,5	4	2
E5	Motyvacija	1	5	5
E6	Stabilūs reikalavimai	2	5	10
E7	Darbuotojai antraeilėse pareigose	-1	0	0
E8	Sudėtinga programavimo kalba	-1	0	0
E9	Žinios apie verslo procesą	0,5	5	2,5
T-lygis 0-5 kur 0 = nėra žinių 5 = ekspertas				
			ETF = svoris * poveikis:	27

2.2. Kainos skaičiavimas funkcinių taškų metodu

58 lentelė. Funkcinių taškų skaičiavimas

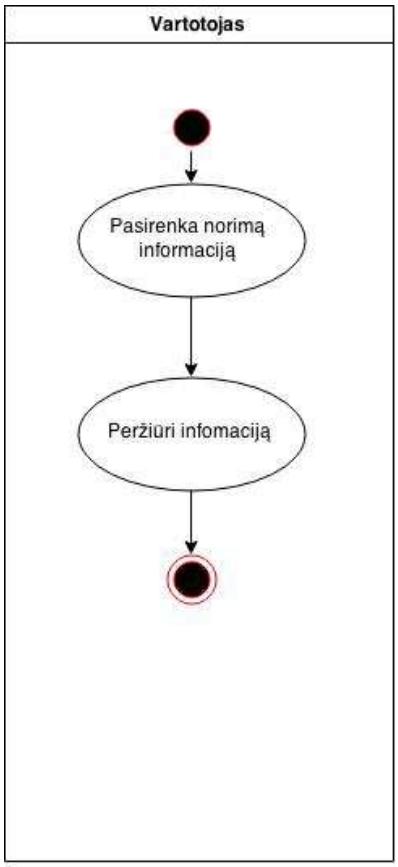
Tipas	Sudėtingumas			
	Žemas			Viso
	Kiek	Koef.	Viso	
IĮ	13	3	13×3	39
II	8	4	8×4	32
IU	34	3	34×3	102
VLF	29	7	29×7	203
IIF	4	5	4×5	20
Nekoreguoti funkciniai taškai				396

59 lentelė. Bendrųjų sistemos charakteistikų sudėtingumo įvertinimas

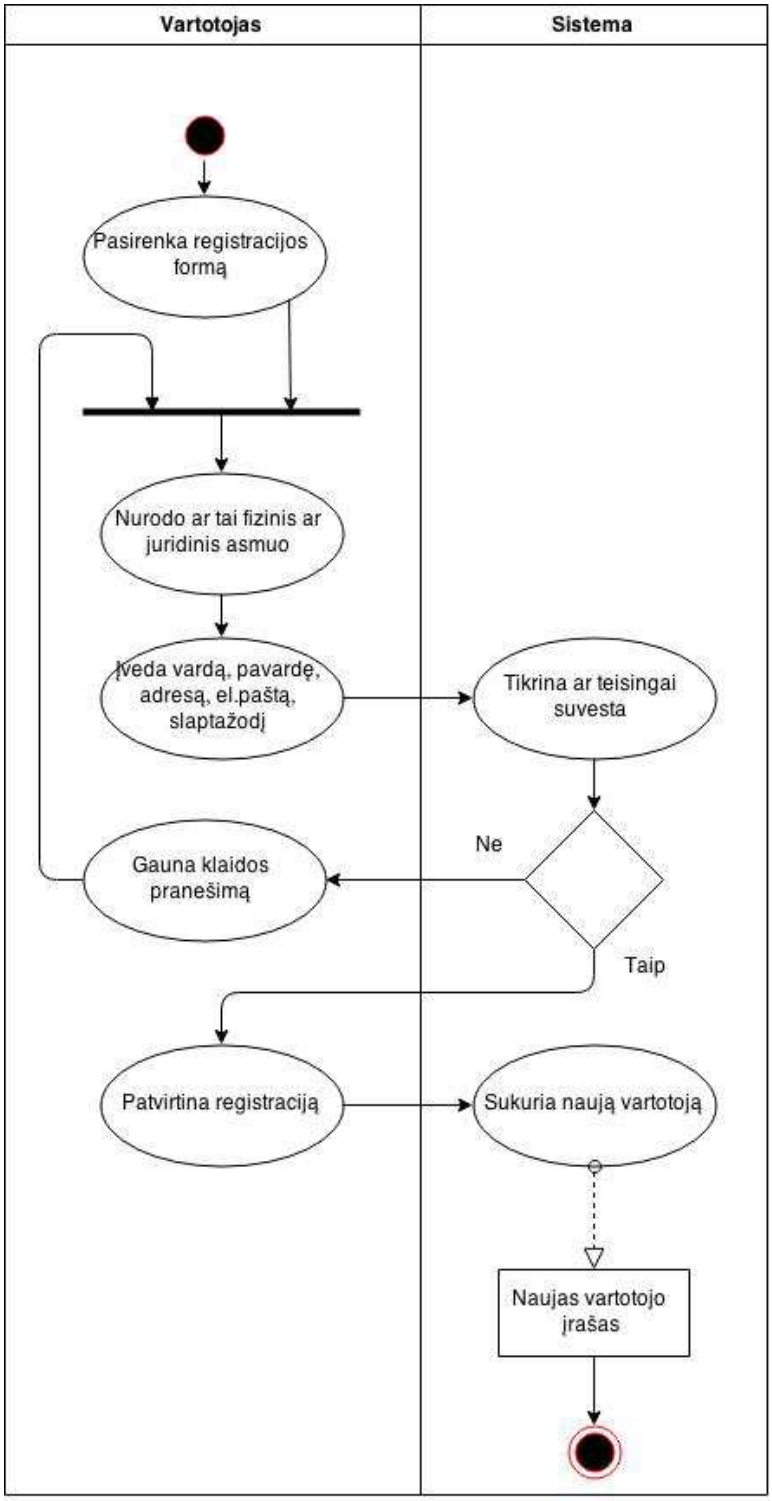
Bendrosios sistemos charakteristikos	Sudėtingumo laispnis
1. Duomenų perdavimas	3
2. Paskirstytų duomenų apdorojimas	5
3. Našumas	4
4. Dažnai naudojama konfigūracija	2
5. Tranzakcijų greitis	5
6. Tiesioginis duomenų įvedimas	4
7. Galutinio naudotojo efektyvumas	4
8. Tiesioginiai pakeitimai	3
9. Sudėtingas apdorojimas	2
10. Pakartotinis panaudojamumas	1
11. Diegimo paprastumas	0
12. Naudojimo paprastumas	3
13. Daugkartinis panaudojimas	1
14. Pakeitimų lengvumas	3

