KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

Laura Kovalenkovienė

PROGRAMINĖS ĮRANGOS KŪRIMO PROJEKTŲ APIMTIES SKAIČIAVIMO METODŲ TYRIMAS

Baigiamasis magistro projektas

Vadovas
doc. dr. L. Čeponienė

KAUNAS, 2018

KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

PROGRAMINĖS ĮRANGOS KŪRIMO PROJEKTŲ APIMTIES SKAIČIAVIMO METODŲ TYRIMAS

Baigiamasis magistro projektas
Informacinių sistemų inžinerijos studijų programa (kodas 621E15001)

Vadovas
doc. dr. L. Čeponienė
2018-05-23

Recenzentas
lekt. dr. E. Mickevičiūtė
2018-05-23

Projektą atliko
Laura Kovalenkovienė
2017-05-23

KAUNAS, 2018

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

(Fakultetas)

Laura Kovalenkovienė

(Studento vardas, pavardė)

Informacinių sistemų inžinerijos studijų programa, 621E15001

(Studijų programos pavadinimas, kodas)

Baigiamojo projekto „Programinės įrangos kūrimo projektų apimties įvertinimo metodų tyrimas“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

20 ___ m. _____ ____ d.

Kaunas

Patvirtinu, kad mano, **Lauros Kovalenkovienės**, baigiamasis projektas tema „PROGRAMINĖS ĮRANGOS KŪRIMO PROJEKTŲ APIMTIES SKAIČIAVIMO METODŲ TYRIMAS“ yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

(vardą ir pavardę įrašyti ranka)	*(parašas)*

Kovalenkovienė, Laura. PROGRAMINĖS ĮRANGOS KŪRIMO PROJEKTŲ APIMTIES SKAIČIAVIMO METODŲ TYRIMAS. Magistro baigiamasis projektas / vadovas doc. dr. Lina Čeponienė; Kauno technologijos universitetas, Informatikos fakultetas.

Mokslo kryptis ir sritis: Informatikos inžinerija, technologijos mokslai

Reikšminiai žodžiai: *COSMIC, UCP, story points, programos apimtis, programos dydis.*

Kaunas, 2018. 71 p.

SANTRAUKA

PĮ kūrimo projekto apimties įvertinimas yra sudėtinga ir labai svarbi projekto vystymo dalis. Sėkmingai projekto eigai svarbu kuo anksčiau ir kuo tiksliau prognozuoti jo apimtį, biudžetą, įvykdymo terminą. Šiam tikslui pasiekti, svarbu išsirinkti tinkamą metodą labiausiai atitinkantį vertinamo projekto charakteristikas ir kontekstą. Egzistuoja daug skirtingų PĮ apimties įvertinimo metodų, todėl pasirinkimas tinkamo tampa iššūkiu projektų vadovams.

Šio darbo tikslas - palengvinti tinkamiausio metodo ar metodų, skirtų programinės įrangos kūrimo projektų apimties vertinimams, pasirinkimą. Tikslui pasiekti, atlikta apimties įvertinimo metodų analizė, išskirtos projektų charakteristikos, darančios įtaką metodo pasirikimui. Sprendimui įgyvendinti suprojektuota ir realizuota ekspertinė sistema.

Atliktas eksperimentas, kurio metu buvo siekiama išsiaiškinti potencialių vartotojų pasitenkinimą sistema ir galimybes tokią sistemą naudoti praktikoje. Eksperimento rezultatai parodė, jog vartotojai teigiamai vertina šią sistemą, taip pat buvo gauti naudingi patarimai tolimesniam ekspertinės sistemos tobulinimui.

Kovalenkovienė, Laura. *RESEARCH ON SOFTWARE DEVELOPMENT EFFORT ESTIMATION APPROACHES*: Master‘s thesis in Information Systems Engineering / supervisor assoc. prof. Lina Čeponienė. The Faculty of Informatics, Kaunas University of Technology.

Research area and field: Informatics Engineering, Technology Science

Key words: *COSMIC, UCP, story points, software effort estimation, software size.*

Kaunas, 2018. 71 p.

SUMMARY

Software effort estimation is a complex and important part of the project development process. It is very important to predict accurately project‘s scope, budget and deadline for successful project implementation. To achieve this, project manager has to make right decision which software effort estimation method is the most suitable for current project characteristics and context.

The purpose of this paper is to facilitate the selection of the best software effort estimation method for current situation. To achieve the goal, an analysis of effort estimation methods was performed and project characteristics, influencing the choice of the method, were distinguished. An expert system for software effort estimation methods was implemented and presented for real users.

An experiment was conducted to find out the practical applicability and usefulness of the system. The results of this experiment showed, that this recommendation system might be used in practice and some useful tips were given for further improvements.

TURINYS

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

TERMINŲ IR SANTRUMPŲ ŽODYNAS

PĮ	programinė įranga
IS	informacinė sistema
FNR	funkciniai naudotojų reikalavimai
UAW	aktoriaus svoris (angl. *unadjusted actor weights*)
UUCW	panaudojimo atvejų svoris (angl. *unadjusted use case weights*)
FP	funkciniai taškai (angl. *function points*)
LOC	kodo eilutės (angl. *lines of code*)
VAF	FP metode skaičiuojamas koreguojantis koeficientas
Scrum	*Agile* projektų valdymo principais paremtas metodas
Agile	projektų valdymo metodika
SP	vartotojų taškai (angl. *story points*)
XP	ekstremalus programavimas (angl. *extreme programming*)
UCP	panaudojimo atvejų taškų metodas (angl. *use case points*)
COSMIC	funkcinės apimties vertinimo metodas
FSM	funkcinės apimties matavimas
Mak II	funkcinės apimties vertinimo metodas
NESMA	funkcinės apimties matavimas
MRE	santykinė paklaida *(angl. Magnitude of Relative Error)*
MMRE	vidutinė santykinė paklaida (angl. *Mean Magnitude of Relative Error)*
PRED	prognozavimo kokybė (angl. *Percentage of the Prediction*)
SVR	atraminių vektorių regresija (angl. *Support Vector Regression*)
AMSE	Pakoreguota vidutinė kvadratinė paklaida (angl. *Adjusted Mean Square Error*)

ĮVADAS

Kasdien susiduriame su įvairiomis informacinėmis sistemomis. Nuolat auga poreikis kurti vis naujas, ar tobulinti jau esamas informacines sistemas. Planuojant, vykdant programinės įrangos kūrimo projektą, svarbu kuo anksčiau ir kuo tiksliau prognozuoti jo apimtį, biudžetą, įvykdymo terminą. Kai kurios įmonės, vertindamos naujo programinės įrangos kūrimo projekto apimtį, vadovaujasi savo, savo verslo partnerių ar kitų įmonių patirtimi ir būsimą projektą lygina su panašiais, jau įgyvendintais projektais. Tokiu būdu numatoma projekto trukmė, biudžetas, reikalingas specialistų kiekis. Tokie skaičiavimai yra preliminarūs, taip didinama rizika patirti nuostolius. Dažnai nuostolių dydis tiesiogiai proporcingas projekto dydžiui. Norint optimaliai įvertinti informacinės sistemos kūrimo projekto apimtį, reikia, tinkamai pasirinkti projekto apimties skaičiavimo metodiką. Yra įvairūs metodai, kurie padeda išmatuoti informacinių sistemų naudingumą, reikalingumą, tikslumą, paprastumą, vertę. Tinkamai įvertinę šiuos ir panašius kriterijus, galime kokybiškai valdyti projektą. Taigi, sėkmingai projekto eigai, svarbus uždavinys – išsirinkti tinkamiausią metodą ar metodus projekto apimčiai įvertinti. Kita vertus, šių metodų vis daugėja ir projekto vykdytojai susiduria su didele painiava rinkdamiesi optimalų skaičiavimo metodą.

Šio tyrimo tikslas - palengvinti tinkamiausio metodo ar metodų, skirtų programinės įrangos kūrimo projektų apimties vertinimams, pasirinkimą.

Tyrimo tikslui pasiekti bus sprendžiami šie uždaviniai:

1. išanalizuoti kelis skirtingus metodus, skirtus programinės įrangos kūrimo projektų apimties skaičiavimams;
2. palyginti šiuos metodus tarpusavyje;
3. parengti klausimų ir rekomendacijų rinkinį;
4. automatizuoti klausimų pateikimo ir išvadų generavimo procesą sukuriant prototipą;
5. eksperimentiškai pasiūlyti sukurtą įrankį projektų vadovams;
6. atlikti eksperimento metu gautų rezultatų analizę.

Projektų vadovai kasdien susiduria su gausybe problemų bandydami apskaičiuoti projekto kainą ar trukmę. Viena iš jų – nėra vieningų rekomendacijų, kurios padėtų apsispręsti kuris metodas labiausiai tiktų vienam ar kitam projektui vertinti. Šiam apsisprendimui palengvinti siūlomas sprendimas – ekspertinė sistema, kuri įvertinusi projekto charakteristikas, rekomenduoja apimties įvertinimo metodą.

Darbas sudarytas iš keturių skyrių, išvadų, literatūros sąrašo ir priedų. Pirmame skyriuje atliekama apimties įvertinimo metodų analizė, tiriamųjų darbų apžvalga, PĮ kūrimo projektų valdymo metodikų apžvalga. Antrame skyriuje aprašomi PĮ apimties įvertinimo metodo pasirinkimo sprendimo reikalavimai, reikalingi duomenys sprendimo įgyvendinimui, aprašomas žinių vaizdavimo modelis.

Trečias skyrius ekspertinės sistemos realizacijos projektui bei testavimui aprašyti. Ketvirtame skyriuje aprašomas eksperimentas, bei jo metu gauti rezultatai.

1. PROGRAMINĖS ĮRANGOS KŪRIMO PROJEKTŲ APIMTIES ĮVERTINIMO METODŲ ANALIZĖ

1.1. Analizės tikslas

Analizės tikslas – išsiaiškinti PĮ kūrimo projektų apimčiai įvertinti naudojamų metodų taikymo galimybes, jų pranašumus ir trūkumus.

1.2. Tyrimo objektas, sritis ir problema

„Projektų valdymo institutas (angl. *Project Management Institute*) siūlo tokį projekto apibrėžimą:

Projektas – tai laikina veikla, kuria siekiama sukurti unikalų produktą ar paslaugą“ [1].

Projektas iš kitų veiklų išsiskiria tuo, jog turi aiškią pradžią ir pabaigą, o jo įgyvendinimui skiriamas aiškiai apibrėžtas biudžetas bei resursai.

Anot knygos „Informacinių sistemų projektų ir kokybės valdymas“ [1] autorių, projektai gali būti skirstomi į:

- tipinius projektus, kurie gali būti atkuriami skirtingose situacijose, juos pakoregavus pagal vietos sąlygas;
- unikalius projektus, kurie negali būti tiražuojami dėl to, kad jų situacijos nesikartoja.

Projektai gali būti skirstomi į grupes pagal jų taikymo sritį:

- inžineriniai projektai,
- produkto (paslaugos) kūrimo projektai,
- sistemų kūrimo projektai,
- tyrimų bei organizacijų pertvarkymų projektai.

Šio magistro darbo tyrimo sritis - informacinių sistemų kūrimo projektai. Žvelgiant į praeitį, kuriamos PĮ vis sudėtingėjo, jų poreikis augo, todėl ir PĮ kūrimo projekto apimties skaičiavimo būdų atsirado ne vienas.

Projekto apimtis (angl. *project scope*) – projektui įvykdyti reikalingų darbų kiekis ir dydis.

Kuriamos PĮ projekto apimties vertinimą galima suskirstyti į 4 etapus:

- įvertinamas kuriamos PĮ dydis;
- apskaičiuojama projekto apimtis žmogaus darbo valandomis;
- apskaičiuojamas projekto tvarkaraštis kalendoriniais mėnesiais;
- įvertinama projekto kaina.

Tinkamai apskaičiuota projekto apimtis leidžia teisingai įvertinti projekto kainą ir projekto trukmę. Tačiau tikslus kuriamos PĮ projekto apimties vertinimas yra iššūkis projektų vadovui, ypač ankstyvoje projekto kūrimo fazėje. Iššūkis todėl, kad IT projektai dažnai yra unikalūs, sudėtingi,

užsakovui nesuprantami. Klaidingai apskaičiuota projekto kaina ar trukmė generuoja riziką patirti didelius finansinius nuostolius, pabloginti santykius su klientu.

Pirmas žingsnis, kurį reikia atlikti, norint įvertinti kuriamos PĮ projekto apimtį – kuo tiksliau apskaičiuoti pačios programinės įrangos dydį. Vertinant PĮ dydį ankstyvuose projekto etapuose, reiktų remtis įvairiais turimais formaliais dokumentais – kliento reikalavimų specifikacija, programinės įrangos preliminaria ar patvirtinta FNR specifikacija. Jei PĮ dydžio skaičiavimai atliekami pakartotinai jau vėlesniuose projekto etapuose, tuomet galima naudotis projektavimo dokumentų informacija. Net ir formalių dokumentų trūkumas nėra kliūtis atliekant kuriamos PĮ dydžio skaičiavimus. Tokiu atveju tikslinga naudotis bet kokia turima informacija apie būsimą sistemą, o gavus tikslesnius duomenis, būtina atlikti naujus skaičiavimus.

Naudojami įvairūs būdai kuriamos PĮ apimčiai pamatuoti:

- PĮ dydžio skaičiavimas paremtas analogiškais, praeityje darytais projektais;
- formalūs PĮ dydžio vertinimo metodai, standartai, paremti algoritminiais ar funkcinės apimties skaičiavimais.

Kiekvienas PĮ apimties vertinimo būdas turi savų privalumų ir trūkumų. Čia vėlgi susiduriama su problema, kada ir kokį būdą ar metodą geriausia naudoti, nes nėra vieno kriterijaus, pagal kurį galėtume teisingai pasirinkti. Daugelis organizacijų atlieka eksperimentus, daro tyrimus, užsako apklausas ir, remdamosi gautais rezultatais, renkasi esamoje situacijoje tinkamiausią būdą projekto apimčiai įvertinti [2].

1.3. Programinės įrangos kūrimo projektų valdymas

PĮ apimties vertinimo metodo pasirinkimas dažnai priklauso nuo projektui pasirinkto taikyti gyvavimo ciklo modelio. Projekto gyvavimo ciklo modelis diktuoja projekto valdymo taisykles, apibrėžia projekto fazes, jų vykdymo eiliškumą ir perėjimo iš vienos fazės į kitą kriterijus. PĮ kūrimo modelio pasirinkimui įtaką daro pats kuriamos sistemos tipas, organizacinė aplinka, kliento keliami reikalavimai. Apibendrinant, galima teigti, kad projekto gyvavimo ciklas – tai procesų, apimančių PĮ kūrimo etapus, seka. Šią seką sudaro tokie etapai: projekto inicijavimas, reikalavimų specifikavimas, projektavimas, programavimas, testavimas, sistemos priėmimas, sistemos naudojimas.

Remiantis literatūra [3] PĮ projektų kūrimo modelius galima suskirstyti taip:

- „kieta“ arba tradicinė projekto vystymo metodika (angl. *Heavyweight Development Methodology*);
- „lengva“ arba lanksti projekto vystymo metodika (angl. *Lightweight Development Methodology*);
- mišri projekto vystymo metodika (angl. *Hybrid Development Methodology*).

1.3.1. Tradiciniai projektų valdymo metodai

Tradicinės projektų vystymo metodikos pasižymi griežtomis taisyklėmis, sudėtinga dokumentacija. Populiariausi tradiciniai projektų vystymo modeliai [3], [4]:

- **krioklio** (angl. *waterfall*) **modelis** – tai klasikinis gyvavimo ciklo modelis. Taikant šį modelį, projektas vystomas nuosekliai, etapais - naujas etapas negali prasidėti tol, kol nebus baigtas vykdyti prieš jį esantis. Taikant šį modelį, PĮ kūrimo procesas yra lėtas, sudėtinga reaguoti į vartotojo reikalavimų pasikeitimą. Šis modelis labiausiai tinkamas projektuose, kuriuose yra tiksliai žinomi būsimos sistemos reikalavimai;
- **V-ciklo** (angl. *V-cycle*) **modelio** principas tas, kad kol nesibaigė prieš tai esanti projekto fazė, negali prasidėti sekanti. Lyginant su krioklio modeliu, šis yra lankstesnis, nes kai kurios projekto fazės gali būti vykdomos, validuojamos ir verifikuojamos tuo pačiu metu. Šis modelis tinkamiausias projektams, turintiems aiškius ir fiksuotus reikalavimus;
- **prototipų** (angl. *Prototyping*) **modelis**. Taikant šį modelį, PĮ kūrimo projektas vystomas kuriant nepilnas PĮ versijas, t.y. prototipus iki galutinio, reikalavimus atitinkančio produkto;
- **RUP** (angl. *Rational Unified Process*) unifikuotas procesas – tai PĮ kūrimo procesas, siūlantis taikyti iteracinį kūrimo modelį. Visas produkto gyvavimo ciklas sudarytas iš 4–5 fazių. Fazė gali būti sudaryta iš kelių iteracijų, trunkančių nuo 2 iki 6 savaičių. PĮ kūrimas pagal RUP metodiką yra panašus į krioklio metodą. RUP proceso pagrindinės dalys yra šios: pradinis etapas, pasirengimo etapas, kūrimo etapas, pereinamasis etapas. RUP sukurtas itin didelių projektų vystymui;
- **Iteracinis** (angl. *Iterative*) **modelis**, lyginant su krioklio modeliu, yra lankstesnis ir reikalauja mažiau informacijos apie kuriamą sistemą ankstyvose projekto stadijose. Tai leidžia klientui greičiau pamatyti rezultatą. Projektas padalinamas į atskiras mažesnes dalis – iteracijas. Kiekviena iteracija yra valdoma krioklio modelio principais. Kiekvienos iteracijos pabaigoje klientas gali pamatyti ir įvertinti atlikto darbo rezultatus – kuriamo produkto tam tikrą dalį.

1.3.2. Lankstieji (*Agile*) projektų valdymo metodai

Lankstieji projektų valdymo metodai tinkami tada, kai kinta reikalavimai ir PĮ kūrimas yra sunkiai kontroliuojamas. Toliau aprašomi populiariausi *Agile* projektų valdymo metodai:

- ***Scrum*** principais valdomas programinės įrangos kūrimo projektas vyksta trumpomis iteracijomis, kitaip vadinamomis sprintais. Kiekviena iteracija trunka keletą savaičių, dažniausiai 30 dienų [5]. Dėmesys sutelkiamas ne į detalias specifikacijas, o į kuriamos IS ar PĮ greitą pristatymą klientui. Programinės įrangos kūrimo procesas taikant *Scrum* metodiką yra lankstesnis lyginant su tradicinėmis projektų valdymo metodikomis. Vystant projektą *Scrum* metodikos principais, klientas turi galimybę keisti reikalavimus sistemai, atsisakyti arba sugalvoti naujų funkcionalumų ir pan. Vietoje projekto apimties skaičiavimų sudaromas darbų

sąrašas, kuris prioretizuojamas ir peržiūrimas kiekvienos iteracijos pradžioje. Dėl šių priežasčių ne visi IS apimties įvertinimo metodai tinkami naudoti PĮ, kuriamos pagal *Scrum* principus, dydžiui įvertinti.

- **ekstremalusis programavimas** [4] taikomas tada, kai reikia greitai ir dažnai reaguoti į būsimos sistemos pasikeitimus. Naudojant šį metodą, reikalavimų išsiaiškinimui, o juo labiau jų specifikavimui laikas nėra skiriamas, taip taupant kaštus ir iškart pereinant prie programavimo. Taikant ekstremaliojo programavimo metodiką, projekto vystyme intensyviai dalyvauja užsakovas. Kūrėjų komanda betarpiškai bendrauja tarpusavyje įvairiais, su projektu susijusiais, klausimais. Šio metodo trūkumas – pradinio etapo, skirto reikalavimų specifikavimui, nebuvimas gali padidinti projekto kaštus vėlesniuose etapuose.

1.3.3. Mišri projektų vystymo metodika

Ši projektų vystymo metodika taikoma tada, kai neužtenka vien tik tradicinių ar lanksčiųjų metodų.

- **Greitasis programų kūrimo modelis RAD** (angl. *Rapid Application Development*) [3] yra orientuotas į trumpą PĮ kūrimo ciklą, sukurtas derinant RUP ir *Scrum* metodikų savybes. RAD modelio esmė – yra greitas produkto kūrimas, kai reikalavimai sistemai yra aiškūs ir detaliai aprašyti.

Apibendrinant, galima teigti, jei projektas valdomas remiantis krioklio modeliu, tuomet apimties vertinimas turi būti atliekamas visam projektui iškarto. Jei taikomas iteracinis PĮ kūrimo modelis, apimtis vertinama kiekvienai iteracijai atskirai. Abiem atvejais PĮ apimties įvertinimą atlieka projektų vadovas. Jei projekto valdymas paremtas Agile principais, PĮ apimties įvertinimo procese dalyvauja visa projekto komanda. Apimties vertinimas gali būti atliekamas susirinkimų metu darbų prioretizavimo ir taškų priskyrimo darbui principu.

1.4. Programinės įrangos kūrimo projektų klasifikavimas pagal dydį.

PĮ projektų skirstymas pagal dydį yra interpretuojamas įvairiai. Vieną ir tą patį projektą skirtingi respondentai įvertintų skirtingai. Tai kas vienam atrodytų kaip mažas projektas, kitam būtų vidutinis ir atvirkščiai. Įvairiuose šaltiniuose teigiama, kad kuo didesnis projektas, tuo apimties įvertinimo tikslumas yra mažesnis. Taigi, galima daryti išvadą, kad rinktis didelio projekto apimties vertinimo metodą reikia ypač atsakingai. Toliau atliekama projektų skirstymo pagal dydį apžvalga.

Šaltinyje [6] apžvelgiami skirtingų autorių darbai ir jų siūlymai klasifikuoti PĮ projektus pagal dydį. Pavyzdžiui Watts‘as Hamphrey‘us siūlo PĮ produktus, atsižvelgiant į jų dydį, skirstyti į 5 etapus.

0 etapas Labai maži programų elementai, kuriuos kuria tik programuotojas.

1 etapas Mažos programos, kurias projektuoja, diegia ir testuoja tik programuotojas.

2 etapas Vidutinės programos ar jų komponenetai. Kuriant paprastai dalyvauja komanda.

3 etapas Labai dideli PĮ projektai. Kuriant dalyvauja daug komandų, kurioms vadovauja projektų valdymo syrius.

4 etapas Didžiulės sistemos, sudarytos iš atskirų didelių sistemų. Įtraukiami skirtingi autonominiai projektai ir jų komandos.

Solvita'os Berzisa'os [6] teigimu, PĮ projektus pagal dydį galima skirstyti atsižvelgiant į komandos dydį, projekto trukmę, projekto biudžetą. Taip pat, remiantis ISBSG (angl. *The International Software Benchmarking Standards Group*) bazėje sukauptų projektų duomenimis, dažnesni yra mikro ar maži projektai. Skirstant projektus pagal dydį, siūloma naudotis lentele (1.1 lentelė).

1.1 lentelė. PĮ projektų klasifikavimas pagal dydį, remiantis ISBSG [6]

Projekto dydis	Pastangos, darbo valandomis	Trukmė mėnesiais	Reikalingas personalo skaičius
Mažas	8–360	0–3	1 žmogus
Vidutinis	361–3600	3–6	3–7 žmonės
Didelis	3601–24000	6-12	7–24 pilnai apkrauti žmonės

Išanalizavę įvairių mokslininkų siūlymus, patirtį, požiūrį, šaltinio [6] autoriai siūlo tokį PĮ projektų skirstymą dydžiais (1.2 lentelė):

1.2 lentelė. PĮ projektų klasifikavimas pagal dydį

Projekto dydis	Apibūdinimas
Mikro	▪ Dydis (kodo eilutėmis) – nuo 1600 iki 9600 ▪ Kūrimo trukmė (mėnesiais) – nuo 1 iki 3 ▪ Kūrimo trukmė (valandomis) – nuo 160 iki 960 ▪ Komandos narių skaičius – 1–2 ▪ Kaina (doleriais) – 1200–4800*
Mažas	▪ Dydis (kodo eilutėmis) – nuo 9601 iki 38400 ▪ Kūrimo trukmė (mėnesiais) – nuo 3 iki 6 ▪ Kūrimo trukmė (valandomis) – nuo 961 iki 3840 ▪ Komandos narių skaičius – 2–4 ▪ Kaina (doleriais) – 7501–19200*
Vidutinis	▪ Dydis (kodo eilutėmis) – nuo 38401 iki 960000 ▪ Kūrimo trukmė (mėnesiais) – nuo 7 iki 60 ▪ Kūrimo trukmė (valandomis) – nuo 3841 iki 96000 ▪ Komandos narių skaičius – 5–10 ▪ Kaina (doleriais) – 192050–480000*
Didelis	▪ Dydis (kodo eilutėmis) – nuo 960000 ▪ Kūrimo trukmė (mėnesiais) – nuo 61 ▪ Kūrimo trukmė (valandomis) – nuo 96001 ▪ Komandos narių skaičius – nuo 11 ▪ Kaina (doleriais) – nuo 480000*
	*kaina skirtingose šalyse yra skirting. Analizuotame šaltinyje taikyta kaina 50 dol. už 1 žmogaus darbo valandą.

