ŠIAULIŲ UNIVERSITETAS

Informacinių technologijų katedra

Kęstutis Visialga

Programinės įrangos projektų valdymo modelių tyrimas

Magistro baigiamasis darbas

Vadovė doc. dr. A.Slotkienė

Šiauliai, 2013

ŠIAULIŲ UNIVERSITETAS

Informacinių technologijų katedra

TVIRTINU

IT katedros vedėjas doc. dr. M. Bernotas
2013-05-30

Programinės įrangos projektų valdymo modelių tyrimas

Informatikos inžinerijos bakalauro baigiamasis darbas

Vadovė
IT katedros docentė dr. A.Slotkienė
2013m.

Recenzentai
IT katedros lektorė dr. A.Drukteinienė
2013m.
IT katedros lektorė
2013m. dr. S.Ramanauskaitė

Atliko
ITM-11 gr. Studentas K.Visialga
2013

Šiauliai, 2013

UŽDUOTIS

Šio lapo realiai pildyti nereikia, tiesiog vietoj jo įsegsite darbo užduotį (Atsižvelkite, jei užduotis sudaryta iš 2 puslapių, kad būtų tinkamai atvaizduojama puslapių numeracija).

SANTRAUKA

Tradiciniai programinės įrangos projektų valdymo modeliai palaipsniui yra pakeičiami judriaisiais modeliais. Tai vyksta dėl šių modelio trūkumo - lėtos adaptacijos prie kintamų reikalavimų, dėl to šie projektai gali viršyti jiems skirtą biudžetą ir laiką.

Atlikus krioklio, SCRUM ir RUP modelių analizę, siūlomas naujas hibridinis modelis, kuris apjungia šių modelių privalumus. Šis modelis turėtų būti naudingas vykdant vidutinio dydžio programinės įrangos projektus, kuriuose yra tikėtinas reikalavimų kitimas.

Raktiniai žodžiai: projektas; projektų valdymo modeliai; hibridinis projektų valdymo modelis.

SUMMARY

Conventional software development methods have gradually been replaced by lightweight agile software development methods. This phenomenon is mainly due to the conventional methods' shortcomings, including a slow adaptation to rapidly changing business requirements, and a tendency to be over budget and behind schedule.

This research, analyzes Waterfall, Scrum and RUP models and suggest new hybrid model, which combines the advantages of these models. This model should be useful in medium-scale software development projects where requirements are likely to change.

Keywords: project; software development models; hybrid software development model.

TURINYS

ĮVADAS

Programinė įranga (PĮ) nuolat tobulėja, atsiranda naujos taikymo sritys, auga kodo apimtys. Todėl yra poreikis valdyti programinės įrangos kūrimo projektus, siekiant sumažinti programinės įrangos kūrimo kaštus, užtikrinti kokybę, atlikti juos laiku.

Šio darbo tikslas - išanalizavus ir ištyrus taikomus programinės įrangos projekto valdymo modelius, pasiūlyti naują, hibridinį modelį, kuris optimaliau išnaudotų išteklius.

Šiam tikslui pasiekti, išsikelti šie uždaviniai:

1. Išnagrinėti programinės įrangos gyvavimo ciklo modelius ir jų ypatumus.
2. Išnagrinėti programinės įrangos projektų valdymo ypatumus ir taikomas technologijas.
3. Išanalizuoti ir teoriškai palyginti programinės įrangos kūrimo modelius.
4. Ištirti programinės įrangos projektų valdymo modelius
5. Sudaryti naują hibridinį projekto valdymo modelį.

Darbą sudaro įvadas, analitinė ir praktinė dalys, išvados ir literatūros sąrašas. Analitinėje dalyje nagrinėjamas programinės įrangos projektų valdymo problematika ir šių projektų valdymui taikomi modeliai, analizuojami jų privalumai, trūkumai, panašumai.

Praktinėje dalyje teoriškai ištiriami krioklio, SCRUM ir RUP modeliai ir pagal gautus tyrimo rezultatus bei pastebėjimus sudaromas naujas, hibridinis modelis. Šis modelis ištiriamas teoriškai. Taip pat aprašomas tyrimas ir jo rezultatai apie projektų valdymo modelių taikymą praktikoje.