3 priedas. Funkcijų veiklos diagramos

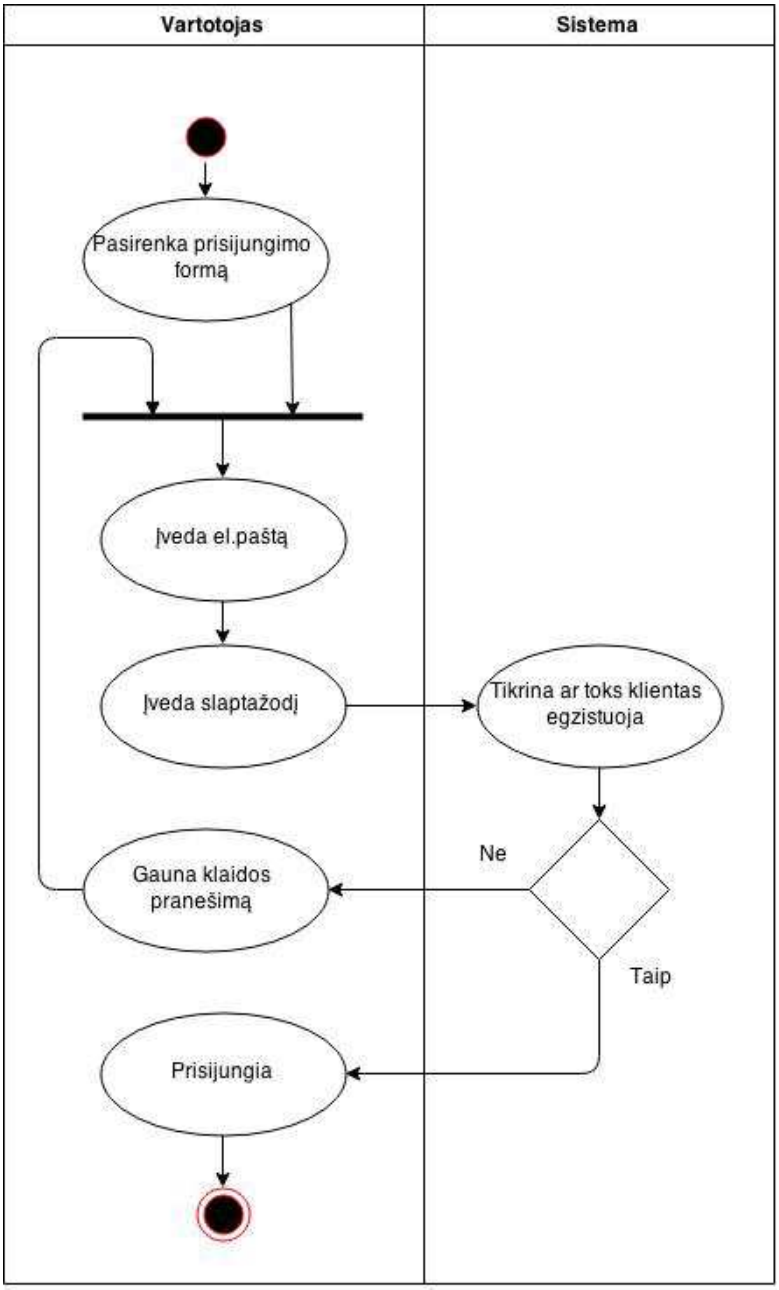
20 pav. *Prekių peržiūros funkcijos veiklos diagrama*



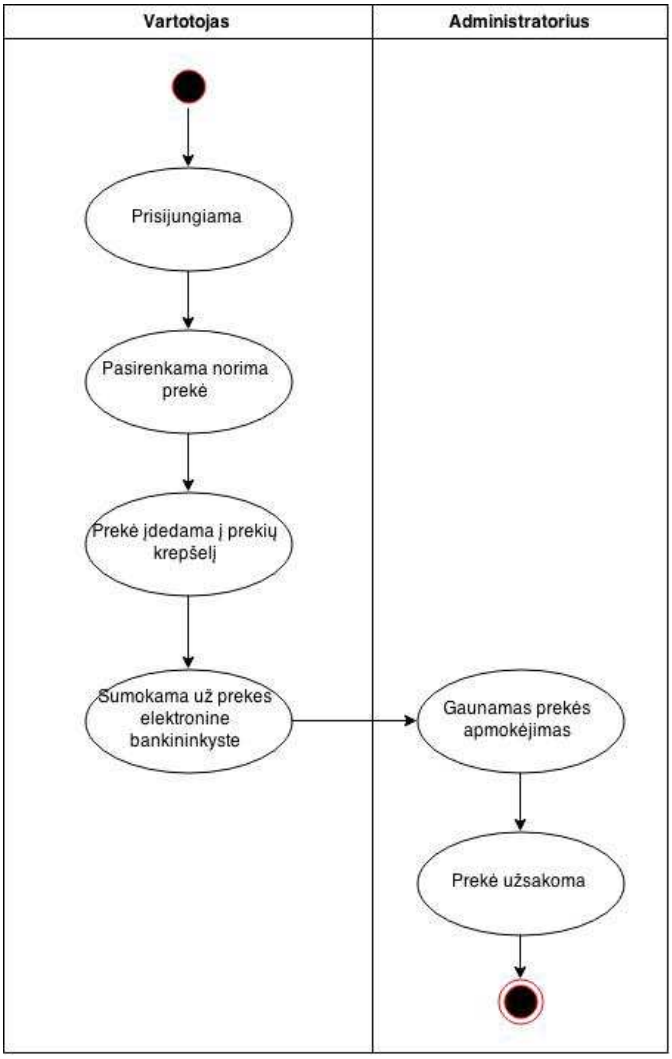
21 pav. *Informacijos peržiūros funkcijos veiklos diagrama*



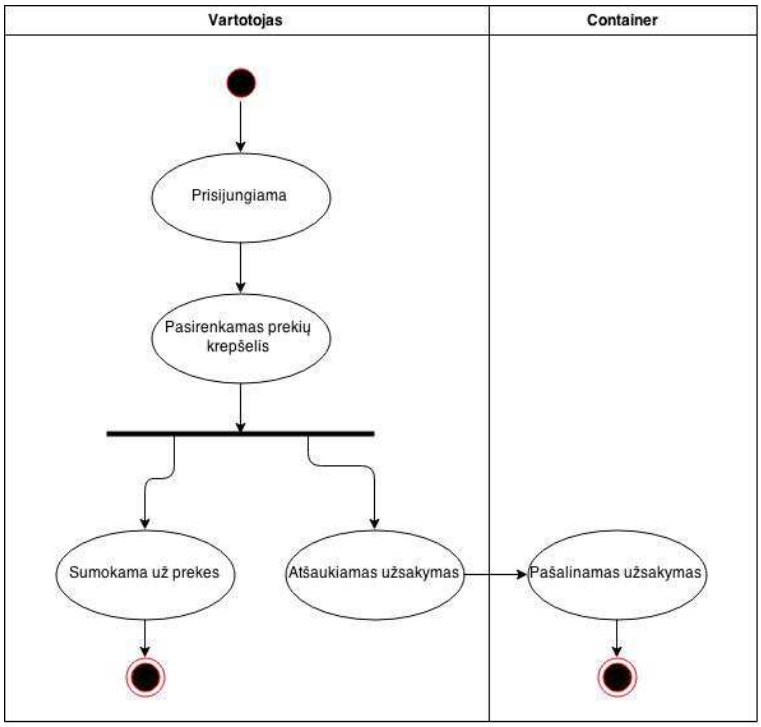
22 pav. *Registracijos funkcijos veiklos diagrama*



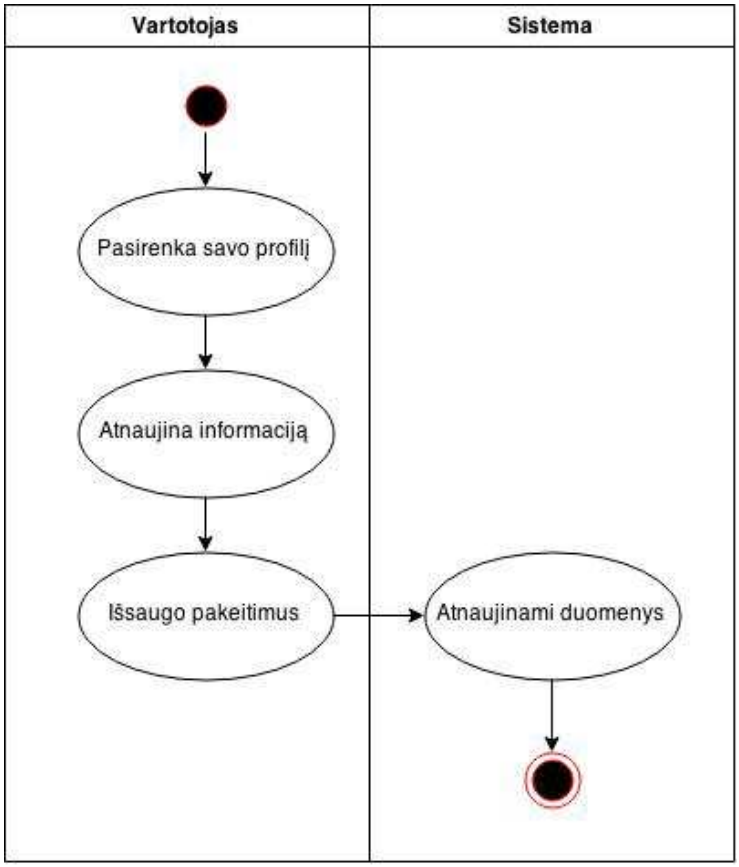
23 pav. *Prsijungimo funkcijos veiklos diagrama*