1.5. Programinės įrangos apimties skaičiavimo metodų apžvalga

Tikslus PĮ apimties vertinimas yra itin svarbus sėkmingam projekto vykdymui. Nesvarbu ar projekto apimtis nuvertinama ar pervertinama, projektas įstumiamas į pavojų. Netikslaus skaičiavimo pasekmės gali būti įvairios:

- nuvertintas projektas nebus atliktas laiku ir/arba bus viršytas jo biudžetas;
- pervertintas projektas gali tapti nebepatrauklus užsakovui ir jis jo atsisakys;
- pervertintas projektas beveik niekada nebus įvykdytas greičiau, nors tiksliai apskaičiuotų pastangų atveju jo gyvavimo ciklas sutrumpėtų;
- projektas gali būti nutrauktas vykdymo eigoje, tuomet tiek užsakovas, tiek vykdytojas patiria didžiulius nuostolius.

Per pastaruosius kelis dešimtmečius buvo sukurta daug naujų ar modifikuota jau esamų metodų PĮ kūrimo projektų apimčiai vertinti. Šiuos metodus galima suskirstyti į tokius tipus [7]:

- ekspertiniai vertinimo metodai;
- formalūs vertinimo metodai;
- mišrūs vertinimo metodai.

Apie ekspertinius vertinimo metodus šiame darbe plačiau nebus kalbama, nes šių metodų taikymas pasižymi subjektyviais ir neatkartojamais rezultatais. Toliau bus apžvelgiami formalūs vertinimo metodai.

1.5.1. Kodo eilučių skaičiavimas

Seniausias dydis naudotas PĮ apimčiai įvertinti – kodo eilučių (angl. *lines of code*) skaičiavimas. Galimi trumpiniai LOC, SLOC, KLOC. Šis matavimo būdas žinomas jau nuo 1960 metų, o pirmą kartą publikuotas 1976 metais. Kodo eilučių skaičiavimas yra itin paprastas ir visiems suprantamas, tačiau šis būdas buvo tinkamas naudoti tuo metu, kuomet programavimo kalbos buvo žemo lygio [8], [9]. Tarkim, programa parašyta Visual C++ kalba ir programa parašyta *Asembler* kalba, negali būti lyginamos taikant fizinių kodo eilučių skaičiavimo būdą. Net ir tos pačios programavimo kalbos kontekste, kodo eilučių skaičiavimas gali būti dviprasmiškas. Pavyzdžiui, skaičiuojant logines kodo eilutes, nėra aiškaus sutarimo kaip elgtis su komentarais, duomenų ataskaitomis, todėl kaskart reikia susidaryti taisyklių rinkinį šiam būdui taikyti. Programuotojo stilius taip pat turi nemenką įtaką kodo eilučių skaičiui, o tai duoda netikslius rezultatus analizuojant ir lyginant istorinius PĮ apimties matavimo duomenis. LOC skirtas naudoti vėlyvuose PĮ kūrimo etapuose. Nors yra nemažai trūkumų taikant šį būdą, jis yra naudojamas ir dabar. Viena iš priežasčių – organizacijos turi sukaupę nemažai tokių skaičiavimų istorinių duomenų ir juos sėkmingai naudoja analizei atlikti. Be to, kodo eilučių skaičiavimą galima automatizuoti ir tai sutaupo laiko ir pastangų vertinant projektą.

Kuriamos PĮ kaštų vertinimui taip pat naudojami algoritminiai kaštų modeliai, tokie kaip konstruktyvusis kainos modelis COCOMO (angl. *COnstructive COst MOdel*), PUTNAM modelis

(SLIM) ir kt. Apžvalga pateikiama šaltinyje [10]. Dauguma algoritminių kaštų modelių remiasi kodo eilučių skaičiumi, todėl jie nėra tinkami naudoti ankstyvose IS kūrimo projekto fazėse ir šiame darbe plačiau apie juos nebus rašoma. Sekančiame skyriuje analizuojami funkcinės apimties matavimo metodai.

1.5.2. Funkcinių taškų metodas

Projektų vadovo siekiamybė - kuo tiksliau įvertinti projekto apimtį ankstyvoje projekto stadijoje. IS funkciniai reikalavimai – tai tie duomenys, kurie žinomi ankstyvose stadijose ir jais remiantis galima apskaičiuoti PĮ projekto apimtį.

1979 m. Allan J. Albrecht‘as pristatė funkcinių taškų (angl. *Function points*) [11] metodą PĮ apimties matavimui, kaip geresnę alternatyvą kodo eilučių skaičiavimu pagrįstiems metodams. Šis metodas gali būti taikomas ankstyvesnėse projekto stadijose, kai analizuojami PĮ reikalavimai ar pradedami projektavimo darbai. Funkciniais taškais matuojamas funkcionalumas, kurį suteikia įrankis arba atskira jo dalis, neatsižvelgiant į technines arba kokybines įrankio specifikacijas.

Naudojant funkcinių taškų metodą, identifikuojami kuriamos PĮ funkciniai taškai, kurie skirstomi į tipus [12]:

- ✓ išoriniai įvedimai (angl. *External inputs*) – tai duomenų įvedimo procesas iš sistemos išorės;
- ✓ išoriniai išvedimai (angl. *External outputs*) – tai procesas, siunčiantis duomenis į programos išorę;
- ✓ vidiniai loginiai failai (angl. *Internal logical file*) – tai gali būti vidiniai failų tipai, duomenų bazės lentelės, kur saugomi programos duomenys;
- ✓ išorinės vartotojo sąsajos (angl. *External interface*) – tai ryšys su išoriniai failais ar lentelėmis, kuriuo perduodami matuojamos PĮ duomenys;
- ✓ išorinės užklausos (angl. *External inquiry*) – tai duomenų išvedimas į išorę, nekeičiant sistemos elgsenos.

1.3 lentelė. Funkcinių tašų sudėtingumo svoriai

FP tipai	Sudėtingumo svoriai		
	Žemas	Vidutinis	Aukštas
Išoriniai įvedimai	3	4	6
Išoriniai išvedimai	4	5	7
Vidiniai loginiai failai	7	10	15
Išorinės vartotojo sąsajos	5	7	10
Išorinės užklausos	3	4	6

Suskaičiavę nagrinėjamos programos FP tipus ir atitinkamai juos padauginę iš sudėtingumo svorių įverčių (1.3 lentelė), gauname nepakoreguotą funkcinių taškų dydį UFP (angl. *unadjusted function points*).

Tuomet skaičiuojamas koreguojantis koeficientas VAF (angl. *value adjustment factor*) – VAF.

Tam reikia įvertinti nagrinėjamos IS charakteristikas ir jų įtaką IS apimčiai. Įtaka įvertinama svoriu nuo 0 iki 5. Nagrinėjamos charakteristikos: transakcijų dažnumas, techninės įrangos panaudojimas, diegimo paprastumas, sistemos pakartotinis naudojimas, paskirstytas duomenų apdorojimas. Apskaičiavę bendrą IS charakteristikų svorį *N*, skaičiuojame koreguojantį koeficientą:

$$VAF = 0.65 + \frac{N}{100} \quad (1)$$

čia *0,65* – sudėtingumo reguliavimo faktorius, jo reikšmė gali svyruoti nuo 0,65 iki 1,35.

FP skaičiuojamas:

$$FP = UFP \times VAF \quad (2)$$

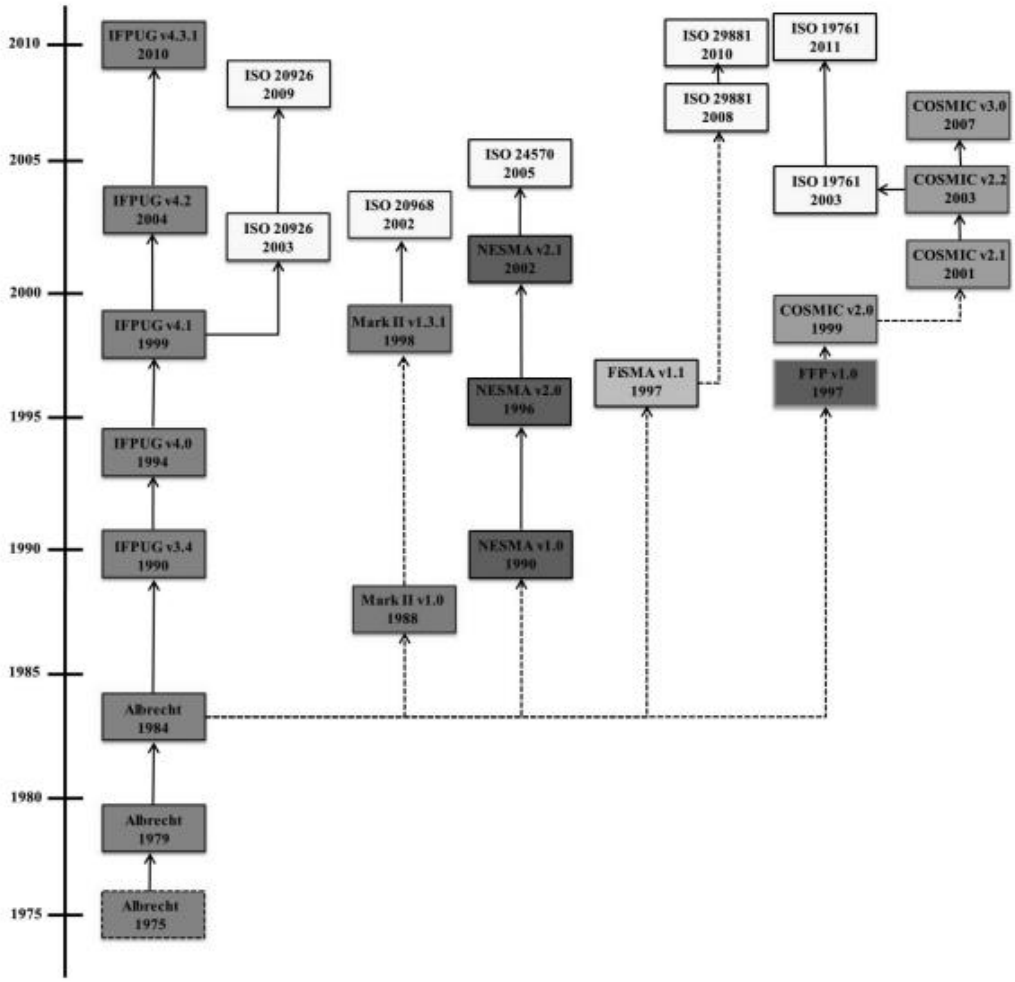
Funkciniai taškai gali būti konvertuojami į kodo eilutes ir atvirkščiai.

Funkcinės apimties matavimo metodų evoliucija

Tobulėjant programavimo kalboms, sudėtingėjant IS, atsirado poreikis tobulinti ar kurti naujus metodus PĮ apimčiai apskaičiuoti. Remiantis FP taikymo taisyklėmis buvo sukurti nauji PĮ apimties skaičiavimo metodai ir standartai iš kurių daugiausiai šaltiniuose minimi pagrindiniai, tapę ISO standartais [11], [13], [14], kurių grafinis evoliucijos vaizdas pateikiamas 1 pav. [15]:

a) pirmos kartos metodai:
   - ✓ IFPUG metodas (angl. *International Function Point Users Group method*), kuris PĮ dydį matuoja IFPUG funkciniais taškais (IFPUG FP);
   - ✓ Mark II funkcinių taškų metodas;
   - ✓ NESMA (angl. *Netherlands Software Metrics Association*) funkcinių taškų metodas, kurio matavimo vienetai NESMA funkciniai taškai (NESMA FP);

b) antros kartos metodai:
   - ✓ COSMIC (angl. *Common Software Measurement International Consortium*) funkcinių taškų metodas, PĮ dydį vertinantis COSMIC funkciniais taškais (CFP);
   - ✓ FiSMA (angl. *Finnish Software Measurement Association*) metodas.



1 pav. FSM metodų evoliucija [15]

Pirmos kartos metodai buvo sukurti verslo informacinių sistemų apimties vertinimui. Augant informacinių sistemų poreikiui ir didėjant jų įvairovei, atsirado poreikis atlikti tikslesnius apimties vertinimo skaičiavimus. Taip pat reikėjo metodų, kuriuos būtų galima pritaikyti ne tik verslo IS, bet ir realaus laiko ar kt. IS. Dėl šių priežasčių buvo sukurti antros kartos metodai. Šie metodai giriami dėl savo platesnių pritaikymo galimybių, tikslesnių skaičiavimo rezultatų, lyginant su pirmos kartos metodais. Trumpas funkcinės apimties matavimo metodų palyginimas pateiktas lentelėje (1.4 lentelė).

1.4 lentelė. Standartizuoti Funkcinės apimties matavimo metodai.

Savybės	Pirmos kartos metodai			Antros kartos metodai	
	IFPUG	Mark II	NESMA	COSMIC	FiSMA
Panaudojamumas (kokių tipų PĮ tinka)	verslo IS			verslo, infrastruktūros, realaus laiko IS; Agile projektai	įvairios IS
Prieinamumas (standartų aprašymai, taikymo metodika)	prieinama iš dalies			lengvai prieinama	
Rezultatų atkartojamumas	vertinamas blogai			vertinamas gerai (lyginant su pirmos kartos metodais)	
Skaičiavimo rezultatų subjektyvumas	subjektyvūs			subjektyvūs iš dalies	

Remiantis įvairiais šaltiniais [16], [17], [18], plačiausiai iš standartizuotų FSM metodų vis dar naudojamas IFPUG metodas. Nepaisant to, kad yra sukurti tobulesni apimties vertinimo metodai, IFPUG išlieka populiarus todėl, kad įmonės turi sukaupę daug istorinių duomenų, gautų taikant šį metodą. Dar 2009 m. amerikiečių GAO (angl. *Government Accountability Office*) rekomendavo naudoti tiek IFPUG tiek ir COSMIC metodą [19]. Visgi, COSMIC metodo populiarumas nuolat auga, ir prie to, neabejotinai, prisideda galimybė IFPUG funkcinius taškus paversti COSMIC funkciniais taškais. Taigi, įmonės, žinodamos, kad yra galimybė ir toliau naudotis sukauptais istoriniais duomenimis, vis dažniau renkasi naujesnį ir patikimesnį COSMIC metodą. Kaip teigiama šaltinyje [19], 2016 m. NIST (angl. *National Institute for Standards & Technology*) savo atlikto tyrimo rekomendacijose siūlo naudoti tik COSMIC metodą, kaip turintį aiškiai apibrėžtą matavimo dydį.

FSM metodų privalumas tas, kad šie metodai PĮ apimties vertinimui gali būti taikomi per visą sistemos kūrimo gyvavimo ciklą. Taip pat šiems metodams neturi įtakos naudojama programavimo kalba ar technologijos [11], [20]. Metodų trūkumas – skaičiavimo rezultatai subjektyvūs ir priklauso nuo žmogaus atliekančio skaičiavimus. Todėl net toje pačioje organizacijoje, šių metodų taikymas, duoda skirtingus rezultatus [20]. Viena iš rezultatų subjektyvumo mažinimo galimybių - skaičiavimo proceso automatizavimas. Būtent COSMIC metodas yra patrauklus lengvo automatizavimo galimybėmis.

1.5.3. COSMIC metodo analizė.

COSMIC metodas tai funkcinio dydžio matavimo metodas. COSMIC metodas gali būti naudojamas tokių tipų PĮ vertinti [21]:

- verslo;
- realaus laiko;
- infrastruktūros (pvz. operacinės sistemos);
- aukščiau išvardintų PĮ hibridams.

Bendra charakteristika jungianti aukščiau išvardintų tipų PĮ yra tai, kad jos dirba su įvesties, saugojimo ir atkūrimo bei išvesties duomenimis.

COSMIC metodas nėra tinkamas mokslo ar inžinerijos PĮ apimties vertinimui, kurių funkcija manipuliuoti duomenimis [21].

COSMIC metodo matavimo procesas susideda iš trijų etapų [11], [22]:

1. Įvertinimo (matavimo) strategijos etapas
2. Žymėjimo etapas.
3. Įvertinimo (matavimo) etapas.

Įvertinimo (matavimo) strategijos etape apibrėžiami matavimo tikslai ir apimtis. Čia išsiaiškinama kam reikalingas matavimas ir kur bus panaudoti gauti rezultatai. Šiame etape numatome, kurie funkciniai naudotojų reikalavimai (FNR) bus naudojami matavimo metu. Pagal FNR nustatomi ir funkciniai naudotojai. Funkciniai naudotojai yra PĮ funkcinių naudotojų reikalavimų duomenų siuntėjai ir/arba gavėjai.

Žymėjimo etape nustatomi matuojamos PĮ funkciniai procesai, duomenų grupės, į kurias kreipiasi matuojama PĮ bei duomenų judėjimai kiekviename funkciniame procese. Funkciniai procesai nustatomi iš FNR. Kiekvienam funkciniam procesui yra nustatomi duomenų judėjimai (įėjimas, išėjimas, skaitymas, rašymas).

Įvertinimo (matavimo) etape įvertinamas kiekvienas duomenų judėjimas ir gaunami matavimo rezultatai.

COSMIC apimties vertinimo metodas remiasi tokiais programinės įrangos inžinerijos principais:

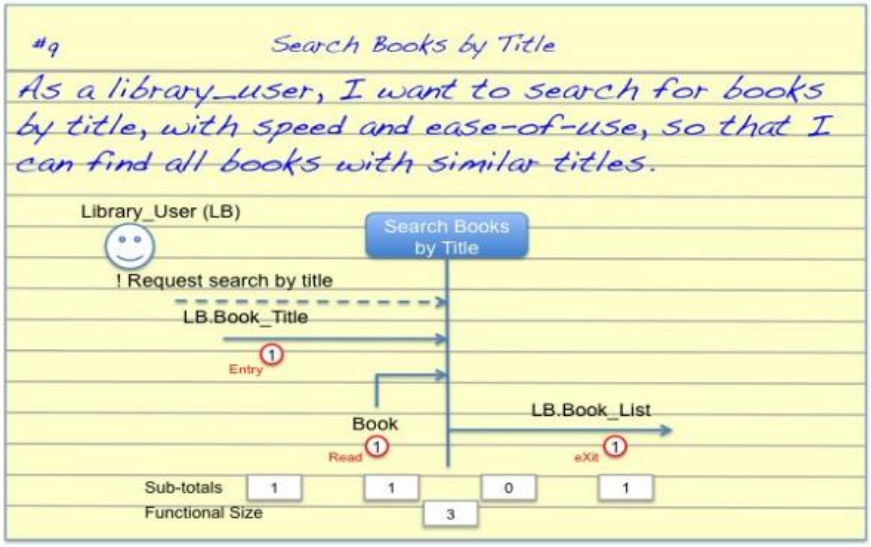
- programinės įrangos funkciniai vartotojų reikalavimai gali būti analizuojami kaip unikalūs funkciniai procesai, susidedantys iš duomenų judėjimo arba duomenų apdorojimo operacijų (sub-procesų);
- kiekvienas funkcinis procesas inicijuojamas funkcinio vartotojo perduodant informaciją programinei įrangai;
- funkcinis procesas laikomas baigtu kai atlikta viskas, kad numatytas įvykis būtų laikomas įvykdytu;
- duomenų judėjimo ar apdorojimo operacijos (sub-procesai) skirstomos į 4 tipus: *įėjimas, išėjimas, rašymas, skaitymas*.

Įėjimu (angl. *entry*) vadinamas toks duomenų judėjimas, kai funkcinio vartotojo siunčiami duomenys perduodami programinei įrangai; *išėjimu* (angl. *exit*) vadinamss informacijos išvedimas už informacinės sistemos ribų; *rašymas* (angl. *write*), tai tokia duomenų apdorojimo operacija, kada duomenys įrašomi į duomenų saugyklą; *skaitymas* (angl. *read*) yra duomenų apdorojimo operacija, kurios metu duomenys perkeliami (nuskaitomi) iš duomenų saugyklos.

Programinės įrangos dydis įvertinamas suskaičiavus visų funkcinių procesų duomenų judėjimo ir apdorojimo operacijas, t.y. suskaičiuojami visi sub-procesų tipai: *įėjimas, išėjimas, skaitymas, rašymas*. Kiekvienas jų matuojamas vienu COSMIC funkciniu tašku CFP (angl. *COSMIC function points*). Minimalus funkcinio proceso dydis arba programinės įrangos dalies dydis, negali būti mažesnis kaip 2 CFP. Maksimaliam dydžiui apribojimų nėra [23].

Funkcinių procesų duomenų judėjimo vaizdavimui dažnai braižomos sekų (angl. *sequense*) diagramos. Toks vaizdavimo būdas dažnai taikomas ir pagal *Scrum* kuriamos PĮ planavime. Kiekviena vartotojo istorija dažniausiai užrašoma ant atskiros kortelės, kuri naudojama planavimo susirinkimų

metu. Rekomenduojama praktika, ant kiekvienos kortelės kitos pusės piešti konkrečios vartotojo istorijos sekų diagramą. Toks pavyzdys pateikiamas 2 pav.



2 pav. Vartotojo istorija su sekų diagrama [5]

Sekų diagrama ant vartotojo istorijos kortelės sukuria geresnį supratimą apie kuriamą PĮ. Taip pat toks sekų diagramų taikymas yra itin sveikintinas, kai norima taikyti kelis PĮ apimties prognozavimo metodus – COSMIC ir Story Points [5].

Standartinio COSMIC metodo taikymas reikalauja nemažai laiko ir pastangų bei išsamiai specifikuotų reikalavimų. Tačiau kartais reikia atlikti PĮ dydžio įvertinimą ankstyvoje projekto gyvavimo fazėje arba, nors ir esant detaliems reikalavimams, reikalingas greitas dydžio paskaičiavimas. Tokiu atveju, siūloma naudoti COSMIC metodo versijas:

- ankstyvas metodo taikymas (angl. *early sizing);*
- greitas metodo taikymas (angl. *rapid sizing*).

Supaprastintų COSMIC metodų naudojimas leidžia pasiekti kompromisą, kai reikia atlikti projekto dydžio vertinimą, tačiau nėra laiko arba trūksta detalių reikalavimų.

1.5.4. Panaudojimo atvejų taškų metodas

Remiantis FP ir Mark II metodikomis [18], 1993 m. Gustav‘as Karner‘is pristatė dar vieną PĮ apimties skaičiavimo metodą, – panaudojimo atvejų taškų metodą UCP (angl. *use case points*). Šis metodas naudojamas prognozuoti kuriamos programinės įrangos apimtį ankstyvoje projekto stadijoje. [24] atliekama kitų autorių apžvalgų apie UCP metodą analizė. Analizuota 18 skirtingų mokslinių darbų. Autoriaus analizės rezultatai rodo, kad UCP metodas yra itin populiarus ir naudojamas praktikoje. Skirtingi autoriai šį metodą rekomenduoja kaip efektyvų ir tinkamą naudoti ankstyvose projekto vykdymo stadijose. Visgi metodas nėra standartizuotas ir reikalauja aiškesnių naudojimo gairių.

Pagrindinis panaudojimo atvejų (angl. *Use Case*) tikslas yra parodyti sąveiką tarp aktoriaus ir tarp projektuojamos sistemos atliekamų funkcijų. Ši sąveika atvaizduojama UML panaudojimo atvejų diagrama [18]. Aprašomi panaudojimo atvejai gali būti pateikti paprastoje lentelėje, nurodant panaudojimo atvejo pavadinimą, trumpą jo aprašymą, aktorius ir bendras pastabas.

Taikant UCP metodą PĮ apimčiai vertinti, pirmas žingsnis yra suskaičiuoti nepakoreguotus panaudojimo atvejų taškus UUCP (angl. *unadjusted use case points*). Tam reikia įvertinti nepakoreguotą aktoriaus svorį UAW (angl. *unadjusted actor weight*) (1.5 lentelė) ir nepakoreguotą panaudojimo atvejų svorį UUCW (angl. *unadjusted use case weights*) (1.6 lentelė), kuris priklauso nuo kiekviename panaudojimo atvejyje esančių transakcijų (scenarijaus žingsnių) skaičiaus (1.3 lentelė). [18], [25].

1.5 lentelė. Aktorių svoris

Svorio tipas	Aprašymas	Aktoriaus svorio įvertis (asvoris)
Paprastas	Aktorius atstovauja sistemą, kuri bendrauja su kitų taikomųjų programų aktoriais.	1
Vidutinis	Aktorius atstovauja sistemą, sąveikaujant per protokolus (pvz. HTTP, FTP)	2
Sudėtingas	Aktorius komunikacijai su sistema naudoja grafinę naudotojo sąsają.	3

Įvertinus aktorių svorį (asvoris), suskaičiavus aktorių kiekį (askaičius), yra apskaičiuojamas UAW kiekvienam svorio tipui, o jų suma bus bendras UAW:

$$UAW = \sum_{i \in C} asvoris(c) \times askaičius(c) \quad (3)$$

čia $C = \{paprastas, vidutinis, sudėtingas\}$.