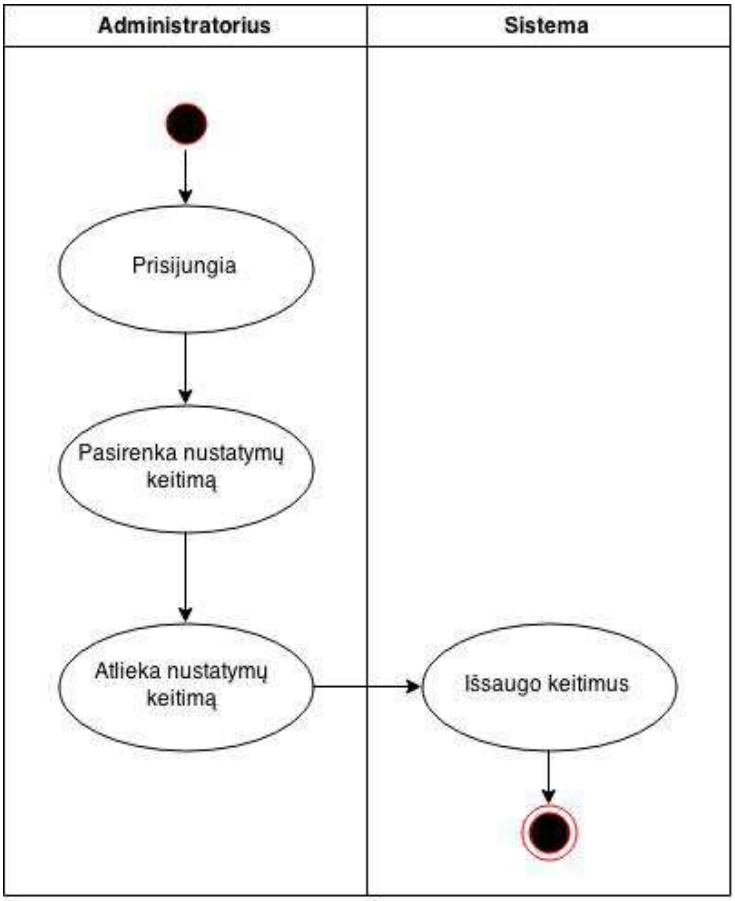
24 pav. *Prekės užsakymo funkcijos veiklos diagrama*



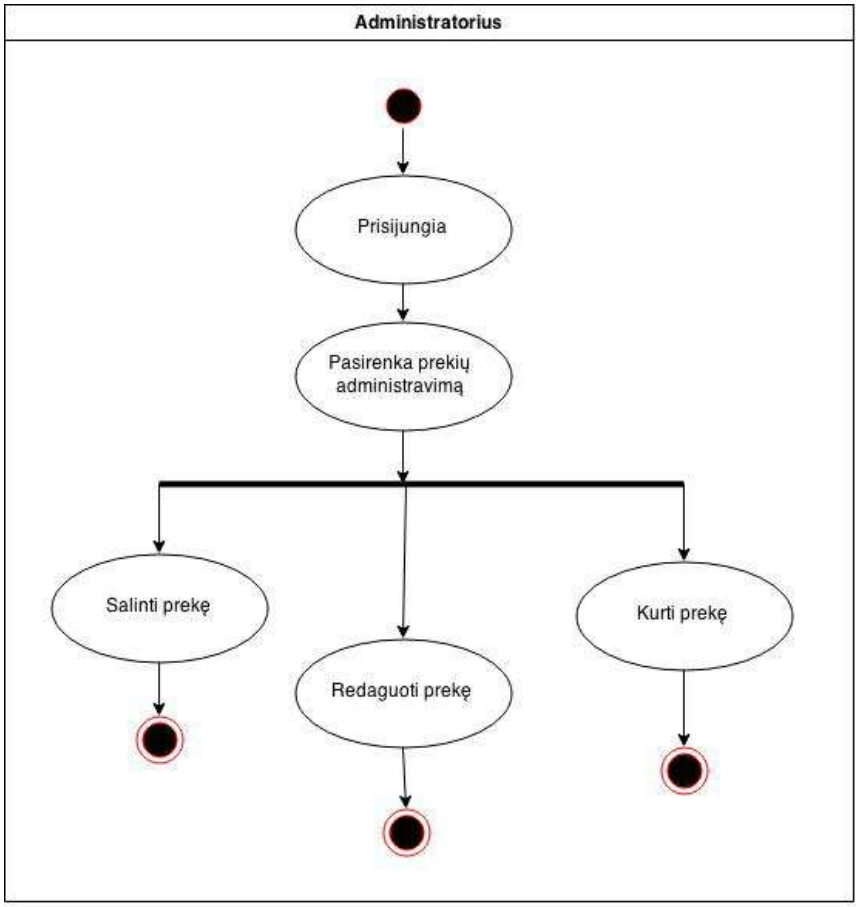
25 pav. *Krepšelio redagavimo funkcijos veikos diagrama*



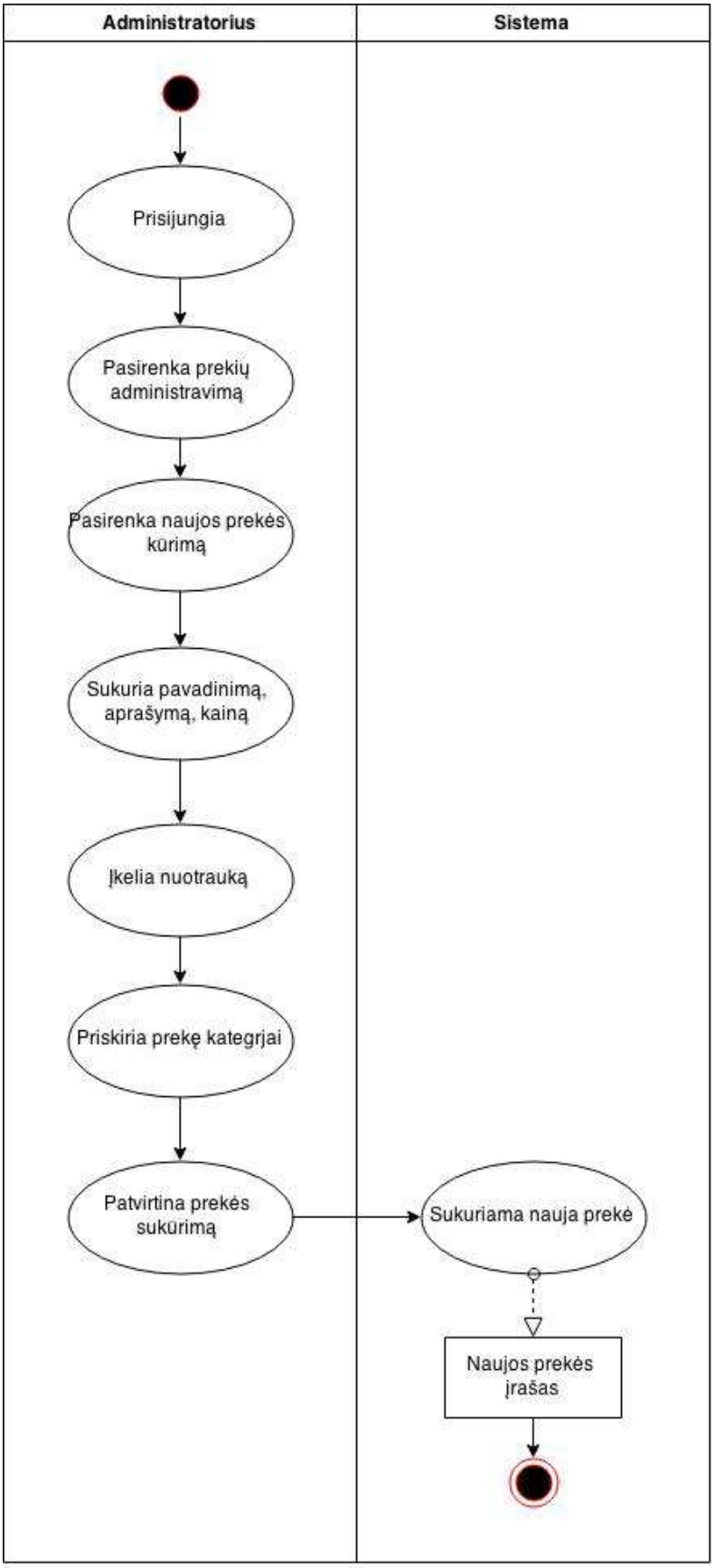
26 pav. *Profilio keitimo funkcijos veiklos diagrama*



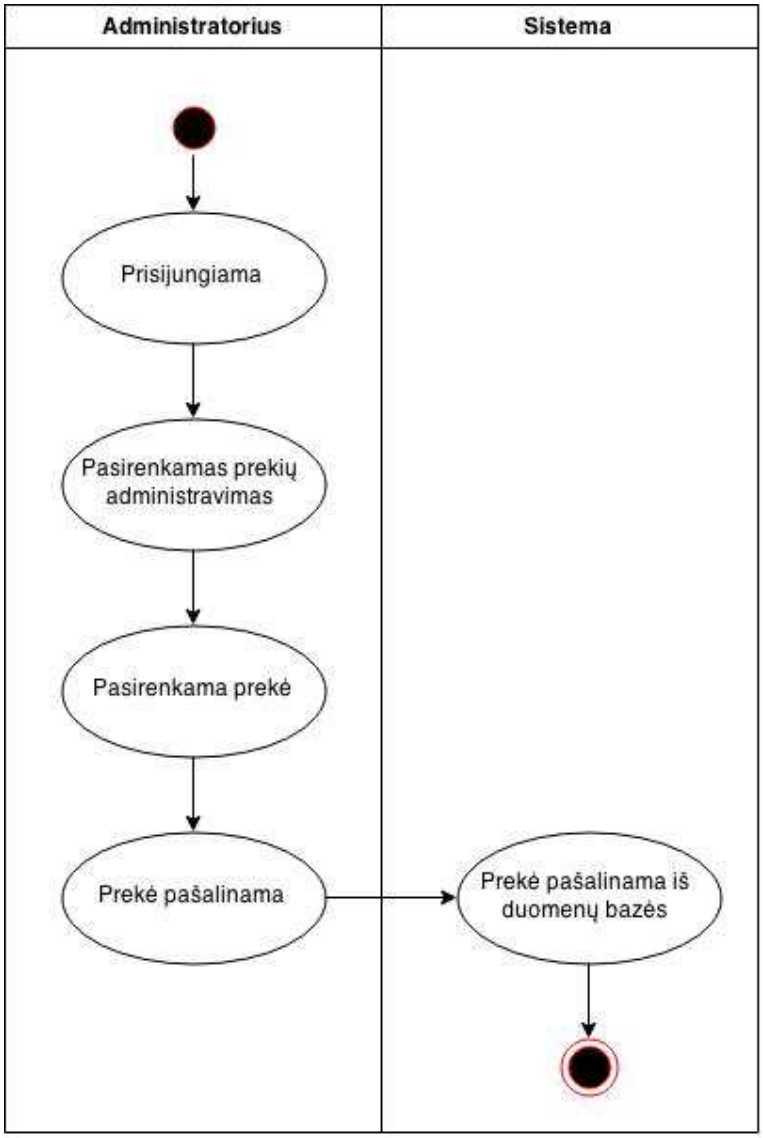
27 pav. *Sistemos nustatymų keitimo funkcijos veiklos diagrama*



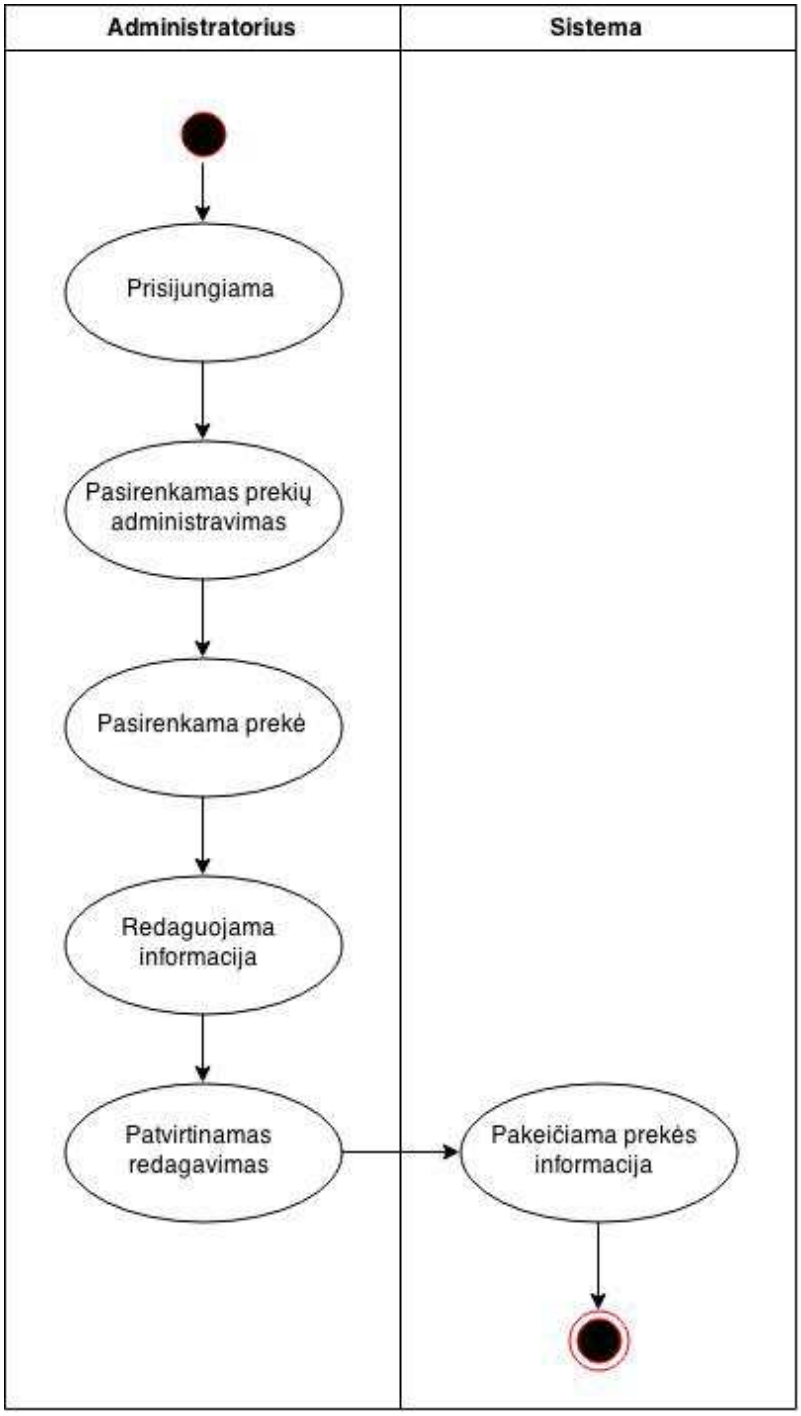
28 pav. *Prekių administravimo funkcijos veiklos diagrama*