Toliau vertinamas nepakoreguotas panaudojimo atvejų svoris UUCW (1.6 lentelė).

1.6 lentelė. Panaudojimo atvejų svoris.

Svorio tipas	Aprašymas	Panaudojimo atvejo svorio įvertis (usvoris)
1.Paprastas	Scenarijaus žingsnių skaičius <4.	5
2.Vidutinis	Scenarijaus žingsnių skaičius >= 4 ir <=7	10
3.Sudėtingas	Scenarijaus žingsnių skaičius >7	15

Nepakoreguotas panaudojimo atvejų svoris UUCW apskaičiuojamas įvertinus kiekvieno panaudojimo atvejo svorį *usvoris* ir suskaičiavus jo žingsnių skaičių *uskaičius*.

$$UUCW = \sum_{i \in C} usvoris(c) \times uskaičius(c) \quad (4)$$

čia $C = \{paprastas, vidutinis, sudėtingas\}$.

UUCP skaičiuojamas pagal formulę:

$$UUCP = UAW + UUCW \quad (5)$$

Sekančiame žingsnyje reikia įvertinti techninio sudėtingumo faktorių TCF (angl. *technical complexity factor*) ir aplinkos sudėtingumo faktorių ECF (angl. *environment complexity factor*).

Yra 13 techninių faktorių TF (1.7 lentelė), kiekvienas jų turi tam tikrą svorį. Kiekvienam faktoriui reikia priskirti reikšmę nuo 0 iki 5, kuri atspindėtų šio faktoriaus įtaką projektui (kur 0 – visiškai nesvarbi įtaka; 5 – didžiausia įtaka). Sudauginus kiekvieno TF svorį ir jam priskirtą įtaką ir gautus rezultatus susumavus, gauname projekto bendrą techninį faktorių TTF. Toliau skaičiuojame TCF pagal formulę:

$$TCF = 0.6 + (0.01 \times TTF) \quad (6)$$

1.7 lentelė. Techniniai faktoriai ir jų svoriai.

TF	Apibūdinimas	Svoris
T1	Sistemos pasiskirstymas	2
T2	Našumas	1
T3	Galutinio vartotojo efektyvumas	1
T4	Vidinio apdorojimo sudėtingumas	1
T5	Pakartotinis panaudojamumas	1
T6	Diegimo paprsatumas	0,5
T7	Naudojimo paprastumas	0,5
T8	Portatyvumas	2
T9	Keitimo paprastumas	1
T10	Lygiagretumas	1
T11	Saugumo funkcijos	1
T12	Tiesioginis priėjimas trečiosioms šalims	1
T13	Specialių vartotojų apmokymų reikalingumas	1

Aplinkos sudėtingumo faktorius ECF skaičiuojamas pirmiausia įvertinant projekto bendrą aplinkos faktorių ETF. ETF skaičiavimo metu įvertinama projekto komandos patirtis. Yra 8 aplinkos faktoriai EF (

1.8 lentelė), kurių kiekvienas turi savo svorį. Kiekvienas aplinkos faktorius įvertinamas priskiriant jam įtakos reikšmę nuo 0 iki 5 (kur 0 – nėra įtakos, 1 – stipri neigiama įtaka, 5 – stipri teigiama įtaka). Sudauginus kiekvieno EF svorį ir priskirtą įtakos reikšmę, bei gautus rezultatus susumavus, gauname projekto bendrą aplinkos faktorių ETF. Tuomet aplinkos sudėtingumo faktorius skaičiuojamas pagal formulę:

$$ECF = 1.4 + (-0.03 \times ETF) \quad (7)$$

1.8 lentelė. Aplinkos faktoriai ir jų svoriai

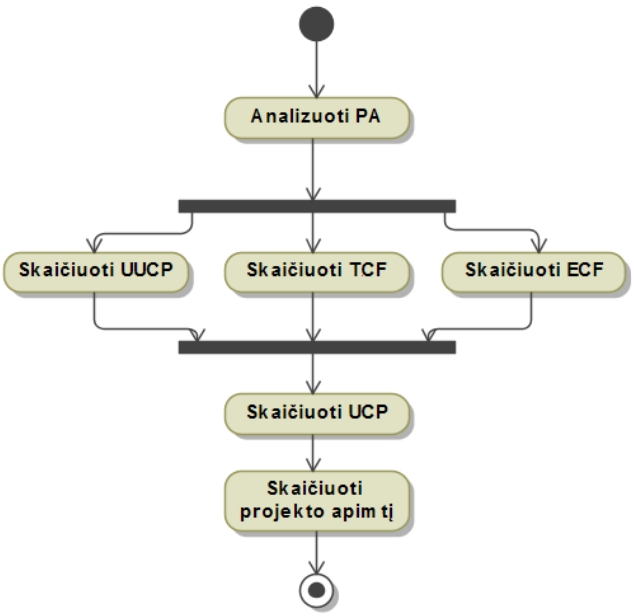
EF	Apibūdinimas	Svoris
E1	UML mokėjimas	1,5
E2	Patirtis kuriant panašias sistemas	0,5
E3	Objektinio kūrimo patirtis	1
E4	Vadovaujančio analitiko gebėjimai	0,5
E5	Motyvacija	1
E6	Reikalavimų stabilumas	2
E7	Darbuotojai dirbantys ne pilną darbo dieną	-1
E8	Programavimo kalbos sudėtingumas	-1

Toliau skaičiuojami panaudojimo atvejų taškai UCP:

$$UCP = UUCP \times TCF \times ECF \quad (8)$$

Norėdami projekto apimtį įvertinti reikalingo žmogaus darbo valandų skaičiumi T, reikia projekto UCP padauginti iš projekto produktyvumo faktoriaus PF (angl. *productivity factor*). Siūloma numatytoji PF reikšmė yra 20 žmogaus darbo valandų [18]. Numatytoji reikšmė naudojama tada, kai projekto komanda neturi istorinių projektų duomenų. Jei tokių duomenų yra, reiktų remtis jais ir projektui reikalingą PF apskaičiuoti taip: buvusio projekto faktinę apimtį žmogaus darbo valandomis padalinti iš to projekto UCP.

UCP metodo taikymo eiga projekto apimčiai skaičiuoti pavaizduota 3 pav..



3 pav. *Use Case Points* metodo eiga

1.5.5. *Agile* principais valdomų projektų apimties vertinimas

Agile principais valdomų projektų apimties įvertinimo procesas iš esmės skiriasi nuo vertinimo proceso, kai PĮ kūrimo projektas vystomas taikant tradicinę metodiką.

Šiuo metu itin populiaru IS kūrimo projektams taikyti *Agile* valdymo metodus: *Scrum, Kanban, Lean, FDD* (angl *Feature Driven Development*), *XP* (angl. *Extreme Programming*), *AUP* (angl. *Agile Unified Process*), *Agile modeling*, ir kt. *Agile* metodų esmė – PĮ kūrimas organizuojamas iteracijomis, t.y. visas projektas suskaldomas į kuo trumpesnius etapus. Taigi projektų, vykdomų taikant *Agile* metodiką, apimties vertinimas turi būti organizuojamas kitaip nei tradiciniuose projektuose. Naudojami įvairūs būdai *Agile* projektams įvertinti, tokie kaip *Wideband Delphi*, balti drambliai (angl. *white elephant*), panašumų (angl. *affinity*) vertinimas. Visi jie puikiai tinka trumpoms iteracijoms vertinti ir motyvuotoms komandoms. Daugumos komandų nariai, išskyrus projektų vadovą, atmestinai žiūri į projekto apimties įvertinimo procesą, todėl vertinimas, taikant anksčiau minėtus būdus, atliekamas atmestinai ir neorganizuotai [26]. Didesniam ir efektyvesniam komandos įtraukimui į vertinimo procesą, literatūroje minimi *Agile* principais valdomų projektų apimties vertinimo būdai, tokie kaip istorijos taškų SP (angl. *Story Points*) skaičiavimas ar planavimo pokeris (angl. *Planning Poker*) [9].

Vienas iš populiariausių *Agile* valdymo metodų – *Scrum*. Pagal *Scrum*, projektas vystomas 1–4 savaičių trukmės iteracijomis, kurių metu sukuriamas ir įdiegiamas išbaigtas funkcionalumas. Projekto inicijavimo fazėje, sudaromas būsimų darbų sąrašas. Darbai išdėstomi pagal prioritetus. Pirmiausia įgyvendinami svarbiausi darbai. Projekto apimtis ima aiškėti po pirmųjų iteracijų. Klientas taip pat aktyviai dalyvauja projekto vystyme, išsako savo pageidavimus reikalingiems funkcionalumams pasiekti, kurie vadinami vartotojo istorijomis (angl. *user storys*). Vartotojo istorijos taip pat prioretizuojamos ir pagal tai sudaromas reikalingų darbų sąrašas konkrečiai iteracijai. *Scrum* komanda kiekvieną darbą vertina vartotojo taškais SP (angl. *Story Points*). Susumavus visus vartotojo taškus, gaunamas bendras iteracijos taškų skaičius. Toks apimties skaičiavimo būdas taikomas tik konkrečiai komandai, atskirų komandų skaičiavimai negali būti lyginami. Apimties vertinimas vartotojo taškais yra ypač subjektyvus, rezultatai neatkartojami, tačiau praktikoje taikomas itin dažnai.

Kuriant PĮ pagal *Scrum* metodiką ir planuojant darbus, kyla klausimas kiek darbų galima padaryti per vieną sprintą. Atsakyti į šį klausimą gali padėti planavimo pokeris (angl. *Planning Poker*) – dar viena planavimo metodika. Planavimo pokerio metu, komanda turi įvertinti darbus. Kiekvienas komandos narys savo vertinimus užrašo ant kortelių, kurios yra užverčiamos. Visa komanda vėliau aptaria kiekvieną vertinimą, diskutuoja ir visi kartu priima sprendimus kada ir kokie darbai bus atliekami pirmiausia ir kiek tai gali trukti. Planavimo pokeris giriamas už tai, kad visa komanda įtraukiama į vertinimo procesą, tačiau šiuo metodu gauti apimties įvertinimai negali būti lyginami su kitos komandos panašiais projektais, o rezultatai yra neatkartojami.

1.5.6. Story Points metodikos analizė

Pagal Scrum metodiką, PĮ kūrimas atliekamas sprintais. Sprintas trunka apie 30 dienų ir negali būti pratęstas. Scrum komanda kartu su PĮ užsakovu sudaro reikalingų atlikti darbų sąrašą, kurį vėliau prioretizuoja. Darbų sąrašas arba kitaip scenarijus, yra tam tikri išskirtiniai užsakovo reikalavimai programinės įrangos kūrimo komandai. Darbų sudėtingumui įvertinti naudojami istorijų taškai (angl. *Story Points*), o pats vertinimo procesas vadinamas Story Points metodu. Scrum komanda, pagal priskirtą taškų skaičių kiekvienam darbui, nusprendžia kuri vartotojo istorija bus atliekama konkretaus sprinto metu. Vartotojo istorija gali būti išskaidyta į daugiau nei vieną sprintą. [5]

Story Points dydis išreiškiamas Fibonačio seka [27]. Vertinant konkretų scenarijų, pirmiausia sudaroma vertinimo taškų skalė, kuri dažniausiai prasideda Fibonačio sekos pirmais šešiais skaitmenimis ir baigiasi vienu ar dviem iš esmės didesnių verčių skaičiais. Story Points vertinimo skalės pavyzdys: 1, 2, 3, 5, 8, 13, 20, 50. Kadangi skalė sudaryta iš nedidelio ir aiškaus skaičių rinkinio, kiekvieno scenarijaus vertinimas vyksta greitai ir užtikrintai. Jei scenarijui priskiriama 50 ar 20 taškų, valdyti ir kontroliuoti jį vienoje iteracijoje būtų gana sudėtinga, todėl rekomenduojama tokį scenarijų skaidyti į mažesnius ir lengviau valdomus [23].

Vertinimo eiga, taikant SP metodiką, vyksta taip:

- vertintojas pasirenka dvi užduotis;
- atsižvelgiant į užduočių sudėtingumą, joms priskiriami SP įverčiai; šios užduotys tampa pagrindu tolimesnių užduočių vertinimui;
- visos kitos projekto užduotys vertinamos lyginant jas su pirmomis užduotimis.

Šaltinyje [28] teigiama, kad tokių metodų, kaip UCP, taikymui reikia skirti 10-20% viso projekto pastangų. Tuo tarpu SP metodikos taikymui reikia investuoti daug mažiau laiko, o komanda visą laiką gali skirti PĮ kūrimui.

Vertinimo rezultatai, taikant Story Points metodiką, yra subjektyvūs. Siekiant sumažinti rezultatų subjektyvumą, kiekvienai vertinimo skalės vertei yra nustatomi kriterijai. Nustatant kriterijus, dalyvauja visi vertinimo komandos nariai. Kriterijų nustatymas vyksta remiantis istoriniais projektų duomenimis. Komanda nusprendžia ir susitaria kokiais atvejais būtų naudojamas konkretus taškų skaičius, savo susitarimą užfiksuoja. Svarbu, kad kiekvienas, esantis komandoje, suprastų skalėje esančių reikšmių svorį. Taip sumažinamas rezultatų subjektyvumas, tačiau visiškai jo išvengti nepavyksta.

1.5.7. Programinės įrangos apimties skaičiavimo metodų analizės apibendrinimas

Norint atlikti tikslų ir kokybišką PĮ apimties vertinimą, pirmiausia reikia suprasti kaip, kada ir kokį metodą taikyti. Gausu literatūros apie metodų tobulinimą, modifikavimą, kūrimą naujų. Tokiu būdu gauname dar didesnį skaičių metodų, modelių, įrankių, kuriuos galima pritaikyti skaičiuojant PĮ

apimtį, tačiau tai tik dar labiau apsunkina pasirinkimo momentą – kokį metodą pasirinkti vienam ar kitam projektui įvertinti.

Sprendimo įgyvendinimui pasirinkti metodai: COSMIC, UCP, story points (1.9 lentelė). Iš jų vienintelis COSMIC yra standartizuotas, taigi turi detalizuotą metodinę literatūrą. Nepaisant to, šaltiniuose minima, jog trūksta aiškaus ir struktūrizuoto vadovo ar taisyklių rinkinio, kad metodas būtų lengviau išmokstamas. Use Case Points ir Story Points nėra standartizuoti metodai, taigi aiškaus taisyklių rinkinio, kaip ir kada juos taikyti, taip pat trūksta.

1.9 lentelė. Sprendimo įgyvendinimui pasirinkti metodai

Palyginimo kriterijus	**COSMC**	**Use Case Points**	**Story Points**
Matavimo vnt.	CFP	UCP	SP
Matavimo vnt. žymi sudėtingumą	ne	ne	taip
Standartas	ISO	-	-
Matavimo metodas	Iš viršaus į apačią	Iš viršaus į apačią	Iš viršaus į apačią
Tinkamumas ankstyvose projekto stadijose	Taip, vėliau nei UCP	taip	taip
Taikymo sritis	verslo, realaus laiko, infrastruktūros IS; Agile projektai	objektiškai orientuotos sistemos; Agile projektai	Agile projektai (Scrum metodika)
Duomenų šaltinis skaičiavimams	FNR dokumentas	PA specifikacija	Vartotojų istorijos
Matavimo atributai	IS funkciniai procesai: Įėjimai Išėjimai Skaitymas Rašymas	IS aktoriai ir PA	Vartotojų istorijos
Pastangos metodui įsisavinti ir taikyti	76% daugiau nei UCP	10% projekto pastangų	Mažos
Kada geriausia naudoti	Detaliam, išsamiam vertinimui	Tiek detaliam, tiek abstrakčiam vertinimams	Greitam įvertinimui; Tik toje pačioje komandoje
Rezultatų pakartojamumas	taip	taip, iš dalies	ne

1.6. Tyrimo objektų naudotojų analizė.

Asmuo, atsakingas už projekto valdymą, yra projekto vadovas, žmonės, atliekantys projekto darbus, yra projekto komanda. Komandą gali sudaryti tokie specialistai: analitikas, projektuotojas, programuotojas, testuotojas ir kt.

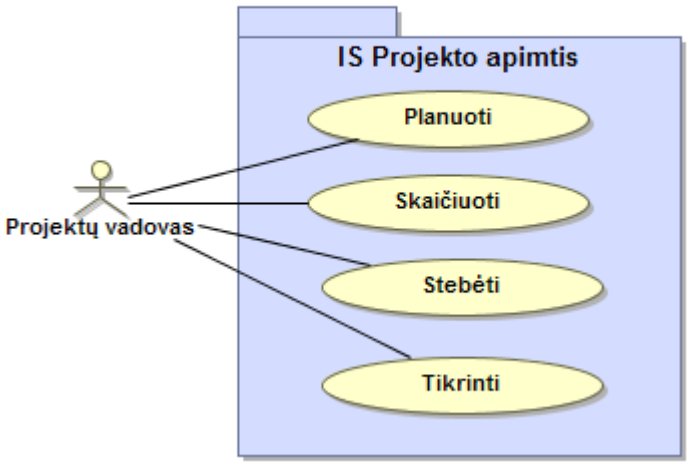
Projektų vadovas turi užtikrinti, kad būtų suplanuoti ir vykdomi visi, tik konkretaus projekto tikslams pasiekti reikalingi, darbai. Dėl galimos darbų trukmės projektų vadovas tariasi su savo komanda. Komandos nariai, kiekvienas pagal savo specializaciją, padeda įvertinti projekto darbų apimtį.

Kalbant apie IS projektų apimties valdymą, projektų vadovas atlieka tokius procesus:

- planuoja projekto apimtį;

- skaičiuoja projekto apimtį;
- stebi projekto apimtį;
- tikrina (verifikuoja) projekto apimtį.

Anksčiau paminėti IS projektų apimties valdymo punktai pavaizduoti PA diagramoje 4 pav.



4 pav. IS projekto apimties valdymas

Tam, kad projekto apimtis būtų įvertinta teisingai, labai svarbu pasirinkti tinkamą būdą. Tobulėjant PĮ ir augant jų poreikiui, projektų apimties valdymas tapo iššūkiu projektų vadovams. Sukurti įvairiausi apimties pamatavimo būdai, modeliai, metodai, tačiau nėra to „vienintelio“, tinkamo visiems IS projektams. Nemažai projektų vadovų, skaičiuodami apimtį, remiasi savo patirtimi, lygina panašių projektų istorinius duomenis. Tokie skaičiavimai nėra patikimi. Tikslesnis apimties įvertinimas pasiekiamas skaičiavimams naudojant konstruktyvų, o geriausiai, kelis konstruktyvius metodus. Problema – kaip žinoti, kada ir kokį metodą naudoti. Didelės kompanijos, valdančios IT projektus, dažnai samdo ekspertus arba taiko kokį nors standartizuotą apimties įvertinimo metodą. Tačiau vidutinėse ir mažose IT įmonėse dažnu atveju nėra taikomas joks metodas, pasikliaujama tik savo ir kolegų patirtimi. Apklausus keletą mažose įmonėse dirbančių projektų vadovų, paaiškėjo, kad jie nėra susipažinę su IS apimties įvertinimo metodais ir nežino, kada ir kokį metodą naudoti. Konkretaus projekto atveju, pasirinkus netinkamą apimties vertinimo metodą, gauti rezultatai, dažnu atveju, bus iškraipyti, netikslūs. Dėl to gali būti sunku įgyvendinti projekto tikslus, blogiausiu atveju, projektas gali būti pasmerktas žlugti. Nesėkmės atveju, projektų vadovas skaičiavimams naudotą metodą įvertins netinkamu ir daugiau jo nenaudos. Taip pat tokia nesėkmė gali išprovokuoti nepasitikėjimą ir visais kitais metodais, skirtais IS projektų apimčiai vertinti. Suformuluotos aiškios taisyklės ir rekomendacijos – kaip, kada ir kokį metodą naudoti – palengvintų darbą projektų vadovams ir visai projekto komandai bei prisidėtų prie įvardintos problemos sprendimo.

Įvertinus tai, kad atsiranda poreikis kurti individualius sprendimus ir informacinės technologijos neišvengiamai tobulėja bei jų poreikis sparčiai auga, galima teigti, jog PĮ kūrimo projektų tik daugės. Vadinasi, vertindami IS projekto apimtį, projektų vadovai vis mažiau galės pasikliauti istoriniais duomenimis ir vis svarbiau bus pasirinkti tinkamą apimties įvertinimo metodą. Kaip matyti iš atliktos analizės, apimties įvertinimo metodų yra tikrai daug, tad logiška, kad tinkamo metodo pasirinkimas kiekvienu atveju turėtų atimti nemažai laiko. Siekiant efektyvinti IS dydžio įvertinimo metodo pasirinkimo procesą, planuojama jį automatizuoti. Tuo tikslu bus kuriama taisyklėmis paremta ekspertinė sistema, kuri vartotojui užduos klausimus, o iš gautų atsakymų sugeneruos rekomendaciją metodo pasirinkimui. Ekspertinei sistemai sukurti bus naudojamas veiklos taisyklių valdymo įrankis.

1.7. Apimties skaičiavimo metodų tiriamųjų darbų apžvalga

Šiuo metu, IS projektų apimties įvertinimo prognozavimui yra sukurta daugybė įvairių metodikų, tad natūralu, kad ir mokslinių tyrimų šia tema atlikta nemažai. Toliau atliekama kitų autorių darbų, tyrinėjančių PĮ projektų dydžio vertinimo metodus, apžvalga.

2016 m. mokslininkų grupė atliko empirinį tyrimą [29], kuriame analizuojamas Funkcinių taškų ir COSMIC metodų taikymas internetinių projektų apimties skaičiavimui. Darbo tikslas – išsiaiškinti ar COSMIC metodo taikymas tokiems projektams yra efektyvesnis, nei funkcinių taškų metodas ir taip pagrįsti perėjimą nuo pirmos kartos metodų taikymo, prie antros kartos metodų. Minėtais metodais gautų rezultatų tikslumo įvertinimui skaičiuotos MdMAR, MMRE, MdMRE, ir Pred(25) reikšmės. Taip pat buvo tiriamas istorinių duomenų, gautų taikant funkcinių taškų metodą, konvertavimas lygčių pagalba į COSMIC funkcinius taškus, kad perėjimo prie naujo metodo taikymo laikotarpiu įmonės turėtų galimybę atlikti esamų ir istorinių rezultatų palyginamąją analizę. Tyrimo išvadose teigiama, kad internetinių projektų apimties skaičiavimams COSMIC metodas yra efektyvesnis ir tikslesnis nei FP metodas. Tačiau turi būti atliekama daugiau tyrimų lygčių, taikytinų duomenų konvertavimui iš FP į CFP, atžvilgiu, kad būtų pasiektas didesnis perskaičiavimo tikslumas.

Rasta ir daugiau atliktų studijų [11], [30], kuriose analizuojamas funkcinės apimties įvertinimo metodų tinkamumas internetinių projektų apimties vertinimui. Studijų rezultatai tikrai geri, patvirtinantys šių metodų ir ypač COSMIC metodo tinkamumą tokių projektų apimčiai vertinti.

Kitame tyrime [31] analizuojami tokie apimties įvertinimo metodai, kaip FP, Mark II, COSMIC, NESMA, UCP. Buvo vertinama IS projektų apimtis, skirtingose projektų gyvavimo fazėse, o gauti rezultatai lyginami su realiais. Rezultatų analizei ir metodų tikslumo bei patikimumo vertinimui buvo skaičiuojama koreliacija tarp išmatuoto apimties dydžio bei faktinio, MRE, MMRE, MREmax. Šio darbo autoriai teigia, kad rezultatų tikslumo vertinimas, naudojant tik MMRE metriką yra kritikuojami. Tyrimo rezultatai parodė, kad funkcinės apimties matavimo metodai pasižymi geru tikslumu. Išimtis – NESMA metodas, kurio rezultatai prasčiausi. UCP metodo tikslumas taip pat

nebuvo gerai įvertintas, tačiau tyrimo autorių nuomone, tai susiję su netiksliais panaudojimo atvejais. Geriausiai šiame tyrime pasirodė COSMIC metodas. Be to, šiame darbe dėmesys atkreipiamas ir į tai, kad tiksliam apimties vertinimui visiškai užtenka funkcinių parametrų ir nebūtina vertinti nefunkcinių kaip, kad technologijų sudėtingumo. Ši studija rodo itin gerą funkcinės apimties metodų rezultatų koreliaciją su realiais duomenimis.

Sweta Kurami savo darbe [32] PĮ projektų dydžiui prognozuoti siūlo naudoti atraminių vektorių regresiją SVR (angl. *support vector regression).* Eksperimento metu gauti rezultatai lyginami su patikslintos COCOMO (angl. *Intermediate*) ir MOPSO (angl. *Multiple Objective Practicle Swarm Optimization*) metodikų rezultatais. Metodų tikslumui įvertinti, apskaičiuoti MARE ir PRED(25). Rezultatai gauti naudojant SVR yra geresni, nei darbe lygintų kitų metodikų.