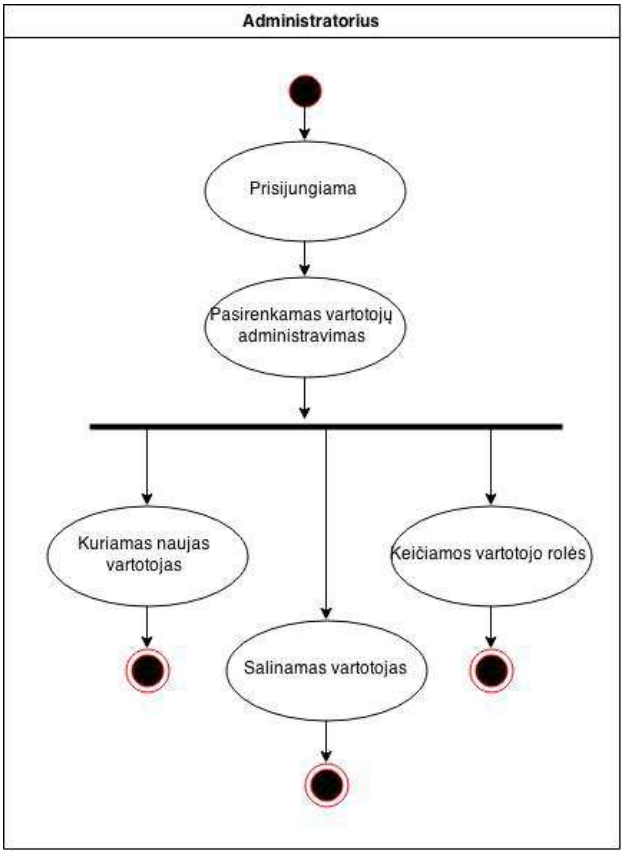
29 pav. *Naujos prekės kūrimo funkcijos veiklos diagrama*



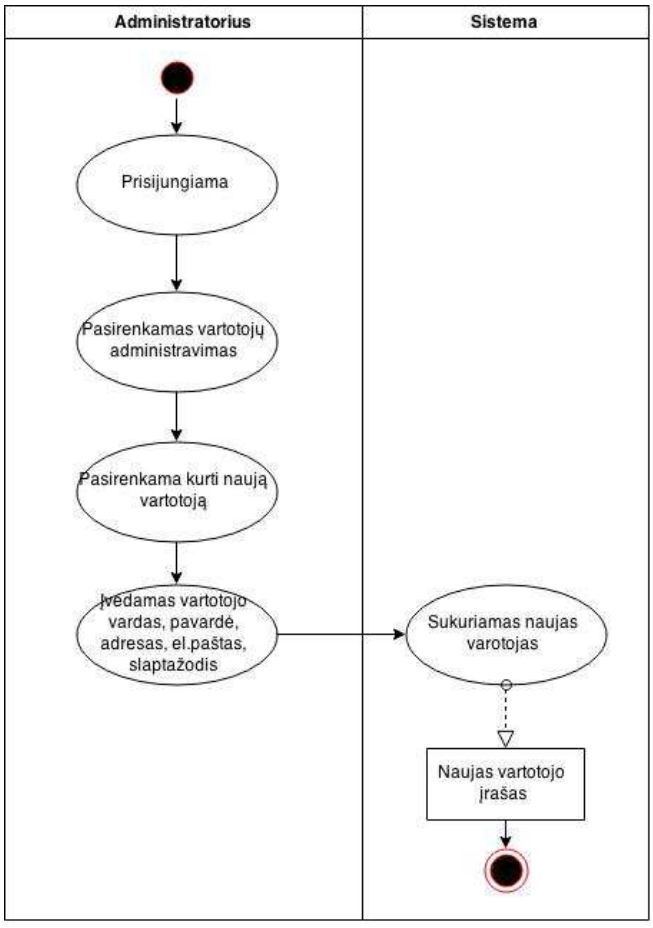
30 pav. *Prekės šalinimo funkcijos veiklos diagrama*



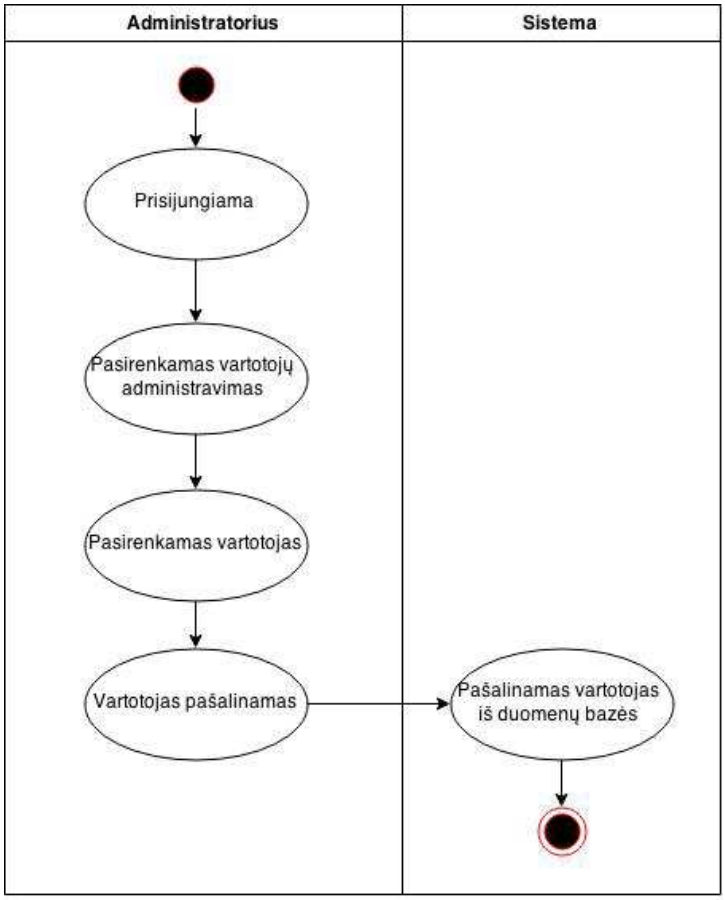
31 pav. *Prekės redagavimo funkcijos veiklos diagrama*



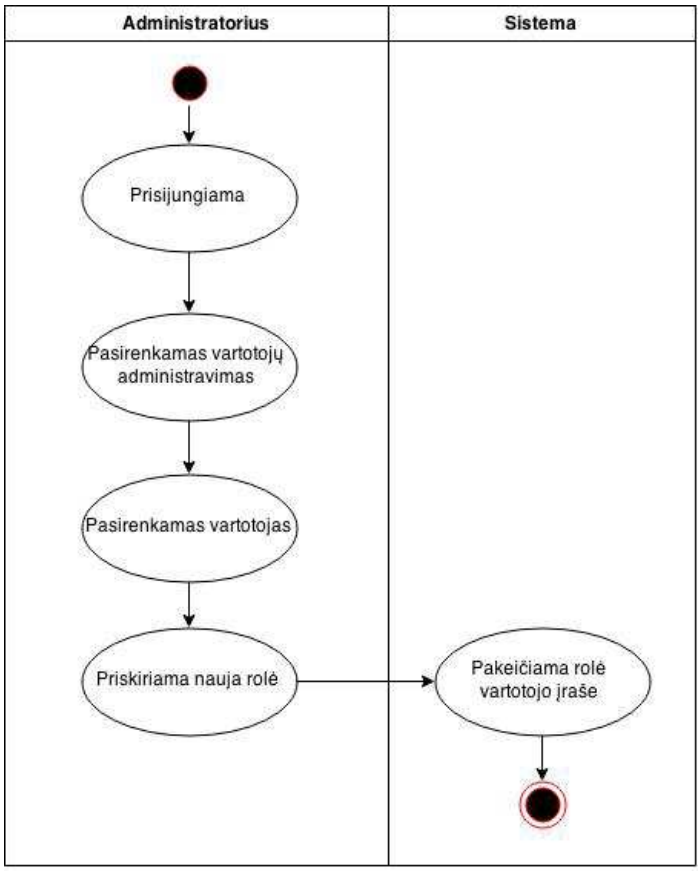
32 pav. *Vartotojų administravimo funkcijos veiklos diagrama*



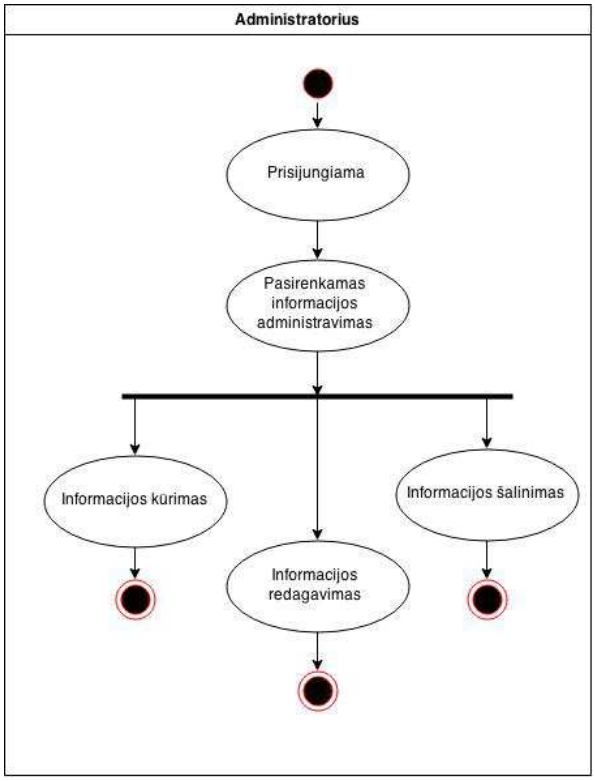
33 pav. *Naujo vartotojo kūrimo funkcijos veiklos diagrama*



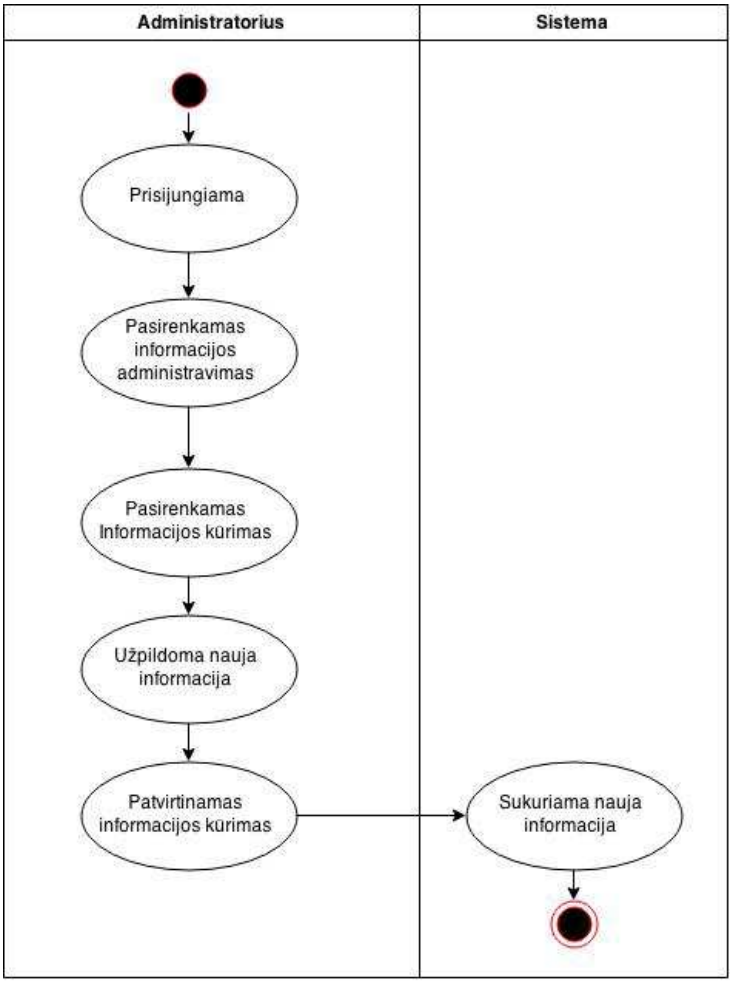
34 pav. *Vartotojo šalinimo funkcijos veiklos diagrama*



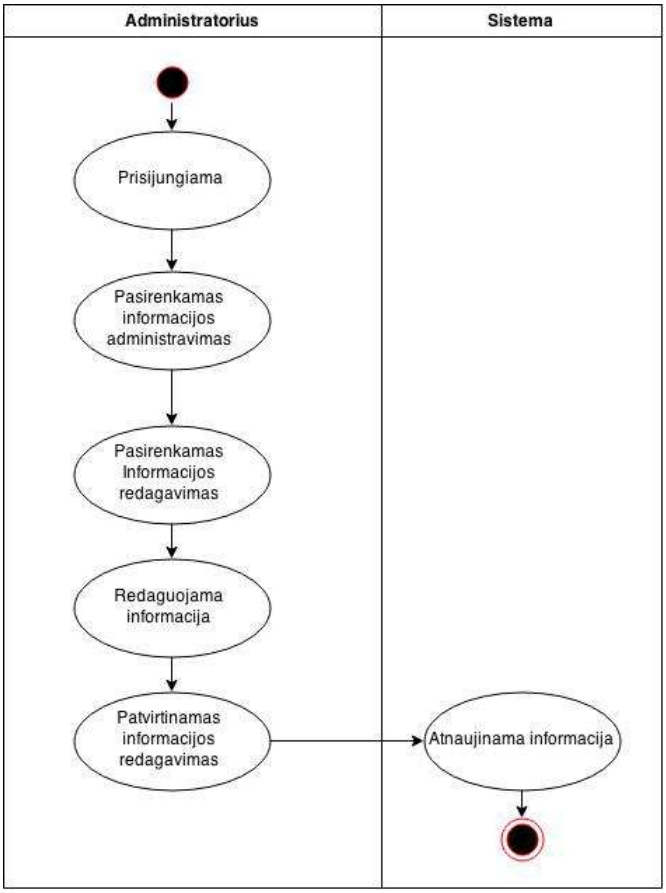
35 pav. *Vartotojo rolės keitimo funkcijos veiklos diagrama*