2014 m. grupė mokslininkų publikavo darbą [33], kuriame pateikė savo sukurtą metodiką, skirtą IS projektų apimties įvertinimui, kaip alternatyvą skaičiavimams, paremtiems analogija. Pagrindinė idėja yra ta, kad istorinius projektus reikia skirstyti į tam tikras grupes. Vietoj naujo projekto lyginimo su visais turimais duomenimis, pagal mokslininkų siūlomą metodiką, palyginimas atliekamas tik su projektais, esančiais tam tikrose grupėse. Siūloma metodika išbandyta su 505 realių projektų duomenimis. Metodikos tikslumui įvertinti buvo skaičiuojami tokie rodikliai: RE, MRE, MMRE, PRED. Atlikus eksperimentą, buvo įrodyta, kad siūloma metodika gali padidinti naujų projektų apimties įvertinimo tikslumą. Taip pat, gauti rezultatai parodė, kad siūlomas metodas yra pakankamai lankstus ir jį galima taikyti įvairiems PĮ projektams.

Miroslawa‘as Ochodek‘as drauge su kitais mokslininkais 2011 m. pristatė darbą [18], kuriame itin detaliai analizuojamas UCP metodas. Tyrimas atliktas su 14 skirtingų PĮ projektų duomenimis. Tyrimo metu siekiama išsiaiškinti galimybes supaprastinti UCP metodą, neprarandant rezultatų tikslumo. Vienas iš nagrinėtų UCP metodo supaprastinimo būdų, buvo UAW eliminavimas iš skaičiavimų. Taip pat buvo vertinama techninių ir aplinkos faktorių įtaka PĮ apimties įvertinimui. Rezultatų, gautų įvairiomis UCP metodo variacijomis, tikslumas analizuojamas skaičiuojant ir lyginant MRE, MMRE ir PRED(25) rodiklius. MMRE pasiskirstymai pateikiami stačiakampėse diagramose. Mokslininkų išvadose teigiama, kad modifikuoti UCP metodai generuoja tikslius skaičiavimų rezultatus, tačiau taip pat teigiama, kad dar reikėtų atlikti daugiau eksperimentinių skaičiavimų su didesniu kiekiu projektų duomenų.

2014 m. straipsnio „*An Early Software Effort Estimation Method Based on Use Cases and Conceptual Classes*“ autoriai pasiūlė metodą, skirtą PĮ apimties prognozavimui ankstyvose projekto stadijose, paremtą konceptualių klasių modeliu [25]. Metodas buvo išbandytas su 14 realių projektų duomenimis, o gauti rezultatai palyginti su faktinėmis projektų trukmėmis, bei UCP metodu gautais skaičiavimais. Siūlomos metodikos tikslumui įvertinti skaičiuoti rodikliai: MRE, PRED(25), AMSE

(pakoreguota vidutinė kvadratinė paklaida). Tyrimo metu pastebėta didelė koreliacija tarp konceptualių klasių skaičiaus ir projekto apimties dydžio.

Kitų mokslininkų darbų apžvalgos apibendrinimas pateikiamas 7.1 priede. Atlikus kitų tiriamųjų darbų analizę, galima daryti išvadą, kad vieno tinkamo metodo įvertinti PĮ dydžiui nėra, kaip ir nėra vieningų taisyklių ar vieno rodiklio apimties prognozavimo rezultatams įvertinti ir metodo tikslumui nustatyti. Dažniausiai metodikos tikslumas vertinamas skaičiuojant ir lyginant su faktiniais rezultatais MRE, MMRE, PRED(25) rodiklius.

1.8. Siekiamo sprendimo apibrėžimas

Darbo eigoje kuriamas klausimų-atsakymų bei rekomendacijų modelis, kurį būtų galima pritaikyti planuojant programinės įrangos kūrimo projektus. Pasiūlytas sprendimas padėtų lengviau, efektyviau ir greičiau pasirinkti tinkamą metodą ar metodus. Sprendimui pasiekti, projektuojama ekspertinė sistema, kuri, uždavusi klausimus vartotojui, iš gautų atsakymų sugeneruos rekomendaciją konkrečiu atveju tinkamiausiam metodui pasirinkti. Ekspertinė sistema – tai dialogo režimu funkcionuojanti programa, skirta palengvinti sprendimų priėmimą sudėtingose situacijose. Nuo įprastų sistemų ji skiriasi tuo, kad operuoja žiniomis, o informacinė sistema – duomenimis. Kuriant ekspertines sistemas naudojamos tam tikros dalykinės srities specialistų-ekspertų žinios. Įprasta, jog žinios greitai sensta ir jas reikia nuolat atnaujinti, taigi ekspertinė sistema taip pat turi būti nuolat atnaujinama papildant žinių bazę naujais faktais ir taisyklėmis ar panaikinant pasenusias. Siekiama, jog šiame darbe projektuojamos ekspertinės sistemos žinių bazės modulio atnaujinimas būtų kuo paprastesnis.

2. SPRENDIMO REIKALAVIMŲ SPECIFIKACIJA IR PROJEKTAS

Dauguma IT kompanijų, kuriamų IS apimties įvertinimui, vis dar nenaudoja formalizuotų skaičiavimo metodų, o dažniau remiasi savo patirtimi ar statistine informacija apie panašių projektų apimtį. Tokie spėjamojo pobūdžio skaičiavimai yra labai netikslūs, o įmonė vadovaudamasi tik tokiais įverčiais rizikuoja patirti didelius nuostolius. Žiūrint į ateities perspektyvą, tikėtina, jog programinė įranga taps vis labiau sudėtingesnė, o kiekvienas IT projektas bus vis labiau unikalus, taigi mažės galimybė skaičiavimuose pasinaudoti panašių projektų duomenimis. Patikimiausi rezultatai gaunami taikant formalius ir daugeliui suprantamus metodus. Yra žinoma daug IS apimties skaičiavimo metodų, jų modifikacijų, šių metodų pagrindu sukurtų įrankių. Žmogui, kuris niekada nesidomėjo formaliais kuriamos PĮ apimties įvertinimo metodais, būtų labai sunku išsirinkti tinkamą ir tektų sugaišti daug laiko įvairių metodikų studijavimui. Siekiant palengvinti ir paspartinti metodikos pasirinkimo procesą, bus projektuojama taisyklėmis paremta ekspertinė sistema.

2.1. Potencialūs sistemos naudotojai

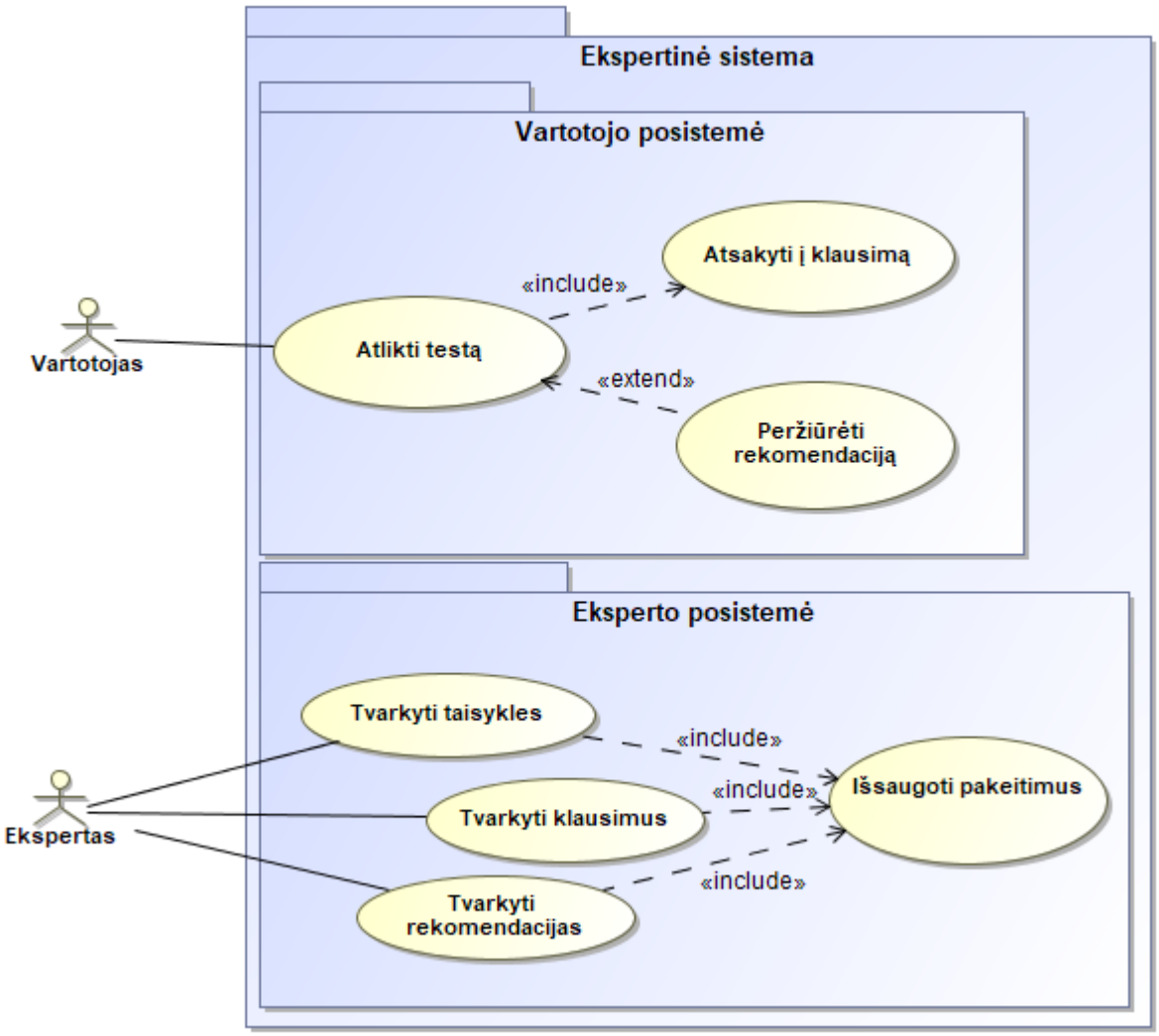
Pagrindiniai projektuojamos sistemos vartotojai – IT srities specialistai, turintys IT srities išsilavinimą arba darbo su IT projektais patirties. Sistemos vartotojai neprivalo turėti programavimo žinių, jų sprendžiami uždaviniai pateikti lentelėje (2.1 lentelė).

2.1 lentelė. Projektuojamos ekspertinės sistemos naudotojai

Naudotojai	Sprendžiami uždaviniai	Patirtis dalykinėje srityte	Patirtis IT
Vartotojas	Renkasi PĮ apimties įvertinimo metodą ir pritaiko jį praktikoje.	Srities specialistas	Patyręs
Ekspertas	Administruoja žinių bazę: kuria, redaguoja, trina taisykles, klausimus, rekomendacijas.	Srities ekspertas	Patyręs

2.2. Reikalavimų specifikacija

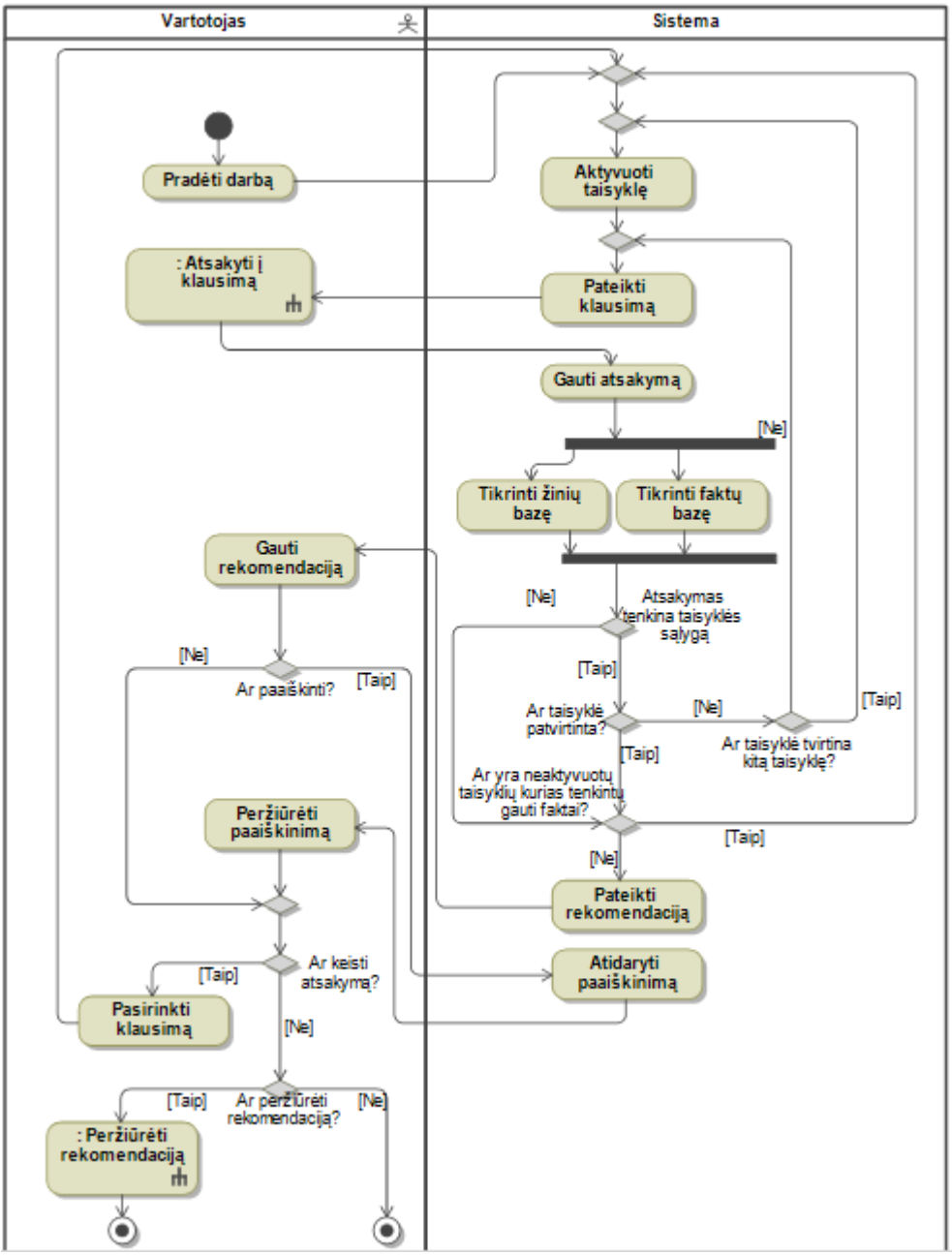
Remiantis dalykinės srities analizės duomenimis, sudarytas kompiuterizuojamos veiklos panaudojimo atvejų modelis. Šia sistema gali naudotis projektų vadovas-ekspertas ir paprastas vartotojas-dalykinės srities specialistas, norintis atlikti klausimų testą ir gauti tinkamo metodo rekomendaciją. Projektų vadovas-ekspertas yra tas žmogus, kuris turi didelę profesinę patirtį ir žinių apie dalykinę sritį bei galintis papildyti turimą žinių bazę naujomis taisyklėmis, faktais ir išvadomis. Vartotojo ir projektų vadovo-eksperto atliekamos funkcijos su sistema pavaizduotos panaudojimo atvejų diagramoje 5 pav.



5 pav. Sistemos panaudojimo atvejų diagrama

Vartotojas, dirbdamas su sistema, atlieka testą. Panaudojimo atvejo „Atlikti testą“ veiklos diagrama pateikiama 6 pav.

Vartotojas, pradėjęs darbą su sistema, gauna klausimą į kurį pateikia atsakymą. Sistema, sprendimų išvedimo mechanizmo pagalba, atsakymą analizuoja ir, jei reikia, pateikia kitą klausimą. Šis ciklas kartojasi, kol sistema pateikia rekomendaciją. Taip pat vartotojas turi galimybę peržiūrėti paaiškinimą, kodėl jam pateikta tokia rekomendacija. Jei reikia, vartotojas gali grįžti į bet kurį klausimą ir pateikti atsakymus iš naujo.

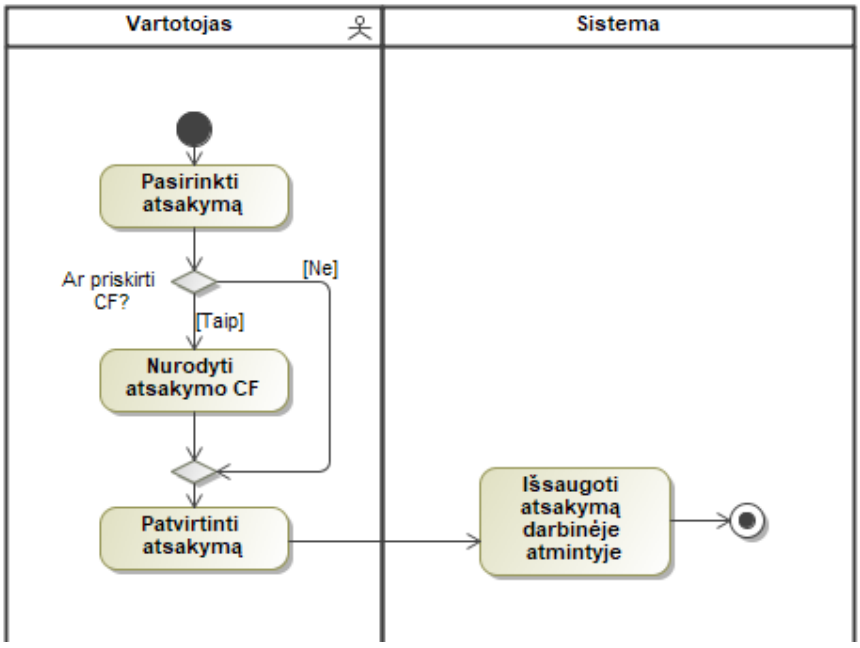


6 pav. Veiklos diagrama „Atlikti testą“

2.2 lentelė. PA „Atlikti testą“ formalus aprašymas

PA „Atlikti testą“		
Prieš sąlyga	Vartotojas turi ekspertinę sistemą rekomenduojančią metodą PĮ apimčiai vertinti	
Sužadinimo sąlyga	Vartotojas nori gauti rekomendaciją	
Susiję panaudojimo atvejai	Išplečia PA	
	Apima PA	Atsakyti į klausimą
	Specializuoja PA	
Pagrindinis įvykių srautas	**Sistemos reakcija ir sprendimai**	
1. Vartotojas spaudžia mygtuką „Pradėti“	Sistema aktyvuoja taisyklę; Sistema pateikia klausimą.	
2. Vartotojas atsako į klausimą	Sistema gauna atsakymą; Sistema atlieka faktų ir žinių bazės tikrinimą; Sistema pateikia rekomendaciją.	
Po sąlyga	Pateikta rekomendacija	
Alternatyvūs scenarijai		
A1. Jei atsakymas netenkina taisyklės sąlygos	Sistema ieško neaktyvuotų taisyklių, kurių sąlygas tenkina gauti faktai; Sistema randa taisyklę; Sistema pateikia klausimą.	
A2. Jei taisyklė tvirtina kitą taisyklę	Sistema aktyvuoja taisyklę; Sistema pateikia klausimą	
A3. Jei yra neaktyvuotų taisyklių, kurias tenkina gauti faktai	Sistema aktyvuoja taisyklę; Sistema pateikia klausimą.	
A4. Jei vartotojas pasirenka paaiškinimo funkciją	Sistema pateikia paaiškinimą, kodėl pateikė tokią rekomendaciją.	
A5. Jei vartotojas pasirenka grįžti į atsakytą klausimą	Sistema aktyvuoja taisyklę; Sistema pateikia klausimą.	
A6. Jei vartotojas pasirenka peržiūrėti rekomendaciją	Sistema atidaro rekomendacijos aprašymą.	

Panaudojimo atvejis „Atlikti testą“ susideda iš jį apimančio panaudojimo atvejo „Atsakyti į klausimą“, kurio veiklos diagrama pateikiama 7 pav.



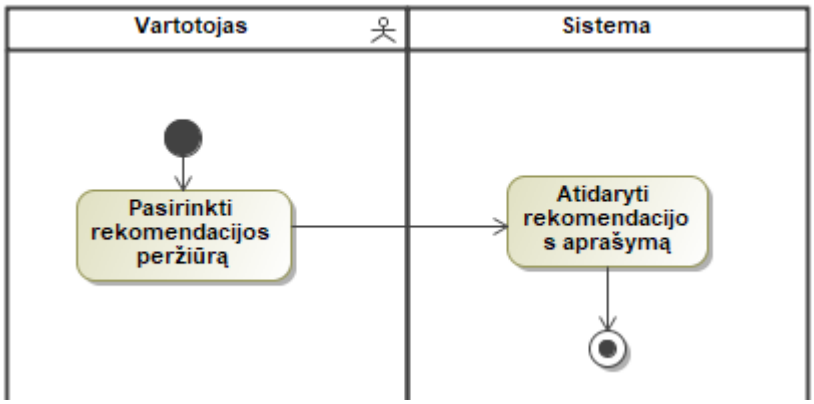
7 pav. Veiklos diagrama „Atsakyti į klausimą“

Formalus PA „Atsakyti į klausimą“ aprašymas pateikiamas lentelėje (2.2 lentelė).

2.3 lentelė. PA „Atsakyti į klausimą“ formalus aprašymas

PA „Atsakyti į klausimą“		
Prieš sąlyga	Vartotojas ekrane mato pateiktą klausimą	
Sužadinimo sąlyga	Sistema nori pateikti rekomendaciją	
Susiję panaudojimo atvejai	Išplečia PA	
	Apima PA	
	Specializuoja PA	
Pagrindinis įvykių srautas	**Sistemos reakcija ir sprendimai**	
1. Vartotojas pasirenka atsakymą		
2. Vartotojas pasirenka atsakymo CF		
3. Vartotojas patvirtina atsakymo CF	Sistema išsaugo atsakymą darbinėje atmintyje	
Po sąlyga	Sistemos darbinėje atmintyje išsaugoti duomenys	
Alternatyvūs scenarijai		
A1. Jei vartotojas nepriskiria atsakymui CF	Sistema priskiria numatytąją CF reikšmę.	

Vartotojas, gavęs rekomendaciją, gali peržiūrėti detalų jos aprašymą. Panaudojimo atvejo „Peržiūrėti rekomendaciją“ veiklos diagrama pateikta 8 pav.

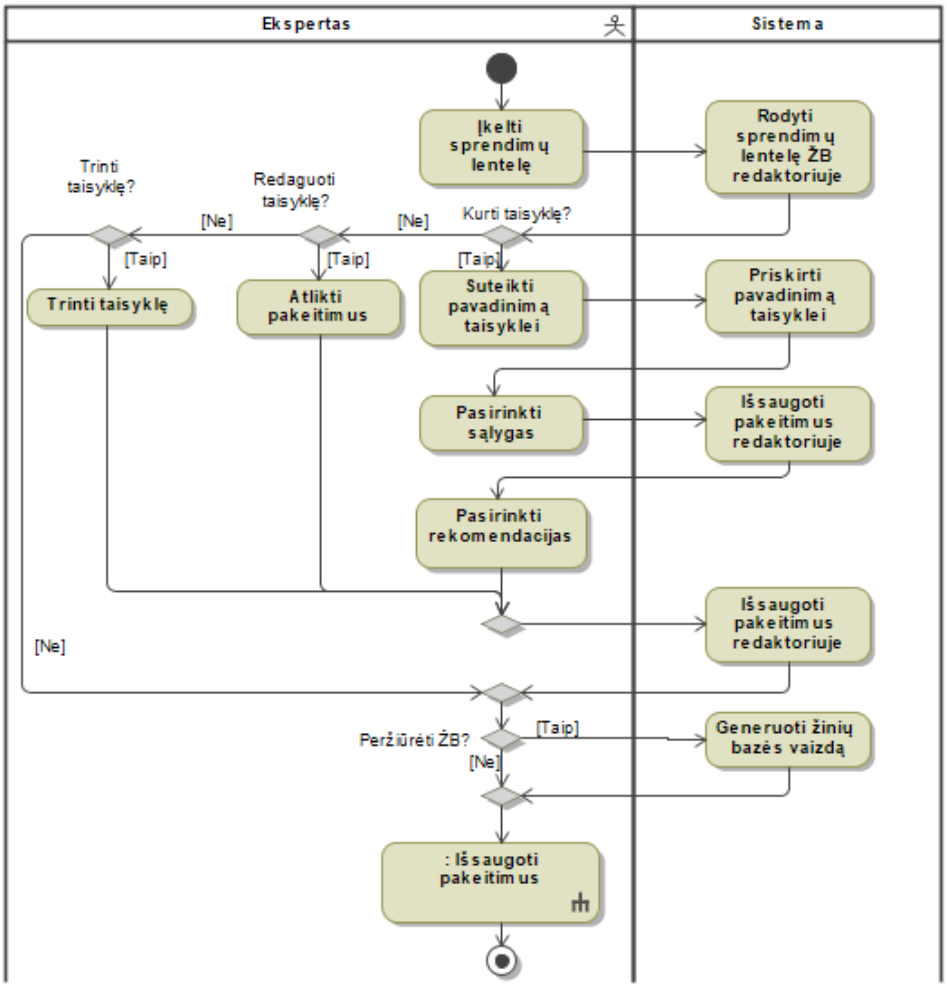


8 pav. Veiklos diagrama „Peržiūrėti rekomendaciją“

Formalus panaudojimo atvejo „Peržiūrėti rekomendaciją“ aprašymas pateikiamas 2.4 lentelėje.