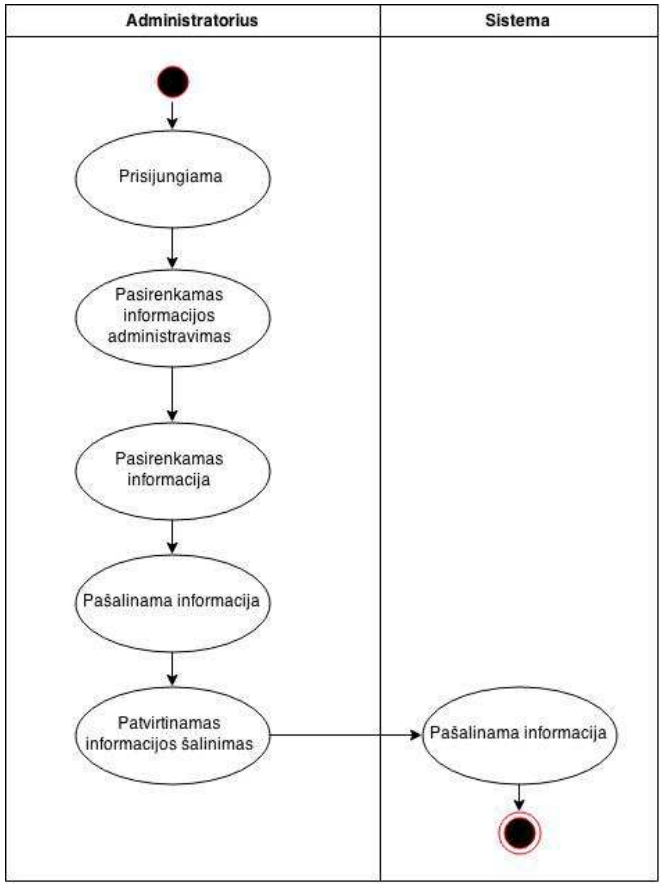
36 pav. *Informacijos administravimo funkcijos veiklos diagrama*



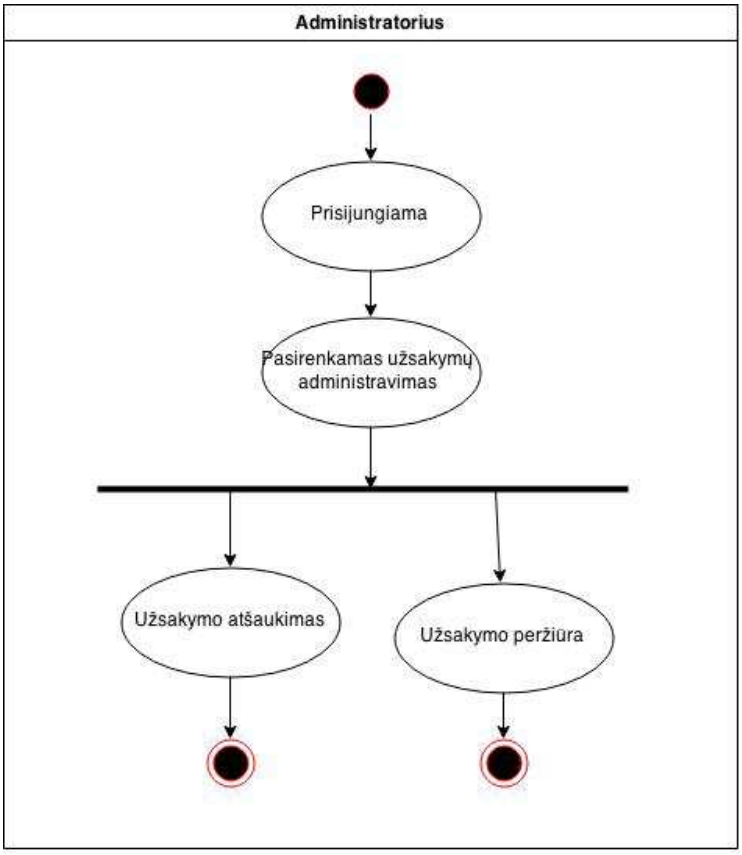
37 pav. *Naujos informacijos kūrimo funkcijos veiklos diagrama*



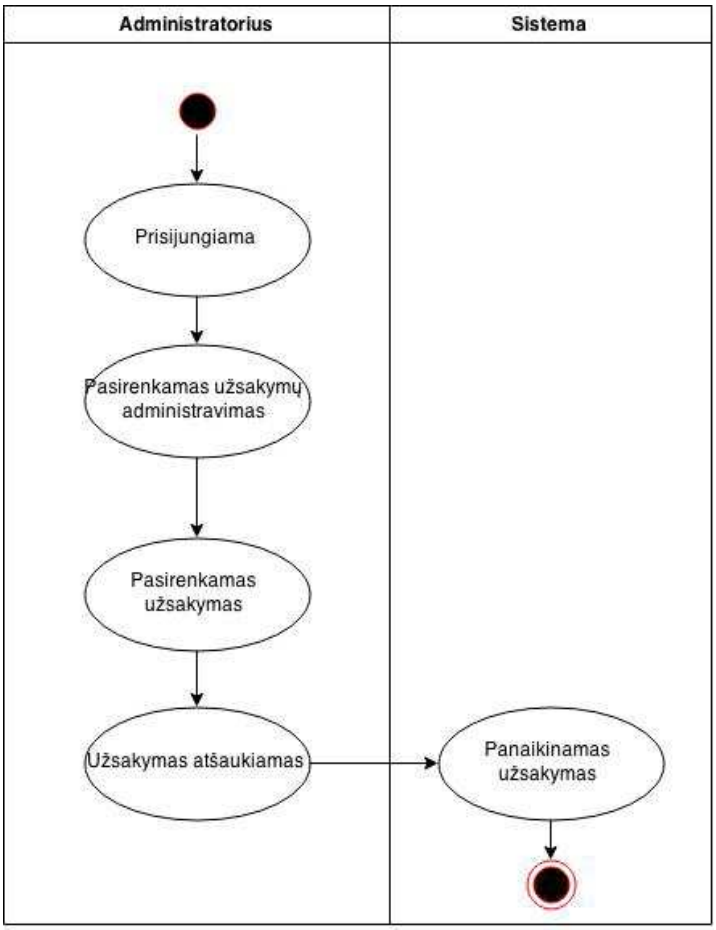
38 pav. *Informacijos redagavimo funkcijos veikos diagrama*



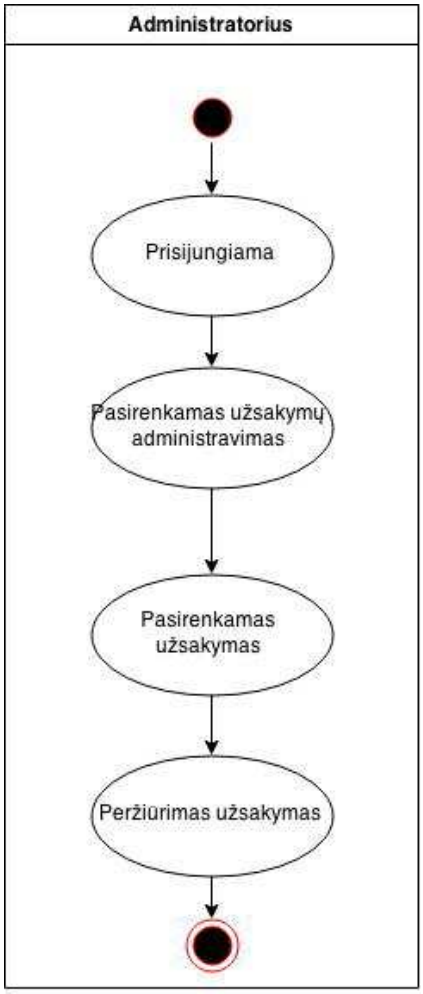
39 pav. *Informacijos šalinimos funkcijos veiklos diagrama*



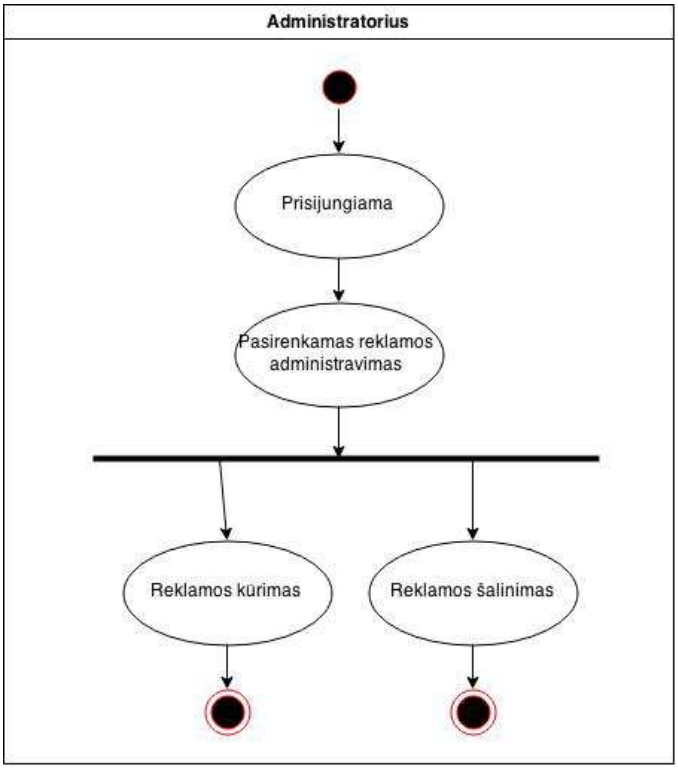
40 pav. *Užsakymų administravimo funkcijos veiklos diagrama*



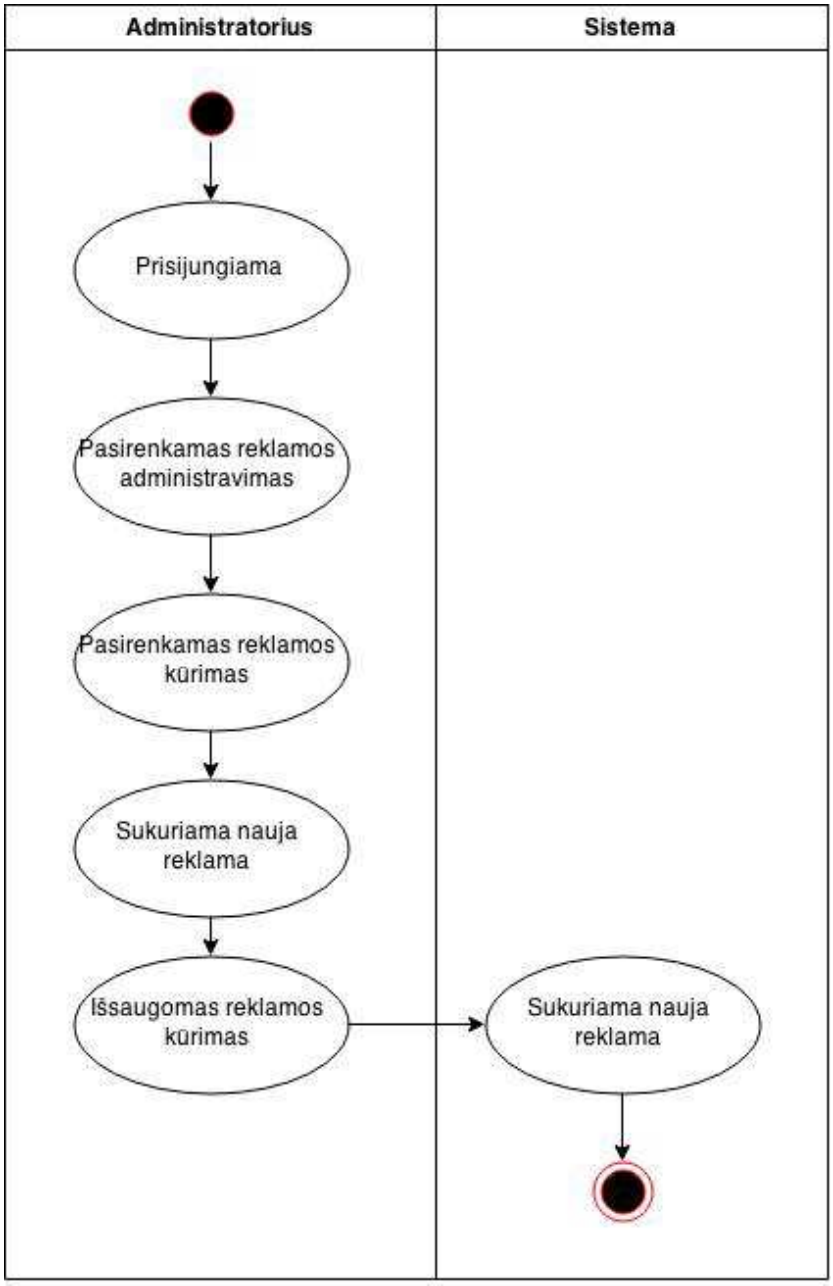
41 pav. *Užsakymo atšaukimo funkcijos veiklos diagrama*



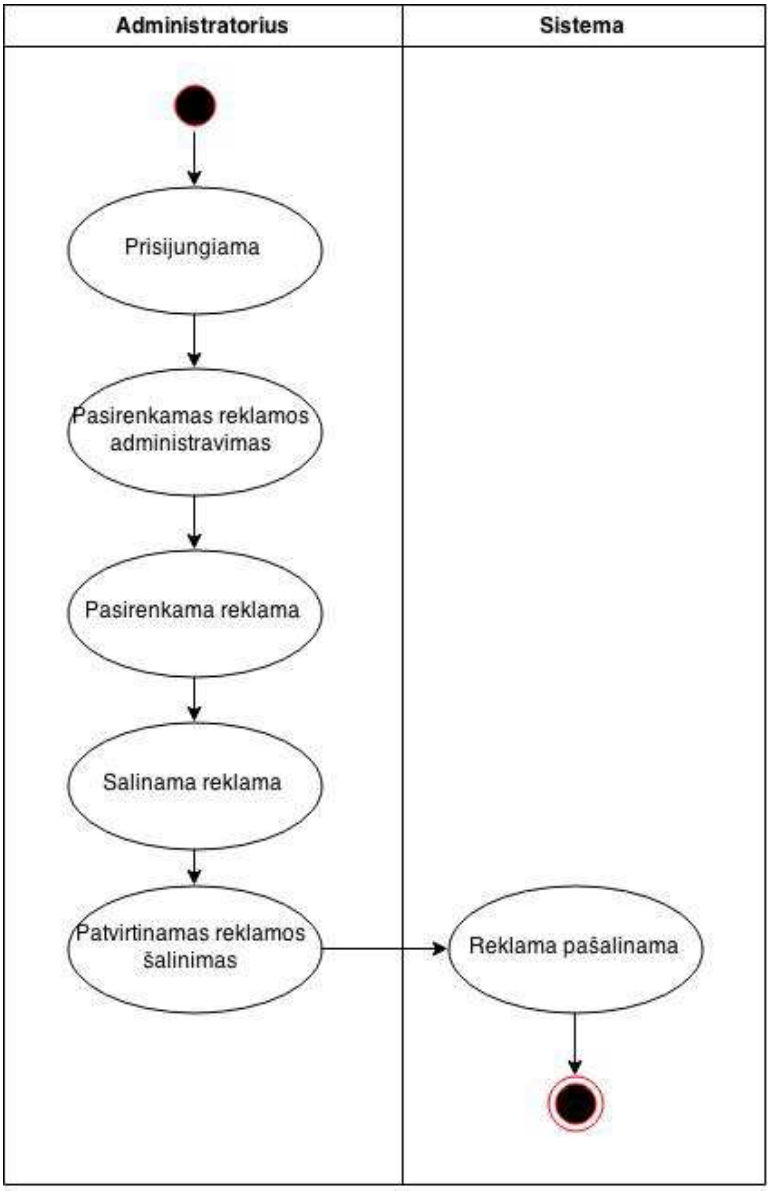
42 pav. *Užsakymo peržiūros funkcijos veiklos diagrama*



43 pav. *Reklamos administravimo funkcijos veikos diagrama*



44 pav. *Reklamos kūrimo funkcijos veikos diagrama*



45 pav. *Reklamos šalinimo funkcijos veiklos diagrama*