2.4 lentelė. PA „Peržiūrėti rekomendaciją“ formalus aprašymas

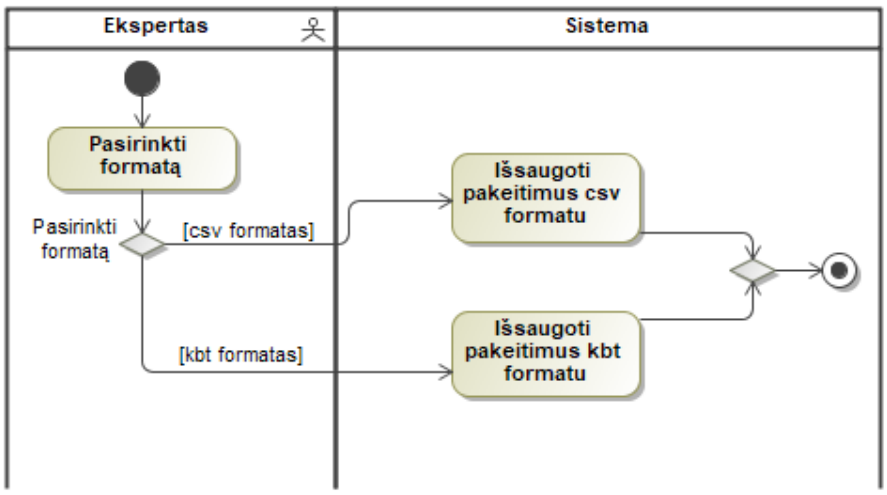
PA „Atsakyti į klausimą“		
Prieš sąlyga	Vartotojas gavo rekomendaciją	
Sužadinimo sąlyga	Vartotojas nori peržiūrėti rekomendaciją	
Susiję panaudojimo atvejai	Išplečia PA	
	Apima PA	
	Specializuoja PA	
Pagrindinis įvykių srautas	**Sistemos reakcija ir sprendimai**	
1. Vartotojas pasirenka peržiūrėti rekomendaciją	Sistema pateikia rekomendacijos aprašymą	
Po sąlyga	Atidarytas rekomendacijos aprašymas	
Alternatyvūs scenarijai		



9 pav. Veiklos diagrama „Tvarkyti taisykles“

2.5 lentelė. PA „Tvarkyti taisykles“ formalus aprašymas

PA „Tvarkyti taisykles“		
Prieš sąlyga	Vartotojas turi žinių bazės failą *kbt* formatu. Vartotojas atsidaręs žinių bazės redaktorių.	
Sužadinimo sąlyga	Vartotojas nori tvarkyti taisykles	
Susiję panaudojimo atvejai	Išplečia PA	
	Apima PA	Išsaugoti pakeitimus
	Specializuoja PA	
Pagrindinis įvykių srautas	**Sistemos reakcija ir sprendimai**	
1. Vartotojas įkelia sprendimų lentelės duomenis.	Sistema rodo žinių bazę redaktoriaus ekrane.	
2. Vartotojas kuria taisyklę, suteikdamas jai pavadinimą.	Priskiria pavadinimą taisyklei	
3. Vartotojas pasirenka sąlygas	Sistema išsaugo pakeitimus redaktoriuje	
4. Vartotojas pasirenka rekomendacijas	Sistema išsaugo pakeitimus redaktoriuje	
5. Vartotojas išsaugo pakeitimus	Sistema išsaugo pakeitimus	
Po sąlyga	Pakeistas žinių bazės failas	
Alternatyvūs scenarijai		
A1. Jei vartotojas redaguoja taisyklę	Sistema išsaugo pakeitimus redaktoriuje	
A2. Jei vartotojas trina taisyklę	Sistema išsaugo pakeitimus redaktoriuje	
A3. Jei vartotojas nori peržiūrėti žinių bazės failą	Sistema generuoja struktūrinį žinių bazės vaizdą	



10 pav. Veiklos diagrama „Išsaugoti pakeitimus“

2.6 lentelė. PA „Išsaugoti pakeitimus“ formalus aprašymas

PA „Tvarkyti taisykles“		
Prieš sąlyga	Vartotojas atliko pakeitimus taisyklėms	
Sužadinimo sąlyga	Vartotojas nori išsaugoti pakeitimus	
Susiję panaudojimo atvejai	Išplečia PA	
	Apima PA	
	Specializuoja PA	
Pagrindinis įvykių srautas	**Sistemos reakcija ir sprendimai**	
1. Vartotojas pasirenka formatą.	Sistema išsaugo pakeitimus pasirinktu formatu.	
Po sąlyga	Išsaugotas žinių bazės failas	

Panaudojimo atvejai „Tvarkyti rekomendacijas“, „Tvarkyti klausimus“ vyksta tuo pačiu principu kaip ir „Tvarkyti taisykles“, todėl jiems nebus braižomos atskiros veiklos diagramos ir daromi formalūs aprašymai.

2.3. Nefunkciniai reikalavimai

Nefunkciniai reikalavimai sistemai pateikiami lentelėje (2.7 lentelė).

2.7 lentelė. Nefunkciniai reikalavimai sistema

Nefunkciniai reikalavimai	**Paaiškinimas**	**Tinkamumo kriterijus**
Panaudojamumui	Sistema turi būti pakankamai suprantama, kuria būtų galima naudotis be papildomų mokymų ar aprašymo	Vartotojas per 15 minučių perpranta sistemą ir gali ja naudotis be papildomų mokymų.
Sistemos išvaizdai	Vartotojo sąsajos kalba - lietuvių.	Vartotojo sąsajos kalba - lietuvių.
Patikimumas	Turi būti galimybė grįžti atgal į bet kurį klausimą.	Yra grįžimo atgal galimybė.

2.4. Reikalingi duomenys sprendimo įgyvendinimui

Remiantis kitų mokslininkų atliktais tyrimais ir eksperimentais, nustatyti kriterijai, kurie daro įtaką renkantis apimties įvertinimo metodą. Nustatyti kriterijai pateikti 2.8 lentelėje. Jie bus naudojami pildant ekspertinės sistemos žinių bazę: formuojant klausimus ir sudarant taisykles. Rekomendacijų rinkinys pateikiamas 2.9 lentelėje.

2.8 lentelė. Pradiniai duomenys taisyklių sudarymui.

Trumpinys	**Aprašymas**
Stand	Kai kurie projektai privalo būti vykdomi laikantis standarto (pavyzdžiui ISO) reikalavimų. Tokiu atveju ir projekto apimties vertinimui turėtų būti naudojamas standartizuotas metodas. Ekspertinė sistema užduoda klausimą, ar vertinant PĮ dydį privalote taikyti standartizuotą (pvz. ISO, LST) metodą/metodologiją?
P_tipas	Klausiama ar vystomas projektas yra tipinis ar unikalus. Unikalaus projekto vertinimui reiktų skirti daugiau dėmesio. Efektyviam rezultatui pasiekti rekomenduojama taikyti kelis apimties įvertinimo metodus.
Kom_vert	Klausiama ar projekto vystymo komandos nariai (ar dalis jų) dalyvaus projekto apimties vertinimo procese. Kai kurių metodų taikymas praktikoje reikalauja, jog vertinimo procese dalyvautų visa komanda.
Vert_L	Klausiama ar vertinimo procesas yra skubus ir abstraktus, ar kruopštus ir detalus. Šio darbo kontekste, laikas, skiriamas vertinimo procesui, apima metodo įsisavinimą ir taikymą. Klausimas *kiek laiko galite skirti projekto apimties įvertinimo procesui* yra itin subjektyvus ir įvairiai interpretuojamas. Nepaisant to, jei vertinimą skirtingais metodais atlieka tas pats žmogus, susipažinęs ar nesusipažinęs su šiais metodais, galima nustatyti, kad vienu atveju vertinimo procesas užtruks ilgiau nei kitu. Remiantis [34], Use Case Points metodo taikymas užima 76% mažiau laiko nei COSMIC.
R_det	Siekiama išsiaiškinti, kurioje projekto fazėje atliekamas vertinimo procesas. Kuo ankstyvesnė projekto fazė, tuo mažiau reikalavimų žinoma ir jie yra aukšto lygio. Remiantis [31], projekto pradžios fazėje (angl. *Inception*) arba ankstyvoje paruošimo (angl. *Elaboration*) fazėje, kuomet nustatomos sistemos ribos, sritis, išskiriamos svarbiausios užduotys (*use case*), išskiriami tik labai aukšto lygio reikalavimai. Tokiu atveju, projekto apimties įvertinimui, rekomenduojama naudoti UCP metodą [31]. Vėlyvesnėje paruošimo fazėje, kuomet išskiriama daugiau kaip 80 procentų reikalavimų, jau galimas ir COSMIC metodo taikymas. Ruošiant apimties įvertinimo metodo parinkimo modelį, klausimas apie projekto gyvavimo fazę nebus formuluojamas. Užtenka atsižvelgti į kokio lygio ir kiek reikalavimų kuriamai sistemai yra išskirta.
Dydis	Remiantis 1.4 skyriuje atlikta analize, formuluojamas klausimas, kuriai kategorijai pagal dydį būtų galima priskirti vertinamą projektą.
Ar_PA	Klausiama ar reikalavimų specifikavimui buvo taikoma panaudojimo atvejų metodika.
Sist_tipas	Klausiama ar sistema skirta dirbti su įvesties, saugojimo, atkūrimo bei išvesties duomenimis. Ne visi metodai tinkami PĮ, skirtų manipuliuoti duomenimis, vertinimui.
Komanda	Vystant projektą *agile* metodika, svarbų vaidmenį vaidina projekto komanda. Jei komandos nariai surinkti tik šiam projektui, anksčiau kartu nedirbę, tai geriau apimties vertinimo procese nenaudoti subjektyvių metodų. Šiuo atveju tikslingiau rinktis objektyvesnius metodus, nors tokių metodų taikymas atima daugiau laiko.

Trumpinys	Aprašymas
Vert_A	Vystant projektą *agile* metodika, svarbu apsibrėžti vertinimo proceso apimties ribas, t.y. nustatoma ar apimtis skaičiuojama vienai iteracijai, ar – visam projektui.
Rez_Pak	Taikant vienus apimties vertinimo metodus, rezultatai būna subjektyvūs, neatkartojami bei nesulyginami su kitų projektų rezultatais. Jei yra poreikis vertinimo procesą atlikti objektyviai, reikia pasirinkti tam tinkamą metodiką.

2.9 lentelė. Rekomendacijų rinkinys

Eil Nr.	Rekomendacijos pavadinimas
1	R1. Rekomenduojama naudoti COSMIC
2	R2 Rekomenduojama naudoti COSMIC aprox
3	R3 Rekomenduojama naudoti COSMIC ir COSMIC aprox
4	R4 Rekomenduojama naudoti COSMIC aprox ir SP
5	R5 Rekomenduojama naudoti COSMIC ir SP
6	R6 Vertinimo atlikti negalima
7	R7 Rekomenduojama naudoti SP
8	R8 Rekomenduojama naudoti UCP
9	R9 Rekomenduojama naudoti COSMIC aprox ir UCP
10	R10 Rekomendacija negalima
11	R11 Rekomenduojama naudoti COSMIC ir UCP

2.5. Žinių atvaizdavimo modelis

Norint efektyviai operuoti realaus pasaulio žiniomis kompiuteryje yra būtinas žinių vaizdavimo modelis. Pagrindiniai žinių vaizdavimo modeliai yra šie :

- loginiai žinių vaizdavimo modeliai;
- produkcinės taisyklės;
- freimai;
- semantiniai tinklai;
- mišrieji modeliai.

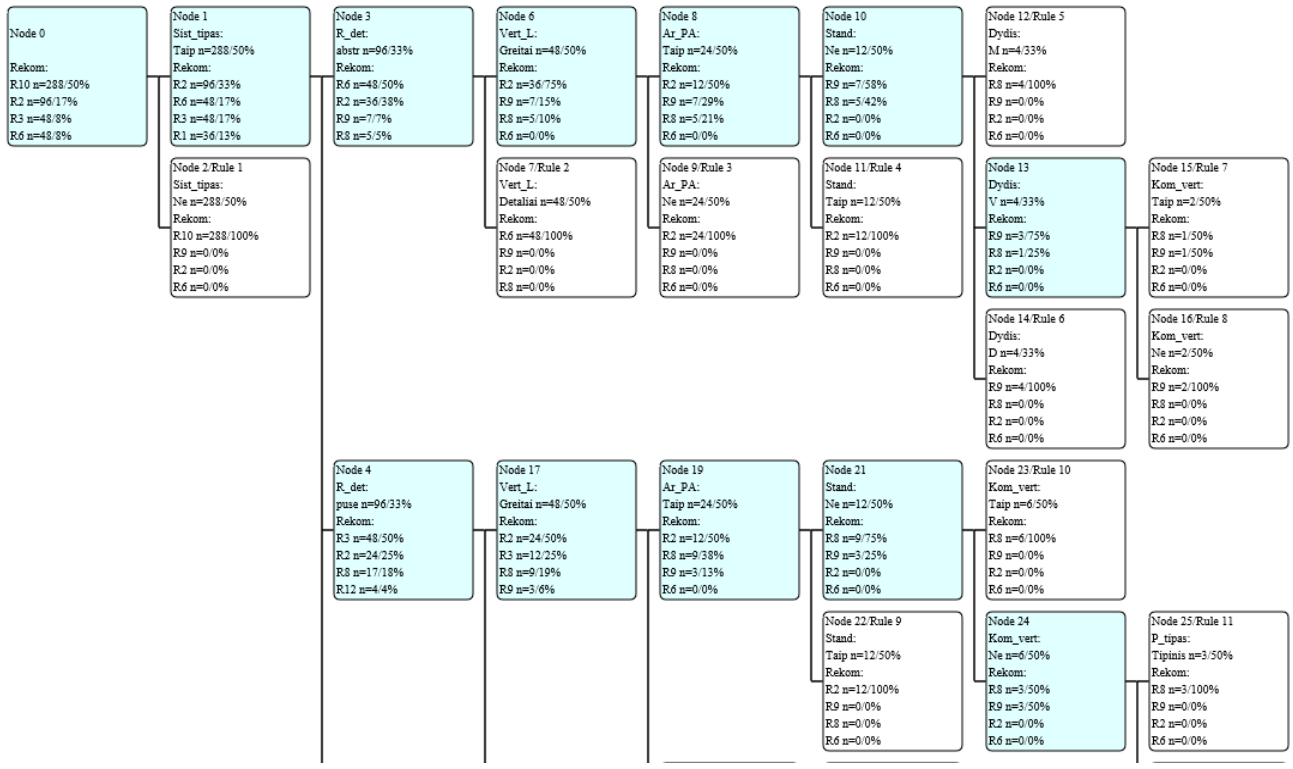
Remiantis kitų ekspertų patirtimi, iš 2.8 lentelėje pateiktų kriterijų buvo sudėliotos galimos apimties įvertinimo proceso situacijos su rekomendacijomis metodo pasirinkimui. Iš šių struktūruotų duomenų aibės buvo formuojamas sprendimų medis. Gautų duomenų klasifikavimo uždaviniui spręsti pasirinkta formuoti sprendimų medį (angl. *decision tree*), kuris vėliau naudojamas produkcinių taisyklių formavimui. Sprendimų medžio klasifikavimo esmės yra kaip padalinti duomenų aibės erdvę į stačiakampes sritis. Duomens suklasifikavimas priklauso nuo to, į kurią sritį jis paklius. Sprendimų medžio formavimui naudotas ID3 algoritmas. Šio algoritmo esmė sukurti sprendimų medį taip, kad

būtų iki minimumo sumažintas tikėtinas palyginimų skaičius. Pagrindinis ID3 algoritmo tikslas yra paklausti tokius klausimus, kurių atsakymai teikia daugiausiai informacijos. Remiantis šiuo algoritmu, klausimų eiliškumas dėliojamas taip: kurio klausimo informacijos nauda aukščiausia, tas klausimas atsidurs sprendimų medžio viršūnėje. Informacijos dydžiui išmatuoti skaičiuojama *entropija*.

Entropija duomenų aibei apskaičiuota ir sprendimų medis suformuotas *e2gRuleInduction* įrankio pagalba. Sprendimų medis turi dvi pagrindines šakas, kurių viršūnėje – klausimas apie naudojamą projekto vystymo metodiką. Ši viršūnė skyla į kitas pagrindines šakas:

- tradicinės metodikos šaka;
- lanksčiosios (angl. *agile*) metodikos šaka.

Sprendimų medžio fragmentas pateikiamas 11 pav. Bendras medžio šakos vaizdas „tradicinei“ projekto valdymo metodikai pateikiamas 7.3 priede, o „agile“ - 7.2 priede.



11 pav. Sprendimų medžio fragmentas

Formuojant sprendimų medį, kiekvienam sprendimo taškui buvo skaičiuojama entropija. Skaičiavimo pavyzdys pateikiamas 12 pav.

```
At node 6 Rekom has: 1 R5, 7 R7, 12 R2, 0 R6, 0 R3, 0 R1, 6 R4 cases
Choices to split node 6:
Rez_pak:                  Gain = 0.556, GainRatio = 0.598
Vert_A:                   Gain = 0.539, GainRatio = 0.542
Komanda:                  Gain = 0.128, GainRatio = 0.129
Dydis:                    Gain = 0.095, GainRatio = 0.060
P_tipas:                  Gain = 0.095, GainRatio = 0.095
Best split attribute is: Rez_pak
Stepwise chosen attribute was: Vert_A
Node 8 visam projektui yields: 1 R5 (8%), 4 R7 (33%), 1 R2 (8%), 0 R6 (0%), 0 R3 (0%), 0 R1 (0%), 6 R4 (50%) Rekom
cases
Node 9 vienai iteracijai yields: 0 R5 (0%), 3 R7 (21%), 11 R2 (78%), 0 R6 (0%), 0 R3 (0%), 0 R1 (0%), 0 R4 (0%) Rekom
cases

At node 8 Rekom has: 1 R5, 4 R7, 1 R2, 0 R6, 0 R3, 0 R1, 6 R4 cases
Choices to split node 8:
P_tipas:                  Gain = 0.541, GainRatio = 0.541
Komanda:                  Gain = 0.405, GainRatio = 0.405
Dydis:                    Gain = 0.355, GainRatio = 0.224
Rez_pak:                  Gain = -0.000, GainRatio = 0.000
Best split attribute is: P_tipas
Stepwise chosen attribute was: P_tipas
Node 10 Unikalus yields: 1 R5 (16%), 0 R7 (0%), 1 R2 (16%), 0 R6 (0%), 0 R3 (0%), 0 R1 (0%), 4 R4 (66%) Rekom cases
Node 11 Tipinis (praeityje jau esate kūrę panašių sistemų) yields: 0 R5 (0%), 4 R7 (66%), 0 R2 (0%), 0 R6 (0%), 0 R3
(0%), 0 R1 (0%), 2 R4 (33%) Rekom cases

At node 10 Rekom has: 1 R5, 0 R7, 1 R2, 0 R6, 0 R3, 0 R1, 4 R4 cases
Choices to split node 10:
Dydis:                    Gain = 0.585, GainRatio = 0.369
Komanda:                  Gain = 0.459, GainRatio = 0.459
Rez_pak:                  Gain = 0.000, GainRatio = 0.000
Best split attribute is: Dydis
Stepwise chosen attribute was: Dydis
Node 12 Didelis (trukmė 6-12 mėn., >7 žmonės, 3601-24000 val.) yields: 1 R5 (50%), 0 R7 (0%), 0 R2 (0%), 0 R6 (0%),
0 R3 (0%), 0 R1 (0%), 1 R4 (50%) Rekom cases
```

12 pav. Entropijos skaičiavimo fragmentas

3. KURIAMOS EKSPERTINĖS SISTEMOS REALIZACIJOS PROJEKTAS

Ekspertinė sistema – tai sistema padedanti priimti sprendimą ar pateikianti rekomendaciją konkrečios dalykinės srities problemai spręsti. Ekspertinės sistemos žinių bazėje kaupiamos siauros dalykinės srities eksperto žinios, kuriomis, pritaikius sprendimo išvedimo mechanizmą, gali pasinaudoti kiti vartotojai. Ekspertinių sistemų kūrimui naudojami įrankiai – tuščios ekspertinės sistemos. Sekančiame skyriuje pateikiama tokių įrankių apžvalga.

3.1. Įrankiai ekspertinėms sistemoms kurti

Daugelis ekspertinių sistemų sukurtos, naudojant įrankius, taip vadinamus ekspertinių sistemų apvalkalus (angl. *shell*). Toliau aprašomi keli tokie įrankiai.

ILOG Rules

Šio įrankio pagalba verslo taisyklės aprašomos naudojant verslo veiksmų kalbą BAL (angl. *Business Action Language*), techninę taisyklių kalbą TRL (angl. *Technical Rule Language*), sprendimų lentelės formatą ar IF-THEN-ELSE formatą.

Blaze Advisor

Tai verslui skirtas įrankis, leidžiantis ne techniniam vartotojui kurti ir valdyti veiklos taisykles, paprasta ir lengvai suprantama taisyklių aprašymo kalba.

VT kurti Blaze Advisor įrankiu naudojama:

- struktūrinė taisyklių užrašymo kalba SRL (angl. *Structured Rule Language*);
- sprendimų lentelė;
- sprendimų medis;
- taisyklių šablonas.

Quick Rules

Tai veiklos taisyklių valdymo sistema, sukurta bendrovės YASU Technologies. *Quick Rules* taisyklių rašymui naudoja *Quick Rules* taisyklių žymėjimo kalbą QRML (angl. *Quick Rules Markup Language*). Taisyklės gali būti saugomos XML formatu [35].

Infrex

Tai įrankis, leidžiantis vartotojui įvesti naujas ar keisti senas taisykles ir taip nesunkiai valdyti vis atsirandančius pokyčius jų veikloje. Skirtingai nei kiti, *Infrex* gali būti integruojamas į kitas programas, naudojančias C/C++/Java/C# kalbas [35]. Taisyklių interpretavimui naudojama paprasta taisyklių kalba. Taisyklėms išreikšti naudojama:

- deklaratyvi IF-THEN forma;
- procedūrinė IF-THEN-ELSE/ELSE IF forma;
- sprendimų lentelė;
- sprendimų medis;

- orientuotas grafas.

Open Rules

Tai nemokamas produktas, skirtas bendrosios paskirties verslo taisyklių ir sprendimų valdymui. Įrankis leidžia kurti kelias arba visą kompleksą sistemų, kurios efektyviai valdo tūkstančius, šimtus tūkstančių ar net daugiau verslo taisyklių.

e2gLite

e2gLite yra Java programa, kuri gali veikti kaip „įskiepis“ (angl. applet) į tinklalapį arba kaip savarankiška programa kompiuteryje. Tai nemokama programa, kuria galima naudotis tiek mokslo, tiek komerciniais tikslais. Sprendimo išvedimui naudojamas tiek tiesioginis, tiek atbulinis sprendimo išvedimas. Žinių bazės pildymas nėra sudėtingas, o tai suteikia lankstumo naudojantis sistema. Turi įrankį *e2gRuleInduction*, skirtą sprendimų medžio formavimui, pritaikius ID3 algoritmą.

Atlikus įrankių, skirtų ekspertinėms sistemoms kurti, sprendimui realizuoti pasirinktas e2gLite įrankis.

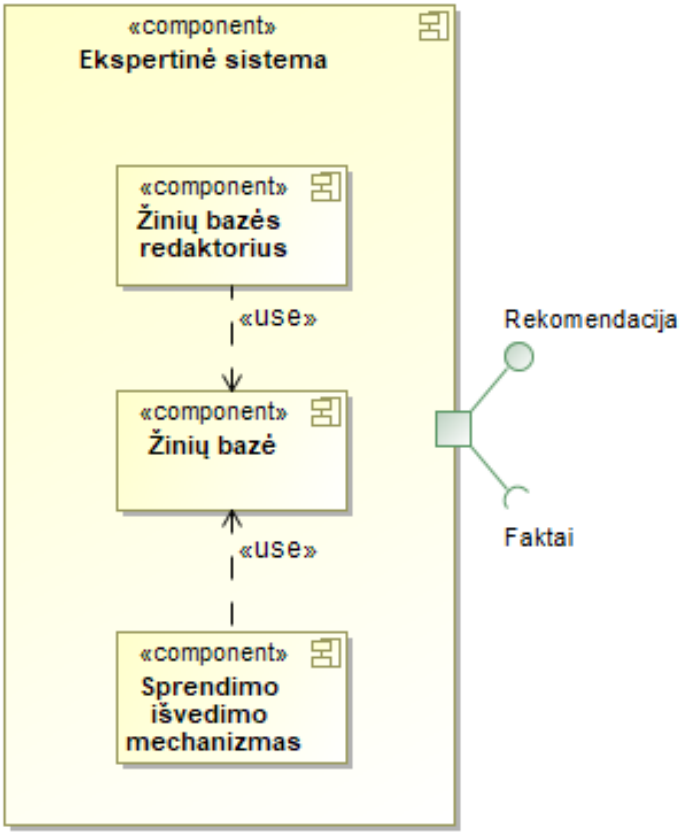
3.2. Ekspertinės sistemos struktūra

Ekspertinių sistemų kūrimas grindžiamas žinių inžinerija. Trumpai tariant, konkrečios dalykinės srities ekspertų žinios formalizuojamos ir perkeliamos į kompiuterinę programą. Taikydama įvairius žinių vaizdavimo modelius, ekspertinė sistema operuoja žiniomis ir pateikia sprendimą (rekomendaciją). Kitaip šį procesą galima vadinti *žinių vaizdavimu*.

Kuriamos ekspertinės sistemos struktūra yra sudaryta iš šių komponentų:

1. Specifinės žinių bazės;
2. Sprendimų išvedimo mechanizmo;
3. Žinių bazės redaktoriaus;
4. Vartotojo sąsajos.

Grafinis ekspertinės sistemos vaizdas pateikiamas 13 pav.



13 pav. Ekspertinės sistemos struktūra

Daugelis žinių vaizdavimo modelių netikslioms žinioms atvaizduoti taiko neapibrėžtumo ar netikslumo įvertinimo būdus.

Kuriant ekspertines sistemas, yra taikomas produkcinėmis taisyklėmis grindžiamas žinių vaizdavimo modelis. Produkcinės taisyklės turi paprastą JEI – TAI (angl. *if – then*) struktūrą (JEI <sąlyga>, TAI <išvada>). Ekspertinėms sistemoms kurti naudojami įrankiai – tuščios ekspertinės sistemos.

Ekspertinės sistemos veikimo principas būtų toks:

1. Sistema renka informaciją apie kuriamos PĮ projektą. Tai atliekama klausimų pagalba.
2. Sistema, remdamasi taisyklėmis, analizuoja vartotojo atsakymus.
3. Sistema pritaiko sprendimų išvedimo mechanizmą ir pateikia rekomendaciją.

Žinių bazė gali būti kuriama tekstiniame redaktoriuje (pvz. Notepad++ programa) arba generuojama įrankio e2gRuleWriter pagalba ir išsaugoma plėtiniu *.kb*. Tame pačiame kataloge talpinamas išvadų generatorius e2gRuleEngine.

Žinių bazės struktūrą sudaro taisyklės, klausimai ir išvados, aprašomos ir saugomos tame pačiame tekstiniame faile. Taisyklės prasideda komanda RULE kartu su sąlygine dalimi IF ir logine išraiška, o baigiama išvados dalimi, kuri prasideda komanda THEN.

Taisyklės pavyzdys:

```
RULE [Rule 27  node 52].1
IF [R_det] = "Det" AND
   [Rez_pak] = "Taip" AND
   [Dydis] = "D" AND
   [Vert_L] = "Detalus" AND
   [Vert_A] = "visam projektui" AND
   [K] = "Pastovi" AND
   [P_tipas] : "Unikalus" "Tipinis"  AND
   [Stand] : "Ne" "Taip"
THEN [Rekomendacija] = "R1" @ 50
```

Klausimai prasideda komanda PROMT su atributo pavadinimu laužtiniuose skliaustuose. Galimi klausimų tipai pateikiami 3.1 lentelėje.

3.1 lentelė. Ekspertinės sistemos klausimų tipai

Klausimo tipas	Aprašymas
MultChoice	Leidžia pasirinkti vieną iš pateiktų atsakymo variantų su „Aš nežinau alternatyva“.
YesNo	Loginė įvestis su galimomis reikšmėmis „Taip“, „Ne“, „Aš nežinau“.
ForcedChoice	Leidžia pasirinkti vieną iš pateiktų atsakymo variantų be „Aš nežinau alternatyvos“.
Choice	Leidžia pasirinkti atsakymą iš išskleidžiamo sąrašo.
AllChoice	Leidžia pasirinkti visus galimus atsakymus į pateiktą klausimą.
Numeric	Priima vartotojo įrašytą skaitinę vertę.

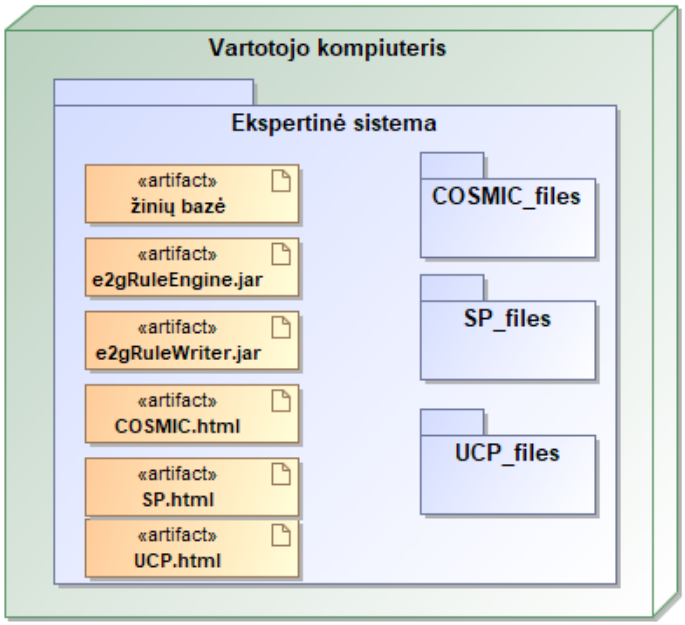
Klausimo eilutę galima užbaigti simboliais CF, tai ekspertinės sistemos naudotojui leis nustatyti pateikto atsakymo pasitikėjimo faktorių. Jei CF yra neapibrėžtas, naudotojo įvestis, pagal nutylėjimą, bus priimta su 100% tikrumu.

Išvados yra pateikiamos komanda GOAL. Išvados gali būti rašomos prieš arba po klausimais, tačiau būtinai po taisyklėmis. GOAL yra atributai, kuriems išvadų generatorius ieško verčių.

Sistemoje veikia paaiškinimo funkciją, kuri paaiškina rekomendacijos pateikimo logiką. Kaip atrodo paaiškinamasis langas parodyta 16 pav.

3.3. Sistemos diegimo modelis

Ekspertinės sistemos diegimo modelis parodytas 14 pav.



14 pav. Sistemos diegimo modelis

Sistemos paleidimui vartotojo kompiuteryje turi būti įdiegta *Java* programa ir kurioje nors direktorijoje patalpintas katalogas „Ekspertinė sistema“. Kataloge privalo būti žinių bazės tekstinis failas, sprendimų išvedimo generatorius *e2gRuleEngine.jar* bei rekomendacijų aprašymams naudojami failai ir katalogai. Taip pat kompiuteryje reikia apsirašyti sisteminį kintamąjį *CLASSPATH*. Norint pildyti, redaguoti žinių bazę reikalingas (bet nebūtinas) žinių bazės redaktorius *e2gRuleWriter.jar*.

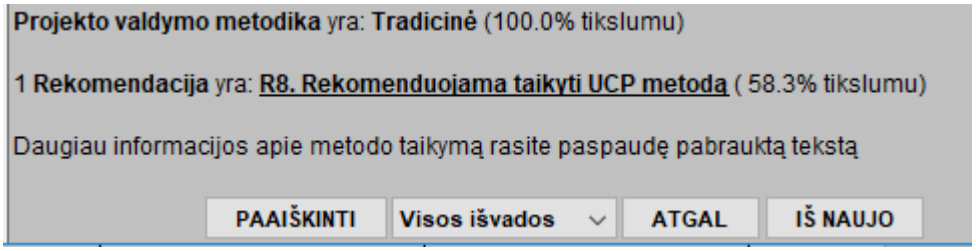
3.4. Sprendimo testavimas

Darbo metu sukurta ekspertinė sistema, kuri vartotojui užduoda klausimus ir pritaikiusi taisykles, pateikia rekomendaciją. Sistemos veikimo pavyzdys pateikiamas lentelėje (3.2 lentelė).

3.2 lentelė. Sistemos veikimo pavyzdys

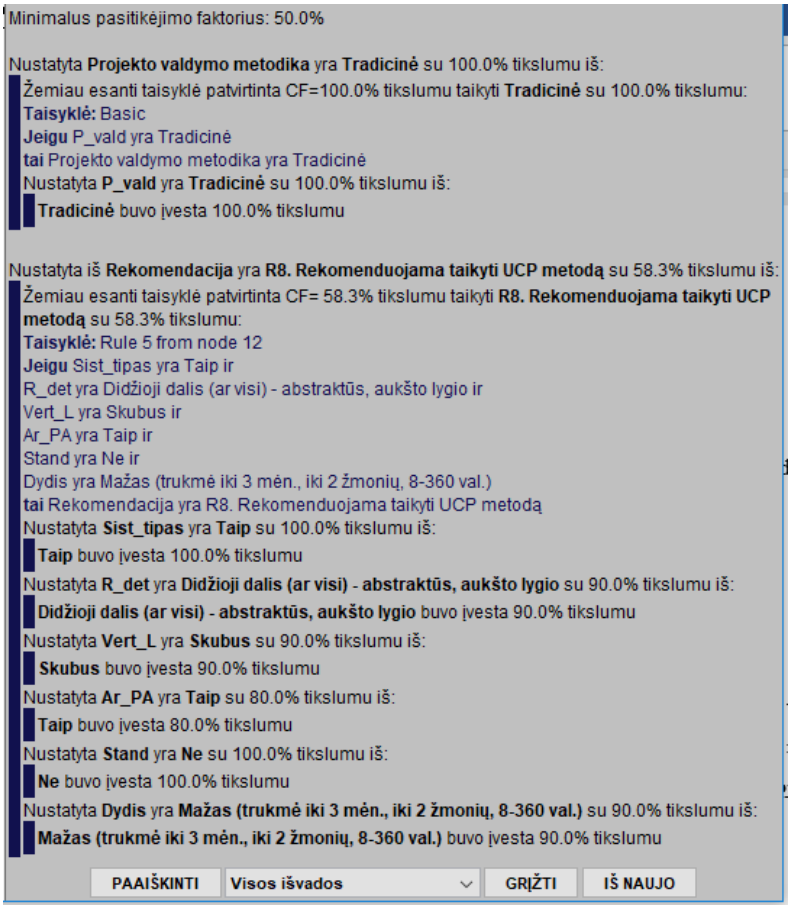
Eil. Nr.	Užduodamas klausimas	Pateikiamas atsakymas	Priskirtas CF
1.	Projekto valdymo metodika, kurią taikysite vystant šį projektą (pasirinkite vieną variantą)	Tradicinė	100
2.	Ar PĮ, kurią norite įvertinti, skirta dirbti su įvesties, saugojimo, atkūrimo bei išvesties duomenimis?	Taip	100
3.	Reikalavimų detalumas vertinimo momentu	Didesnė dalis (ar visi) – abstraktūs, aukšto lygio	90
4.	Kalbant apie vertinimo proceso išsamumą, šis vertinimas yra:	Skubus (preliminarus)	90
5.	Ar sistemos reikalavimams aprašyti taikėte panaudojimo atvejų metodiką?	Taip	80
6.	Ar sistemos apimties įvertinimui privalote naudoti standartizuotą (pvz. ISO, LST) metodą?	Ne	100
7.	Vertinant projektą pagal dydį, kuriai kategorijai priskirtumėte šį projektą (pasirinkite vieną):	Mažas (trukmė iki 3mėn., komanda iki 2 žmonių, 8-360 val.)	90

Pagal 3.2 lentelėje pateiktą klausimų-atsakymų eigą gauta rekomendacija pavaizduota 15 pav.



15 pav. Rekomendacija 3.2 lentelėje atliktam testui

. Sistemos paaiškinimas, kodėl buvo pateikta ši rekomendacija, pateikiamas 16 pav.



16 pav. Rekomendacijos pateikimo paaiškinimas

Sistemos elgseną testo metu, galima stebėti atsidarius testavimo lango režimą *(angl. Debug Mode)*. Sistemos elgsena testo metu, pateikiama 7.4 priede

4. EKSPERIMENTINĖ DALIS

Sukurtam ekspertinės sistemos prototipui įvertinti atliekamas eksperimentas, kurio tikslas – nustatyti sistemos kokybę bei praktinio panaudojamumo galimybes. Eksperimentas sudarytas iš dviejų dalių. Pirmoje dalyje tikrinamas rekomendacijų pateikimo vienodumas tiems patiems projektams, antroje – vertinamas vartotojų pasitenkinimas sistema.

4.1. Pirma eksperimento dalis

Eksperimento dalyviai - du vienos įmonės darbuotojai:

- projektų vadovas;
- programuotojas.

Abu dalyviai dažnai dirba vienoje komandoje vykdydami įvairius PĮ projektus. Kartu jie dalyvauja ir projektų, su kuriais dirba, apimties įvertinimo procese.

Eksperimento užduotis – patikrinti ar sistema pateikia vienodas rekomendacijas skirtingiems vartotojams. Taip pat, vienam iš eksperimente naudotų projektų, pagal gautą rekomendaciją, dalyviai apskaičiuos projekto apimtį.

4.1 lentelėje pateikiama pradinė informacija apie projektus, kuriems ekspertinė sistema pasiūlys apimties įvertinimo metodą.

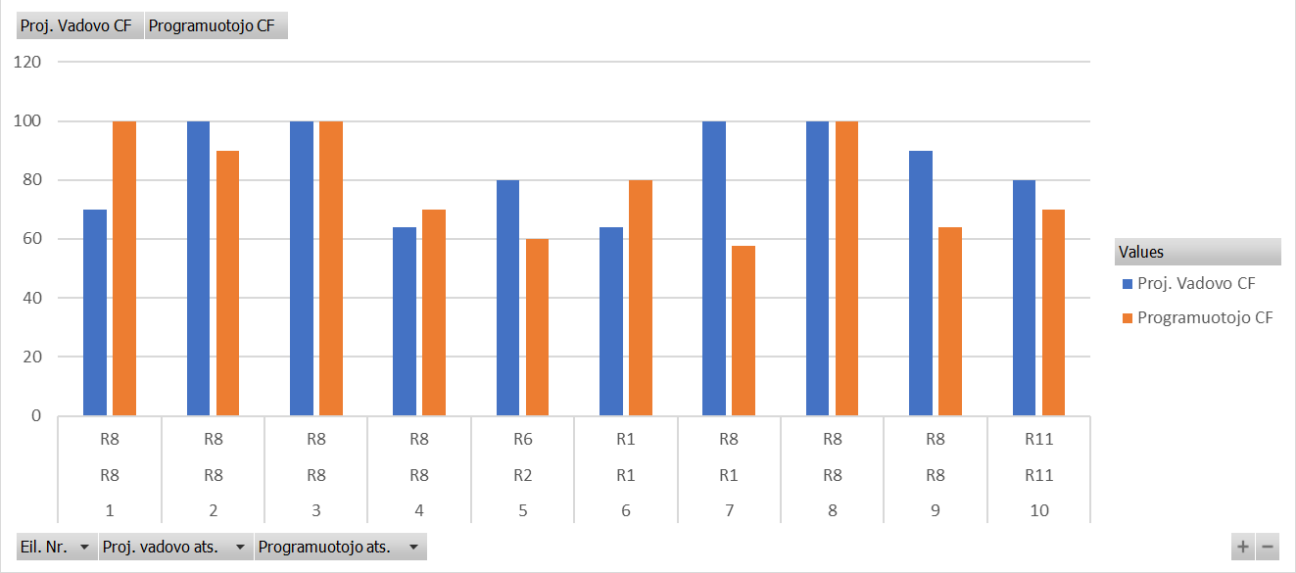
4.1 lentelė. Projektai eksperimentui

Eil. Nr.	Projekto trumpas pavadinimas	Faktinė trukmė, val.	Faktinė trukmė, mėn.	Spėjama trukmė, val.	Spėjama trukmė, mėn.	Naudota vertinimo metodika
1	Gamybos planas	706	4,0	610	3,5	Spėjimas
2	Receptūros	680	3,8	510	2,9	Spėjimas
3	Laiko apskaita	360	2,0	176	1,0	Spėjimas
4	Part. Kredito limitai	205	1,2	180	1,0	Spėjimas
5	Nejudėjusios vertybės	210	1,2	195	1,1	Spėjimas
6	Sutarčių valdymas	700	4,0	350	2,0	Spėjimas
7	Prekių poreikiai	1300	7,4	700	4,0	Spėjimas
8	Darbuotojų skatinimas	355	2,0	440	2,5	Spėjimas
9	Pakavimo specifikacijos	792	4,5	700	4,0	Spėjimas
10	Gamybos apskaita	-	-	3168	18	Konsultacija, spėjimas

4.2 lentelėje pateikiamos abiejų dalyvių gautos rekomendacijos su pasitikėjimo faktoriaus įverčiu. Grafinis šių rezultatų vaizdas pateikiamas 17 pav.

4.2 lentelė. Eksperimento Nr.1 dalyviams pateiktos rekomendacijos

Projekto Nr.	Rekomendacijos projektų vadovui	Projektų vadovo CF, proc.	Rekomendacijos programuotojui	Programuotojo CF, proc.
1	R8	70	R8	100
2	R8	100	R8	90
3	R8	100	R8	100
4	R8	64	R8	70
5	R2	80	R6	60
6	R1	64	R1	80
7	R1	100	R8	57.6
8	R8	100	R8	100
9	R8	90	R8	64
10	R11	80	R11	70



17 pav. Eksperimento Nr.1 dalyvių atsakymų palyginimas

Iš gautų eksperimento rezultatų, matyti, kad iš 10 projektų, tik dviem projektam gautos rekomendacijos nesutapo. Galima numanyti, kad rezultatų nesutapimą sukėlė dalyvių abejonės dėl kai kurių atsakymų į pateiktus klausimus pasirinkimo.

Kitoje eksperimento dalyje abu dalyviai, nepriklausomai vienas nuo kito, apskaičiavo projekto „Receptūrų posistemė“ apimtį. Šio projekto vertinimui abiem dalyviam sistema rekomendavo naudoti UCP metodą. Nei projektų vadovas, nei programuotojas anksčiau nebuvo naudoję UCP metodo ir šįkart, atlikdami skaičiavimus, rėmėsi tik ekspertinės sistemos pateikiamu aprašymu. Projeto aktorių ir panaudojimo atvejų įvertinimas pateikiamas 4.3 lentelėje ir 4.4 lentelėje.

4.3 lentelė. Projekto „Receptūrų posistemė“ aktorių įvertinimas

Aktorius	Projektų vadovo vertinimas	Programuotojo vertinimas
Gamybos sistema	Paprastas	Paprastas
Vartotojas	Sudėtingas	Sudėtingas

4.4 lentelė. Projekto „Receptūrų posistemė“ panaudojimo atvejų įvertinimas

Panaudojimo atvejis	Projektų vadovo vertinimas	Programuotojo vertinimas
Registruoti receptūrą	Vidutinis	Vidutinis
Pildyti receptūrą	Sudėtingas	Sudėtingas
Peržiūrėti receptūrą	Paprastas	Paprastas
Trinti receptūrą	Sudėtingas	Sudėtingas

Nepakoreguotų panaudojimo atvejų taškų įverčiai pateikiami 4.5 lentelėje.

4.5 lentelė. Projekto „Receptūrų posistemė“ panaudojimo atvejų taškai UUCP

Aprašymas	Projektų vadovo vertinimas	Programuotojo vertinimas
UAW	4	4
UUCW	45	45
UUCP	49	49

Abiejų dalyvių techninių faktorių įverčiai pateikiami 4.6 lentelėje

4.6 lentelė. Projekto „Receptūrų posistemė“ techninių faktorių vertinimas

TF	Apibūdinimas	Svoris	Projektų vadovo vert.	Programuotojo vert.
T1	Sistemos pasiskirstymas	2	0	0
T2	Našumas	1	5	5
T3	Galutinio vartotojo efektyvumas	1	4	3
T4	Vidinio apdorojimo sudėtingumas	1	5	5
T5	Pakartotinis panaudojamumas	1	1	0
T6	Diegimo paprastumas	0,5	0	1
T7	Naudojimo paprastumas	0,5	3	4
T8	Portatyvumas	2	1	1
T9	Keitimo paprastumas	1	2	2
T10	Lygiagretumas	1	1	1
T11	Saugumo funkcijos	1	2	1
T12	Tiesioginis priėjimas trečiosioms šalims	1	0	0
T13	Specialių vartotojų apmokymų reikalingumas	1	0	0
$TCF = 0.6 + (0.01 \times TTF)$			0,835	0,815

Projekto „Receptūrų posistemė“ aplinkos faktorių vertės pateikiamos 4.7 lentelėje.

4.7 lentelė. Projekto „Receptūrų posistemė“ aplinkos faktorių vertinimas

EF	Apibūdinimas	Svoris	Projektų vadovo vert.	Programuotojo vert.
E1	UML mokėjimas	1,5	3	2
E2	Patirtis kuriant panašias sistemas	0,5	3	3
E3	Objektinio kūrimo patirtis	1	2	2
E4	Vadovaujančio analitiko gebėjimai	0,5	4	3
E5	Motyvacija	1	0	0
E6	Reikalavimų stabilumas	2	4	4
E7	Darbuotojai dirbantys ne pilną darbo dieną	-1	0	0
E8	Programavimo kalbos sudėtingumas	-1	0	0
$ECF = 1.4 + (-0.03 \times ETF)$			0,86	0,92

Apskaičiuoti projekto panaudojimo atvejų taškai ir projekto apimtis žmogaus darbo valandomis pateikiami 4.8 lentelėje.

4.8 lentelė. Projekto „Receptūrų posistemė“ apimties įvertinimo rezultatai

Apibūdinimas	Projektų vadovo vertinimas	Programuotojo vertinimas
UCP	35,19	36,74
Projekto apimtis, val.	703,8	734,8

Projektų vadovo ir programuotojo gautų rezultatų palyginimas su realiais pateikiamas 4.9 lentelėje.

4.9 lentelė. Projekto „Receptūrų posistemė“ gautų rezultatų palyginimas su realiais

Apibūdinimas	Reali	Spėjama	Apskaičiuota projektų vadovo	Apskaičiuota programuotojo
Projekto apimtis, žm. val.	680	510	703,8	734,8
Nuokrypis		-33.3%	3.5%	8%

Iš 4.9 lentelėje gautų rezultatų matyti, kad pritaikius ekspertinės sistemos pateiktą rekomendaciją ir apskaičiavus PĮ projekto apimtį, gauti rezultatai artimi realiems. Projektų vadovo apskaičiuota projekto trukmė žmogaus valandomis yra labai artima realiai trukmei, gautas rezultato nuokrypis tik 3,5%.

4.2. Antra eksperimento dalis

Šio eksperimento tikslas – pasiūlyti eksperimentiškai išbandyti sukurtą prototipą potencialiems naudotojams ir įvertinti jų pasitenkinimą pasiūlyta sistema.

Eksperimentas atliekamas apklausos būdu, o rezultatų vertinimas paremtas grupinio vertinimo metodų (pvz. *Delphi*) idėjomis. Pirmiausia eksperimento dalyviai išbandė pasiūlytą prototipą, tada atsakinėjo į anketos klausimus. Anketa sudaryta iš bendrųjų klausimų su pasirenkamaisiais atsakymų variantais, bei iš klausimų, skirtų pačiai sistemai įvertinti. Sistemos įvertinimui naudojama 5 balų skalė:

1 balas – rodo visišką vartotojo nepasitenkinimą, neigiamą įvertinimą;

2 balai – rodo silpną pasitenkinimą (30-49%) vertinamu elemenu;

3 balai – rodo pakankamą pasitenkinimą (50-69%) vertinamu elementu;

4 balai – rodo gerą įvertinimą (70-89%);

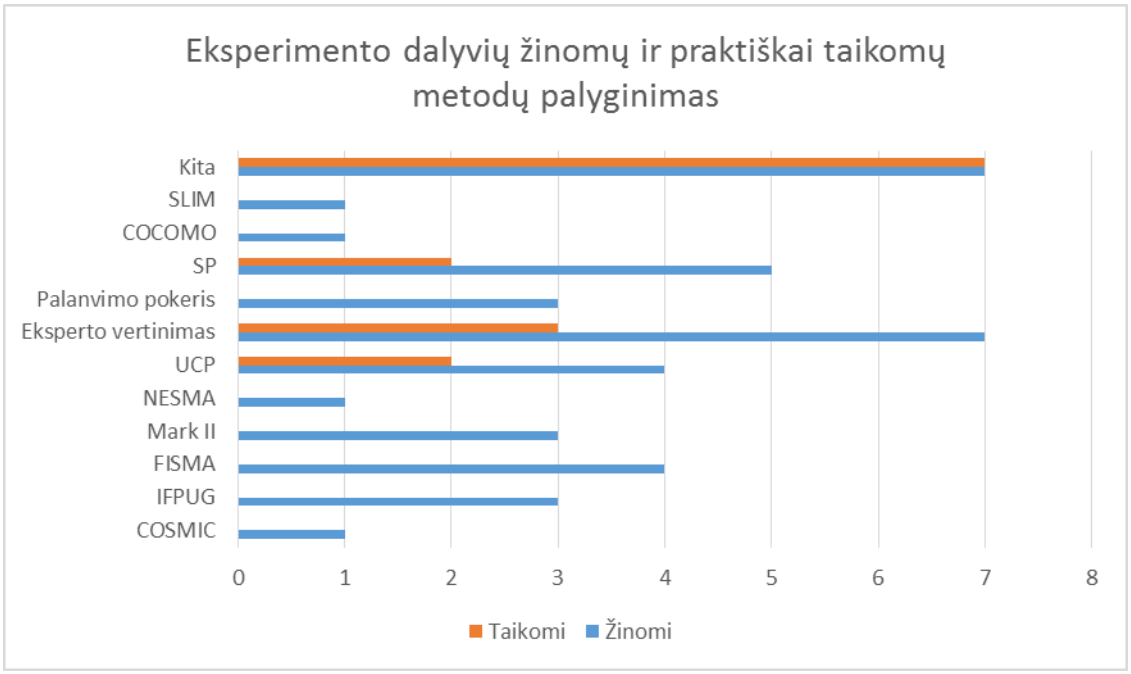
5 balai – rodo visišką vartotojo pasitenkinimą (90-100%).

Apklausoje dalyvavo 7 dalyviai iš 3 skirtingų įmonių. Lentelėje (4.10 lentelė) pateikiami bendrieji anketos klausimai.

4.10 lentelė. Bendrieji anketos klausimai

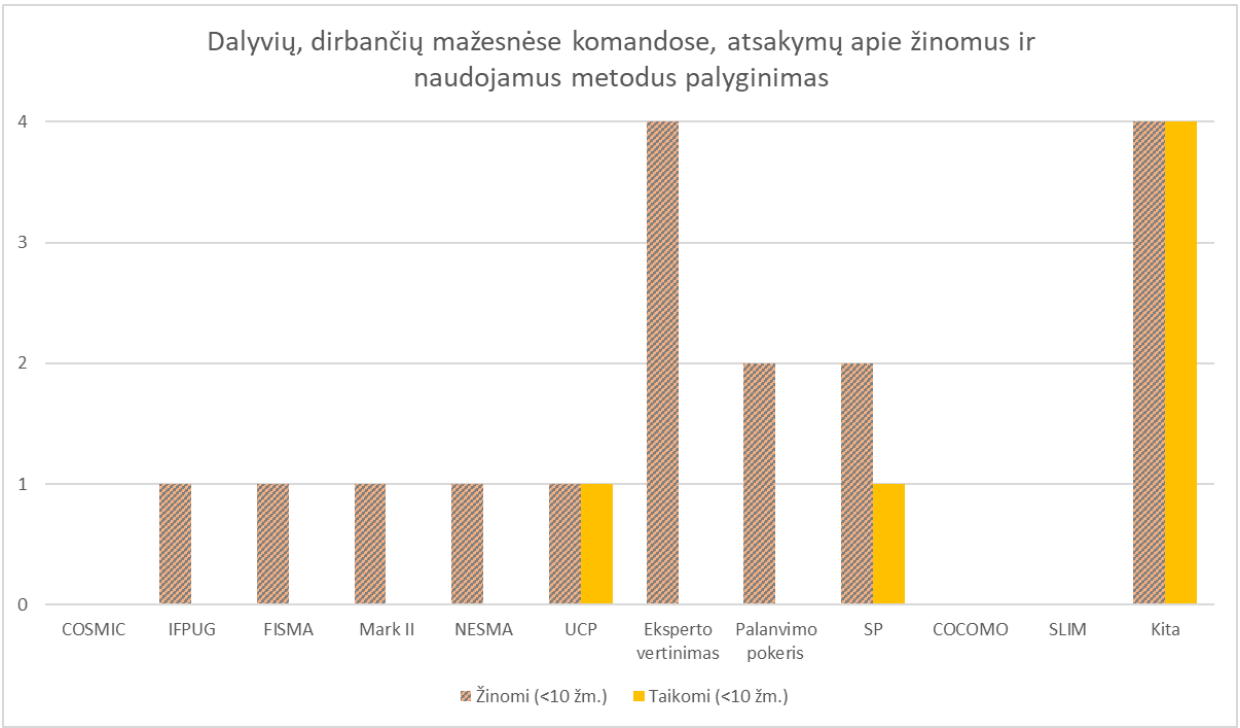
Eil. Nr.	Klausimas	Galimi atsakymų variantai
1.	Kokio dydžio komandose dažniausiai tenka dirbti?	-komanda iki 10 žmonių -komanda 10 ir daugiau žmonių
2.	Kokia Jūsų patirtis (metais) dirbant su informacinių sistemų projektais?	-iki 2 metų -nuo 3 iki 5 metų -daugiau nei 5 metai
3.	Projektai, su kuriais dirbate, dažniausiai yra skirti tarptautinei ar Lietuvos rinkai?	-tarptautiniai projektai -LT rinkai skirti projektai
4.	Kokius programinės įrangos kūrimo projektų apimties įvertinimo metodus (būdus) žinote, esate girdėjęs?	COSMIC, IFPUG, FISMA, UCP (panaudojimo atvejų taškų), SP (story points), eksperto įvertinimas, planavimo pokeris, kodo eilučių skaičiavimas, Mark II, COCOMO, SLIM, kita
5.	Kokius programinės įrangos kūrimo projektų apimties įvertinimo metodus taikote praktikoje?	COSMIC, IFPUG, FISMA, UCP (panaudojimo atvejų taškų), SP (story points), eksperto įvertinimas, planavimo pokeris, kodo eilučių skaičiavimas, Mark II, COCOMO, SLIM, kita

Eksperimento dalyvių atsakymai į bendruosius klausimus pavaizduoti diagramoje 18 pav. ir lentelėje, pateiktoje 7.5 priede.



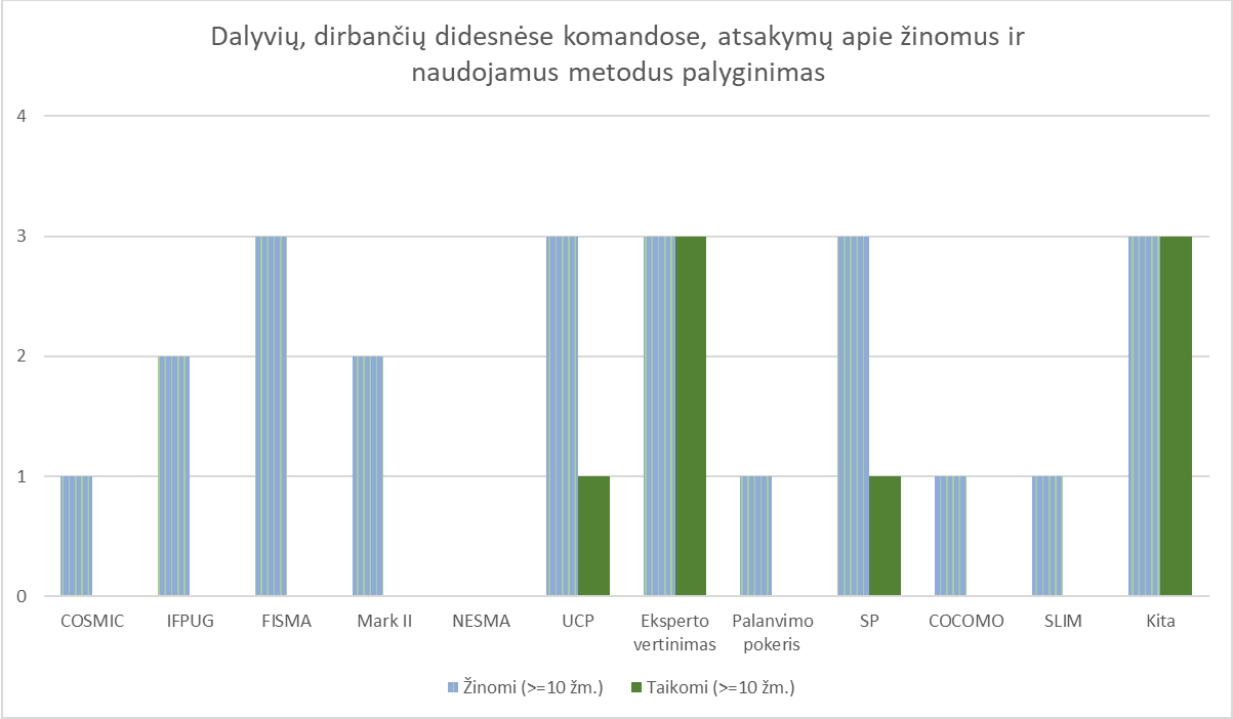
18 pav. Eksperimento dalyvių žinomų ir praktiškai naudojamų metodų palyginimas

Kaip pasiskirstę eksperimento dalyvių, dirbančių komandose iki 10 žmonių, atsakymai matyti 19 pav. pateiktoje diagramoje.



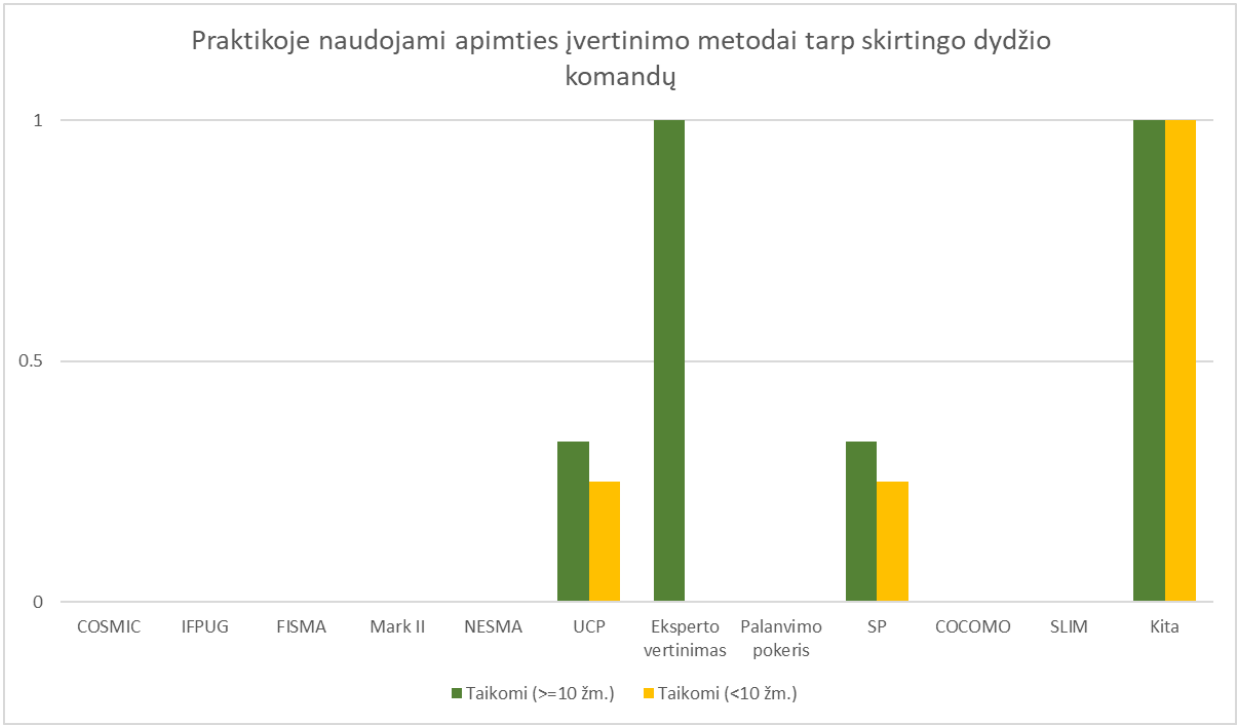
19 pav. Eksperimento dalyvių, dirbančių mažesnėse komandose, atsakymų apie žinomus ir naudojamus metodus palyginimas

Eksperimento dalyvių, dažniau dirbančių didesnėse komandose, atsakymų grafinis vaizdas pateikiamas 20 pav.



20 pav. Eksperimento dalyvių, dirbančių didesnėse komandose, atsakymų apie žinomus ir naudojamus metodus palyginimas

Kokie apimties įvertinimo metodai naudojami praktikoje, tiek mažesnėse, tiek didesnėse komandose, matyti diagramoje 21 pav.



21 pav. Eksperimento dalyvių praktikoje naudojamų metodų palyginimas tarp skirtingo dydžio komandų

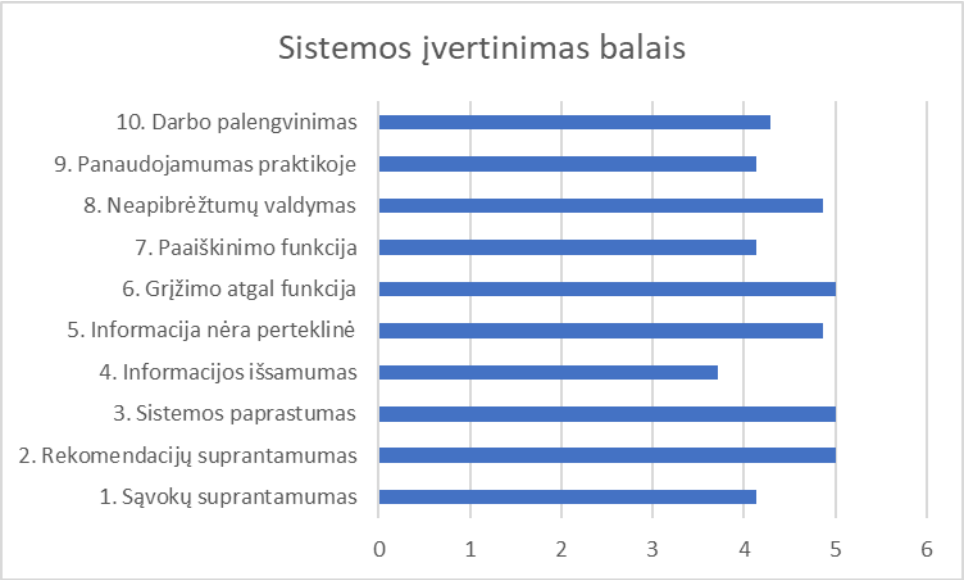
Vertinant aukščiau pateiktas diagramas apie eksperimento dalyvių naudojamus PĮ apimties įvertinimo metodus, matome, kad didesnėse komandose populiaru naudotis ekspertų vertinimais, tuo

tarpu mažesnės komandos šio vertinimo būdo nenaudoja. Taip pat visi dalyviai pažymėjo, jog PĮ dydžiui įvertinti naudoja kitus, anketoje nepažymėtus būdus. Keli dalyviai, dirbantys toje pačioje įmonėje, paminėjo, jog PĮ vertinimui naudoja toje įmonėje patvirtintas, vidines taisykles. Dalyviai, dirbantys mažesnėse komandose, dažnai PĮ apimties įvertinimą atlieka remdamiesi savo patirtimi arba panašių, praeityje darytų, projektų duomenimis ir pripažino, jog tokio vertinimo rezultatai smarkiai skiriasi nuo faktinio.

Dalyvių pasitenkinimo sistema vertinimas pateikiamas 4.11 lentelėje ir 22 pav.

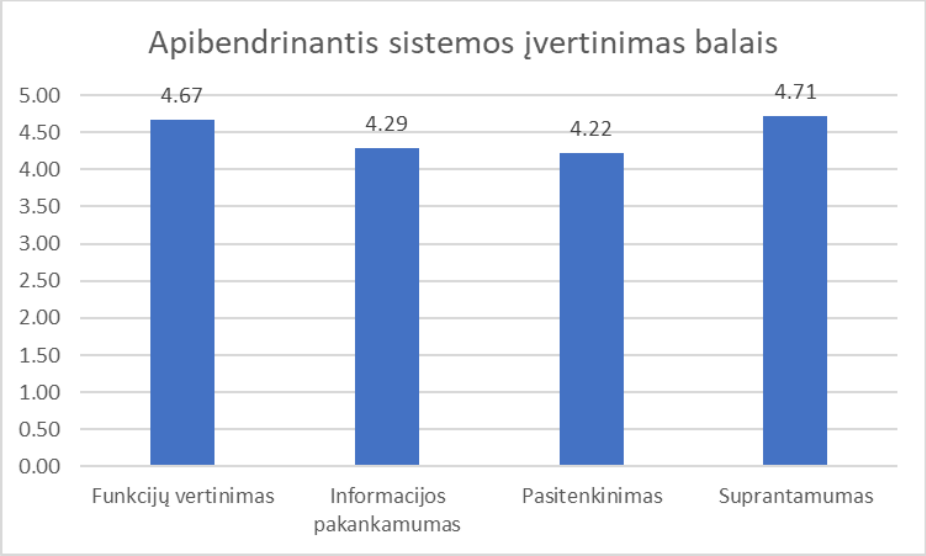
4.11 lentelė. Dalyvių pasitenkinimo sistema įvertinimo rezultatai

Eil. Nr.	Klausimas	Atsakymų vidurkis (balais)	Dalyvių komentarai
Suprantamumas			
1.	**Sąvokų suprantamumas**: Jūsų manymu, sistemoje naudojamos sąvokos yra suprantamos.	4,14	
2.	**Rekomendacijų suprantamumas**: Jūsų manymu, pateikiamos rekomendacijos yra aiškios ir suprantamos.	5	
3.	**Sistemos paprastumas:** Ar manote, kad naudotis sistema galima išmokti savarankiškai per ne ilgesnį nei 15 min. laiko tarpą?	5	
Informacijos pakankamumas			
4.	**Informacijos išsamumas**: Ar manote, kad sistemoje pateikiama informacija yra išsami ir nėra poreikio ieškoti papildomų informacijos šaltinių?	3,71	Trūksta informacijos apie integracijų su kitomis sistemomis įtaką projekto dydžiui
5.	**Informacija nėra perteklinė:** Ar manote, kad dialogo languose yra tik ta informacija, kurios vartotojui reikia (t.y. nėra perteklinės informacijos)?	4,86	
Funkcijų vertinimas			
6.	**Grįžimo atgal funkcija**: Įvertinkite *grįžimo į bet kurį klausimą* funkciją. Ar ji atitinka Jūsų poreikius?	5	
7.	**Paaiškinimo funkcija**: Įvertinkite *paaiškinimo* funkciją. Ar ji atitinka Jūsų poreikius?	4,14	
8.	**Neapibrėžtumų valdymas**: Įvertinkite neapibrėžtumų valdymą sistemoje. Ar pasitenkinimo faktoriaus CF priskyrimo funkcija atitinka Jūsų poreikius?	4,86	
Pasitenkinimas			
9.	**Panaudojamumas praktikoje**: Ar naudotumėte tokią sistemą praktikoje?	4,14	Naudotų, jei dirbtų įmonėje, kurioje nėra vidinių taisyklių, apibrėžiančių vertinimo procesą.
10.	**Darbo palengvinimas**: Ar manote, kad naudojantis šia sistema, palengvina (sutrumpina) apimties įvertinimo metodo pasirinkimo procesą?	4,29	



22 pav. Sistemos pasirinktų kriterijų įvertinimo rezultatai

Apklausos atsakymų apie vartotojų pasitenkinimą sistema apibendrinti rezultatai pateikiami 23 pav.



23 pav. Apibendrinantis sistemos kriterijų įvertinimas balais

Atliktas eksperimentas parodė, jog sukurtas prototipas įvertintas aukštais balais, o bendras įvertinimo balų vidurkis yra 4,51. Tyrimas parodė, jog sistema galėtų būti naudojama praktikoje.

5. IŠVADOS

1. Atlikus PĮ kūrimo projektų apimties įvertinimo metodų analizę, nustatyta, kad yra didelė tokių metodų įvairovė, tačiau nėra aiškių taisyklių kada ir koks metodas yra tinkamiausias naudoti.
2. Atlikus PĮ apimties įvertinimo metodų palyginimą ir kitų mokslininkų tyrimų analizę, nustatytos svarbiausios PĮ kūrimo projekto charakteristikos, tokios kaip projekto vystymo metodika, projekto gyvavimo ciklo fazė, PĮ tipas ir kt., darančios įtaką apimties įvertinimo metodo pasirinkimui. Šios charakteristikos panaudotos kaip pradiniai duomenys sudarant klausimų ir rekomendacijų rinkinį.
3. Atlikus mokslininkų darbų analizę, nustatyta, kad nėra automatizuoto sprendimo, galinčio rekomenduoti vienu ar kitu atveju tinkamiausią apimties įvertinimo metodą.
4. Įrankio *e2glite* pagalba sukurta ekspertinė sistema, kuri vartotojui užduoda klausimus apie projektą ir, įvertinusi atsakymus, pateikia rekomendaciją, o tai leidžia pagreitinti ir palengvinti tinkamo metodo pasirinkimą bei taikymą.
5. Pirmojoje eksperimento dalyje siekiama išsiaiškinti ar projektui siūlomas tas pats apimties įvertinimo metodas, nepriklausomai nuo to, kuris žmogus atliko ekspertinės sistemos testą. Du eksperimento dalyviai, pasitelkdami ekspertinę sistemą, ieškojo tinkamų apimties įvertinimo metodų 10-čiai projektų. Eksperimento rezultatai parodė, jog 8 projektams dalyviai gavo vienodas rekomendacijas. Visgi, 2 projektams gautos rekomendacijos nesutapo, kas leidžia manyti, jog kai kurie sistemos klausimai dalyvių buvo suprasti skirtingai. Galimų interpretacijų skaičiui sumažinti, reikėtų prie kiekvieno klausimo pateikti detalius paaiškinimus, kuriuos, esant poreikiui, vartotojas peržiūrėtų.
6. Pirmosios eksperimento dalies dalyviai pasirinko vieną projektą ir pagal pateiktą rekomendaciją, apskaičiavo projekto apimtį. Gauti rezultatai parodė, jog abiejų dalyvių skaičiavimai nedaug skiriasi nuo realių (vidutinis nuokrypis 7,5%), o tai leidžia manyti, jog rekomendacija projektui parinkta tinkamai ir aprašyta suprantamai.
7. Antrojoje eksperimento dalyje buvo vertinamas vartotojų pasitenkinimas sistema. Iš apklausos metu gautų rezultatų matyti, jog sukurta ekspertinė sistema įvertinta teigiamai – vertinimo vidurkis 4,5 iš 5.
8. Eksperimento rezultatai parodė, jog didžiausią pasisekimą sistema turėtų tarp mažesnėse komandose dirbančių eksperimento dalyvių. Remiantis eksperimento rezultatais, galima teigti, jog tie respondentai, kurie dirba su Lietuvos rinkai skirtais projektais ir mažesnėse komandose, dažniausiai projektų apimtį vertina naudodami spėjimą arba lygindami panašius projektus. Būtent šiai vartotojų grupei pasiūlyta ekspertinė sistema neštų didžiausią naudą.
9. Iš sistemos vertinimui pasirinktų kriterijų blogiausiai įvertintas informacijos išsamumas – 3,71 balai. Sistemoje pasigesta klausimų ir rekomendacijų apie PĮ integracijų įtakos projekto

apimčiai įvertinimą. Taigi ateityje, sistemos tobulinimui būtų galima išplėsti klausimų ir rekomendacijų rinkinį, įtraukiant integracijų su kitomis sistemomis įtakos projekto dydžiui įvertinimą.

6. LITERATŪRA

[1] O. V. Juozas Laucius, Informacinių sistemų projektų ir kokybės valdymas, Vilnius: Technika, 2008.

[2] Norman Fenton, James Bieman, Software Metrics– A Rigorous and Practical Approach, Third Edition, CRC Press, 2014.

[3] M.Ahmad Khan, A.Parveen, M.Sadiq, „A method for the selection of software development life cycle models using analytic hierarchy process," įtraukta *ICICT*, Ghaziabad, 2014.

[4] N. B. Ruparelia, „Software development lifecycle models," *ACM SIGSOFT Software Engineering Notes*, t. 35, nr. 3, pp. 8-13, 2010.

[5] T.Fehlmann, L.Santillo, „From Story Points to COSMIC Function Points in Agile Software Development – A Six Sigma perspective," įtraukta *Magdeburger Schriften zum Empirischen Software Engineering*, Stuttgart, 2010.

[6] J.Aguilar, M.Sanchez, C.Fernandez-y-Fernandez, E.Rocha, D.Martinez, J.Figueroa, „The Size of Software Projects Developed by Mexican," įtraukta *Int'l Conf. Software Eng. Research and Practice*, Las Vegas, 2014.

[7] R.Goswami , M.Jasuja, S.Dhir, „Impact of Different Estimation Approaches in Traditional and Agile Development," įtraukta *CICT*, Ghaziabad, 2016.

[8] A. F. Minkiewicz, „The Evolution of Software Size: A Search for Value," *Software Engineering Technology*, pp. 23-26, 2009.

[9] Evita Coelho, Anirban Basu, „Effort Estimation in Agile Software Development using Story Points," *International Journal of Applied Information Systems*, t. 3, nr. 7, 2012.

[10] Amid Khatibi Bardsiri, Seyyed Mohsen Hashemi, „Software Effort Estimation: A Survey of Well-known Approaches," *International Journal of Computer Science Engineering* , t. 3, nr. 1, pp. 46-50, 2014.

[11] Denis Čeke, Boris Milašinović , „Early effort estimation in web application development," *Journal of Systems and Software*, t. 103, p. 219–237, 2015.

[12] GRAHAM C. LOW, D. ROSS JEFFERY , „Function Points in the Estimation and Evaluation of the Software Process," *IEEE Transactions On Software Engineering*, t. 16, nr. 1, pp. 64-71, 1990.

[13] Manfred Bundschuh,Carol Dekkers, The IT Measurement Compendium– Estimating and Benchmarking Success with Functional Size Measurement, Berlin: Springer, 2008.

[14] Juan J. Cuadrado-Gallego, Pablo Rodríguez-Soria, Alberto Lucendo, Robert Neumann, Reiner Dumke, Andreas Schmietendorf, „COSMIC Measurements Dispersion," įtraukta *2012 Joint Conference of the 22nd International Workshop on Software Measurement and the 2012 Seventh International*, Assisi, 2012.

[15] W.-M. Han, „A Decision Support Approach to the Selection of Functional Size Measurement Methods," *Journal of Information Management*, t. 21, nr. 2, pp. 185-206, 2014.

[16] B. Czarnacka-Chrobot, „Analysis of Software Development and Enhancement Projects Work Effort per Unit Based on the COSMIC Method with Regard to Technological Factors—Case Study," *Journal of Software Engineering and Applications*, t. 3, pp. 597-609 , 2013.

[17] Sohaib Shahid Bajwa, Cigdem Gencel, Pekka Abrahamsson, „Software Product Size Measurement Methods: A Systematic Mapping Study," įtraukta *2014 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement*, Rotterdam, 2014.

[18] M. Ochodek, J. Nawrocki, K. Kwarciak, „Simplifying effort estimation based on Use Case Points," *Information and Software Technology*, t. 53, nr. 3, pp. 200-213, 2011.

[19] Charles Symons, Alain Abran, Christof Ebert, Frank Vogelezang, „Measurement of software size: advances made by the COSMIC community," įtraukta *2016 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement*, Berlin, 2016.

[20] Muhammet Ali Sag, Ayça Tarhan, „Measuring COSMIC Software Size from Functional Execution Traces of Java Business Applications," įtraukta *Software Measurement and the International Conference on Software Process and Product Measurement* , Mensura, 2014.

[21] „COSMIC method," [Tinkle]. Available: http://cosmic-sizing.org/cosmic-fsm/. [Kreiptasi 9 spalio 2016].

[22] „The COSMIC Functional Size Measurement Method Versio 4.0; Measurement Manual; The COSMIC Implementation Guide for ISO/IEC 19761:2011," balandis 2015. [Tinkle]. Available: http://www.cosmicon.com/portal/public/MMv4.0.1.pdf. [Kreiptasi 15 lapkričio 2016].

[23] P. R. Hill, Practical Software Project Estimation: A Toolkit for Estimating Software Development Effort & Duration, ISBSG, 2011.

[24] Mudasir Manzoor Kirmani, Abdul Wahid, „Use Case Point Method of Software Effort Estimation: A Review," *International Journal of Computer Applications,* t. 116, nr. 15, pp. 43-47, 2015.

[25] Tülin Erçelebi Ayyıldız, Altan Koçyiğit, „An Early Software Effort Estimation Method Based on Use Cases and Conceptual Classes," *Journal of Software,* t. 9, nr. 8, 2014.

[26] P. D. Morris, „The Founding Fathers of Agile," *Cross Talk,* pp. 15-22, March/April 2017.

[27] „Sizing in Agile context," [Tinkle]. Available: http://nesma.org/themes/sizing/story-points/. [Kreiptasi 15 spalio 2016].

[28] A. E.-d. Hamouda, „Using Agile Story Points as an Estimation Technique in CMMI Organizations," įtraukta *Agile Conference*, Kissimmee, 2014.

[29] Sergio Di Martino, Filomena Ferrucci, Carmine Gravino, Federica Sarro, „Web Effort Estimation: Function Point Analysis vs. COSMIC," *Information and Software Technology,* t. 72, pp. 90-109, 2016.

[30] E.J.D. Candido, R. Sanches, „Estimating the size of Web applications by using a simplified function point method," įtraukta *Joint Conference 10th Brazilian Symposium on Multimedia and the Web 2nd Latin American Web Congress*, Washington, 2004.

[31] Jovan Popović, Dragan Bojić, „A Comparative Evaluation of Effort Estimation Methods in the Software Life Cycle," *Computer Science & Information Systems,* t. 9, nr. 1, pp. 456-487, 2012.

[32] S. Kumari, „Comparison and Analysis of Different Software Cost Estimation Methods," *International Journal of Advanced Computer Science and Applications* , t. 4, nr. 1, pp. 153-157, 2013.

[33] V. Khatibi Bardsiri, D. N. A. Jawawi, S. Z. M. Hashim, , „A flexible method to estimate the software development effort based on the classification of projects and localization of comparisons," *Empirical Software Engineering* , pp. 857-884, 2014.

[34] Ç.Gencel, L.Buglione, O.Demirors, P.Efe, „A Case Study on the Evaluation of COMIC-FFP and Use Case Points," įtraukta *SMEF*, 2006.

[35] O. I.Valatkaite, „On Business Rules Automation: The BR-Centric IS Development Framework," įtraukta *Advances in Databases and Information Systems*, Berlin, 2005.

[36] „Object oriented size measurement," [Tinkle]. Available: thttp://nesma.org/themes/sizing/use-case-points/. [Kreiptasi 17 spalio 2016].

[37] A. B. N. Mohammad Azzeh, „A hybrid model for estimating software project effort from Use Case Points," gegužė 2016. [Tinkle]. Available: http://www.sciencedirect.com/science/article/pii/S1568494616302095. [Kreiptasi 11 lapkričio 2016].

[38] Bente Anda, Hege Dreiem, Dag I. K. Sjøberg, Magne Jørgensen, „Estimating Software Development Effort Based on Use Cases — Experiences from Industry," įtraukta *≪UML≫ 2001 — The Unified Modeling Language*, Torontas, 2001.

[39] I.Myrtveit, E.Stensrud, „Reliability and Validity in Comparative Studies of Software Prediction Models," *IEEE Transactions on Software Engineering*, t. 31, nr. 5, pp. 380-391, 2005.

[40] K.Dejaeger, W.Verbeke, D.Martens, B.Baesens, „Data Mining Techniques for Software Effort Estimation: A Comparative Study," *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, t. 38, nr. 2, pp. 375-397, 2012.

[41] R.Alves, P.Valente, N.J.Nunes, „Improving Software Effort Estimation with Human-Centric Models: a comparison of UCP and iUCP accuracy," *EICS*, pp. 287-296 , 2013.

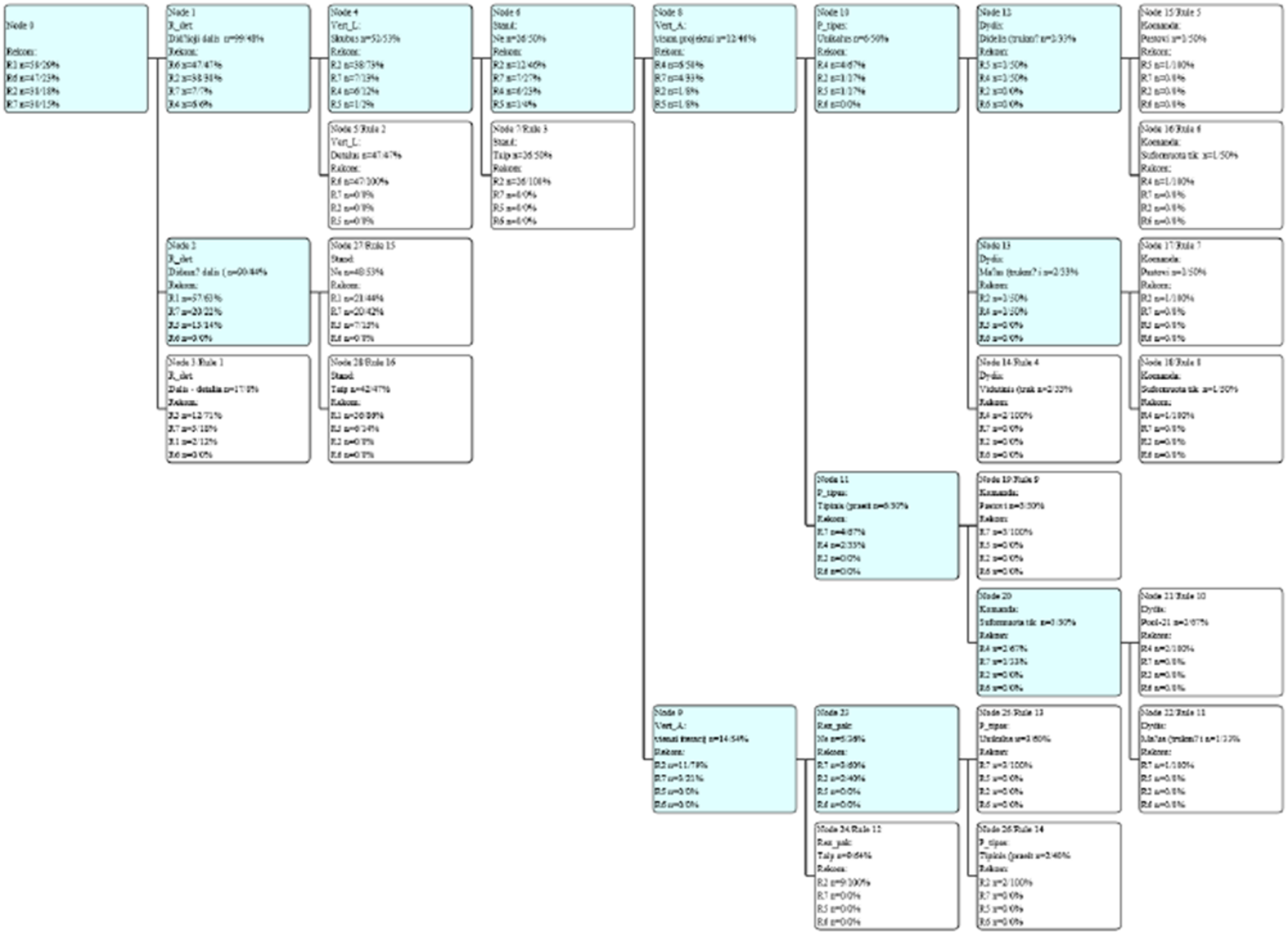
[42] A.Idri, F. azzahra Amazal, „Accuracy Comparison of Analogy-Based Software Development Effort Estimation Techniques," *International Journal of Intelligent Systems*, t. 31, p. 128–152, 2016.

7. PRIEDAI

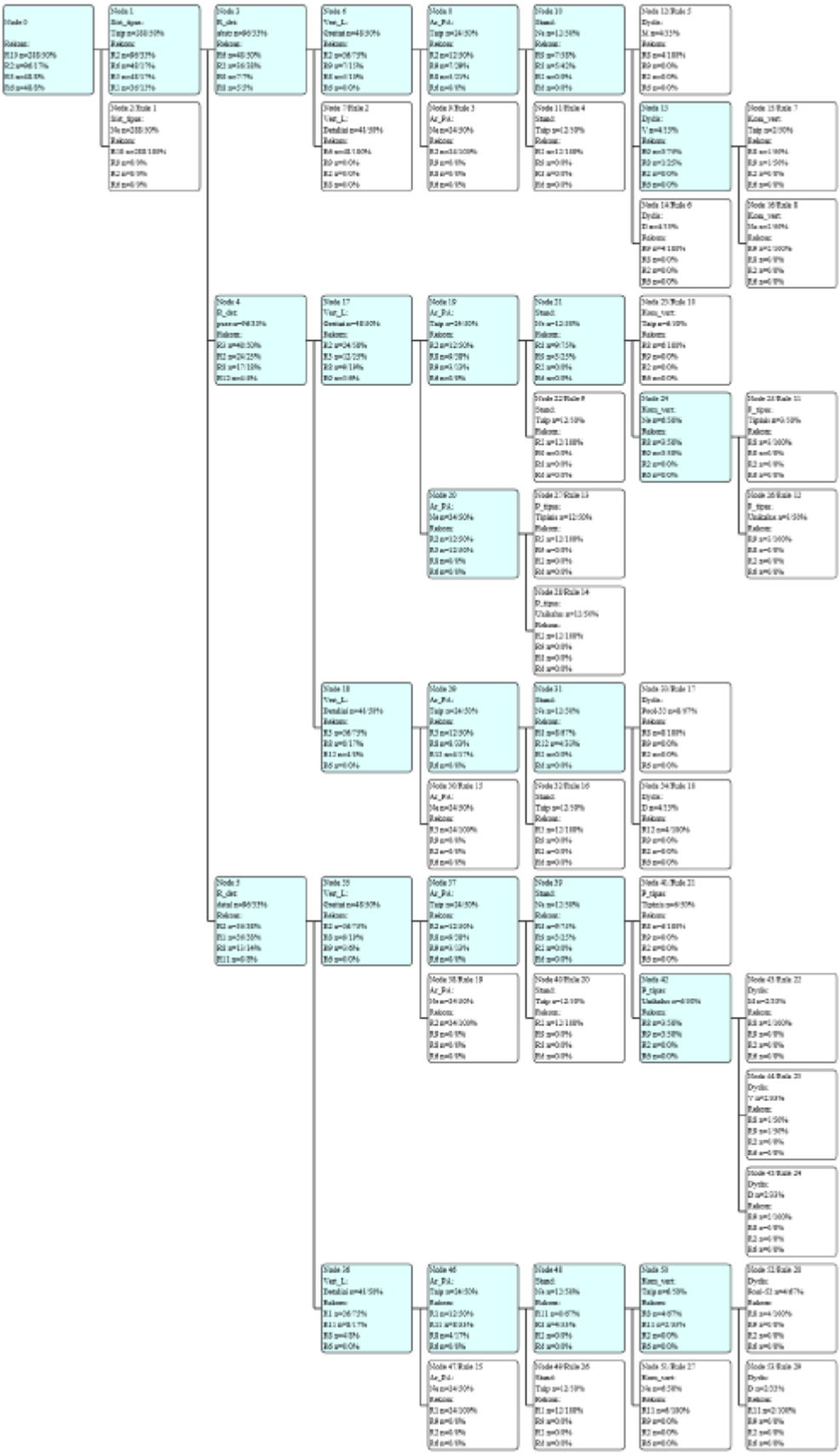
7.1. priedas. Kitų mokslininkų atliktų darbų apžvalgos apibendrinimas

Nr.	Metai	Autorius	Darbo pavadinimas	Darbo tikslas	Tikslumo vertinimas	Gautas rezultatas
1	2016	S.Martino, F.Ferrucci, C.Gravino, F.Sarro	Web Effort Estimation: Function Point Analysis vs. COSMIC [29]	Ištirti FP ir COSMIC metodų taikymą internetinių projektų apimties skaičiavimui ir išsiaiškinti ar COSMIC metodo taikymas tokiems projektams yra efektyvesnis, nei FP metodas.	MdAR, MMRE, MdMRE, Pred(25). Rezultatams vaizduoti – stačiakampės diagramos	Patvirtinta, kad internetinių projektų apimties skaičiavimams COSMIC metodas yra efektyvesnis ir tikslesnis nei FP metodas.
2	2012	J. Popović, D.Bojić	A Comparative Evaluation of Effort Estimation Methods in the Software Life Cycle [31]	Siekiama nustatyti ojektyvų PĮ dydžio įvertinimo metodą. Analizuojami ir lyginami IFPUG, NESMA, Mark II, COSMIC, UCP)	Pirsono koreliacijos koef., MRE, MMRE, MREmax,	Funkcinės apimties matavimo metodai pasižymi geru tikslumu. Išimtis – NESMA ir UCP.
3	2013	S.Kumari	Comparison and Analysis of Different Software Cost Estimation Methods [32]	Straipsnyje siūloma naudoti SVR prognuozuojant PĮ projektų dydį. Gauti rezultatai su SVR, lyginami su COCOMO ir MOPSO metodikų rezultatais. Tyrimo metu naudotas Weka modeliavimo įrankis.	MARE, PRED(25)	Su SVR gauti rezultatai yra geresni nei COCOMO ar MOPSO.
4	2014	V. Khatibi Bardsiri, D.N.A.Jawawi, S.Z.M.Hashim	A flexible method to estimate the software development effort based on the classification of projects and localization of comparisons [33]	Siūloma alternatyvi metodika projektų apimties įvertinimams, paremtiems analogija.	RE, MRE, MMRE, PRED(25) Rezultatams vaizduoti – stačiakampės diagramos	Iš trijų realių duomenų rinkinių gauti eksperimentiniai rezultatai parodė, kad siūloma projektų klasifikacija gali padidinti vertinimo modelių tikslumą
5	2015	D.Čeke, B.Milašinović	Early effort estimation in web application development [11]	Analizuojama galimybė ir kuriamas modelis internetinių projektų apimties įvertinimui taikant funkcinio dydžio ir koncepcinių modelių derinį.	MMRE, MdMRE, PRED(25). Rezultatams vaizduoti – stačiakampės diagramos ir histogramos	Autorių pasiūlytas skaičiavimo modelis yra tinkamas internetinių PĮ projektų apimties vertinimui ankstyvose stadijose.
6	2011	M. Ochodek, J. Nawrocki, K. Kwarciak	Simplifying effort estimation based on Use Case Points [18]	Detaliai ištirti UCP metodą, rasti galimybę šio metodo taikymą supaprastinti.	MRE, MMRE, PRED(25)	Tyrimo rezultatai parodė geras galimybes UCP metodo supaprastinimui. Tačiau rezultatų užtikrinimui reikėtų atlikti daugiau eksperimentinių skaičiavimų.
7	2014	T.E.Ayyıldız, A. Koçyiğit	An Early Software Effort Estimation Method Based on Use Cases and Conceptual Classes [25]	PĮ apimties vertinimui siūlomas metodas, skirtas naudoti ankstyvose projekto fazėse, paremtas konceptualių klasių modelių naudojimu.	MRE, PRED(25), AMSE	Pastebėta didelė koreliacija tarp koncepcinių klasių skaičiaus ir faktinės PĮ apimties.

7.2. Priedas. Sprendimų medžio šakos „agile“ bendras vaizdas



7.3. Priedas. Sprendimų medžio šakos „tradicinė“ bendras vaizdas



7.4. Priedas. Sistemos elgsena testavimo metu

```
STARTING NEW CONSULTATION WITH THE FOLLOWING GOAL(S):
>Rekomendacija
>Projekto valdymo metodika
Minimum confidence factor for accepting a value as a fact: 50.0%
TRYING RULE: Basic
>Add to goal stack: P_vald
>Rule status after evaluation is: Unknown
*Prompt assigned: P_vald=Tradicinė (100.0%)
REMOVE FROM GOAL STACK (Resolved): P_vald
TRYING RULE: Basic
>Rule status after evaluation is: True/Fired
*Rule assigned: Projekto valdymo metodika=Tradicinė (100.0%)
REMOVE FROM GOAL STACK (Resolved): Projekto valdymo metodika
TRYING RULE: Rule 1 from node 2
>Add to goal stack: Sist_tipas
>Rule status after evaluation is: Unknown
*Prompt assigned: Sist_tipas=Taip (100.0%)
REMOVE FROM GOAL STACK (Resolved): Sist_tipas
TRYING RULE: Rule 1 from node 2
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 2 from node 7
>Add to goal stack: R_det
>Rule status after evaluation is: Unknown
*Prompt assigned: R_det=Didžioji dalis (ar visi) - abstraktūs, aukšto lygio (
90.0%)
REMOVE FROM GOAL STACK (Resolved): R_det
TRYING RULE: Rule 2 from node 7
>Add to goal stack: Vert_L
>Rule status after evaluation is: Unknown
*Prompt assigned: Vert_L=Skubus ( 90.0%)
REMOVE FROM GOAL STACK (Resolved): Vert_L
TRYING RULE: Rule 2 from node 7
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 3 from node 9
>Add to goal stack: Ar_PA
>Rule status after evaluation is: Unknown
*Prompt assigned: Ar_PA=Taip ( 80.0%)
REMOVE FROM GOAL STACK (Resolved): Ar_PA
TRYING RULE: Rule 3 from node 9
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 4 from node 11
>Add to goal stack: Stand
>Rule status after evaluation is: Unknown
*Prompt assigned: Stand=Ne (100.0%)
REMOVE FROM GOAL STACK (Resolved): Stand
TRYING RULE: Rule 4 from node 11
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 5 from node 12
>Add to goal stack: Dydis
>Rule status after evaluation is: Unknown
*Prompt assigned: Dydis=Mažas (trukmė iki 3 mėn., iki 2 žmonių, 8-360 val.) (
90.0%)
REMOVE FROM GOAL STACK (Resolved): Dydis
TRYING RULE: Rule 5 from node 12
>Rule status after evaluation is: True/Fired
*Rule assigned: Rekomendacija=R8. Rekomenduojama taikyti UCP metodą (58.320004%)
TRYING RULE: Rule 6 from node 14
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 7 from node 15
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 8 from node 16
>Rule status after evaluation is: False/Fired
```

```
TRYING RULE: Rule 9 from node 22
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 10 from node 23
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 11 from node 25
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 12 from node 26
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 13 from node 27
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 14 from node 28
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 15 from node 30
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 16 from node 32
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 17 from node 33
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 18 from node 34
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 19 from node 38
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 20 from node 40
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 21 from node 41
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 22 from node 43
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 23 from node 44
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 24 from node 45
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 25 from node 47
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 26 from node 49
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 27 from node 51
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 28 from node 52
>Rule status after evaluation is: False/Fired
TRYING RULE: Rule 29 from node 53
>Rule status after evaluation is: False/Fired
REMOVE FROM GOAL STACK (Resolved): Rekomendacija
REPLACE ON GOAL STACK: Rekomendacija
REMOVE FROM GOAL STACK (Failed): Rekomendacija

GOAL STACK EMPTY: CONSULTATION ENDED

Added hyperlink for: Rekomendacija = R8. Rekomenduojama taikyti UCP metodą
file:///C:/Users/Laura/Desktop/mano/mano/UCP.htm
```

7.5. Eksperimento antros dalies dalyvių atsakymai į anketos klausimus

7.1 lentelė. Eksperimento antros dalies dalyvių atsakymai į bendruosius anketos klausimus

Dalyviai	Nr.1	Nr.2	Nr.3	Nr.4	Nr.5	Nr.6	Nr.7
Komandos dydis	>=10 žm.	>=10 žm.	>=10 žm.	<10 žm.	<10 žm.	<10 žm.	<10 žm.
Patirtis vertinime	< 2 m.	3 - 5 m.	< 2 m.	3 - 5 m.	> 5 m.	> 5 m.	> 5 m.
Projektai	tarptaut.	tarptaut.	tarptaut.	LT	LT	LT	LT
Kokius PĮ kūrimo projektų apimties įvertinimo metodus (būdus) žinote?							
COSMIC	1						
IFPUG	1	1		1			
FISMA	1	1	1	1			
Mark II	1	1		1			
NESMA				1			
UCP	1	1	1				1
Eksperto vertinimas	1	1	1	1	1	1	1
Palanvimo pokeris	1			1			1
SP	1	1	1	1			1
COCOMO	1						
SLIM	1						
Kita	1	1	1	1	1	1	1
Kokius PĮ kūrimo projektų apimties įvertinimo metodus (būdus) taikote praktikoje?							
COSMIC							
IFPUG							
FISMA							
Mark II							
NESMA							
UCP			1				1
Eksperto vertinimas	1	1	1				
Palanvimo pokeris							
SP			1	1			
COCOMO							
SLIM							
Kita	1	1	1	1	1	1	1

7.2 lentelė. Eksperimento antros dalies dalyvių atsakymai į anketos klausimus apie sistemą

Dalyvis	**Nr.1**	**Nr.3**	**Nr.2**	**Nr.4**	**Nr.5**	**Nr.6**	**Nr.7**	
Projektai	tarptaut.	tarptaut.	tarptaut.	LT	LT	LT	LT	
Patirtis	< 2 m.	< 2 m.	3 - 5 m.	3 - 5 m.	> 5 m.	> 5 m.	> 5 m.	
Komandos dydis	>=10 žm.	>=10 žm.	>=10 žm.	<10 žm.	<10 žm.	<10 žm.	<10 žm.	vidurkis
Sąvokų suprantamumas	4	4	4	5	4	4	4	4,14
Rekomendacijų suprantamumas	5	5	5	5	5	5	5	5,00
Informacijos pakankamumas	3	3	2	5	5	4	4	3,71
Nėra perteklinės informacijos	5	5	4	5	5	5	5	4,86
Naudojimosi lengvumas	5	5	5	5	5	5	5	5,00
Grįžimo atgal funkcija	5	5	5	5	5	5	5	5,00
Paaiškinimo funkcija	4	5	4	4	4	4	4	4,14
Neapibrėžtumų valdymas	4	5	5	5	5	5	5	4,86
Metodo pasirinkimo palengvinimas	3	4	4	5	4	5	5	4,29
Panaudojamumas praktikoje	3	3	4	5	5	5	4	4,14