Praktinis ir mokslinis naujumas

Darbe pasiūlytas hibridinis projektų valdymo modelis, leidžiantis optimaliau išnaudoti laiko išteklius vidutinio dydžio programinės įrangos kūrimui, nei tai įgalina krioklio, RUP ir SCRUM modeliai.

Darbo aprobacija

2012 m. Šiaulių universiteto Technologijos fakulteto 7-oji konferencija "Studentų moksliniai darbai";

2013 m. Šiaulių universiteto Technologijos fakulteto 8-oji konferencija "Studentų moksliniai darbai";

1. PROGRAMINĖS ĮRANGOS PROJEKTŲ VALDYMO MODELIŲ ANALIZĖ

1. 1 Projekto sąvoka

Pagal Projektų valdymo institutą (Project Management Institute), projektas, tai laikina veikla, siekiant sukurti unikalų produktą, paslaugą ar rezultatą. Projektą galima laikyti apibrėžtu darbu, kurį reikia atlikti laiku, kokybiškai ir neviršijant biudžeto. Apibendrinsime projekto bendrąsias charakteristikas[3]:

- Laikina veikla, nes projektas turi būti įvykdytas per apibrėžtą laiką, projektas laikomas baigtu, kai pasiekti jo tikslai[3].
- Unikalumas yra svarbi projekto charakteristika. Pavyzdžiui, yra pastatyta daug biurams skirtų pastatų, tačiau kiekvienas individualus pastatas yra unikalus - skirtingas savininkas, skirtingas dizainas, skirtinga vieta, skirtingi rangovai, ir t.t.. Pasikartojantys elementai nekeičia fundamentalaus projekto darbo unikalumo[3].
- Palaipsninis plėtojimas – tai kūrimas etapais. Projekto pradžioje, projekto sritis bus apibrėžta labai bendromis sąvokomis ir taps detalesnė projekto eigoje[3].

Projekto valdymas – tai veikla grindžiama žiniomis, patirtimi, modeliais, priemonėmis, technologijomis skirta projekto tikslui (tikslams) pasiekti. Projektų valdymo modelių poreikis atsirado didėjant projektų apimtims, jiems tampant sudėtingesniais ir griežtėjant reikalavimams. Projekto valdymo būtinumą apsprendžia šie veiksniai[1]:

- projekto apimtys ir kaštai;
- projekto sudėtingumas;
- reikalavimai projekto terminams, biudžetui, resursams ir kokybei;
- pokyčių projekte ir išorėje tikimybė;
- projekto dalyvių skaičius ir tarpusavio ryšiai;
- organizacijos (firmos) dydis ir struktūra;
- konkurentai;
- projekto prestižas ir t.t.

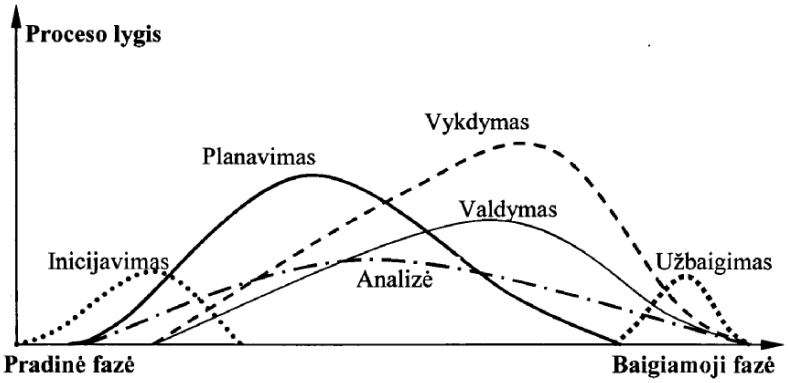
1. 2 Projektų procesai

Projekto valdymas - tai veikla grindžiama žiniomis, patirtimi, metodais, priemonėmis, technologijomis ir skirta projekto tikslui (tikslams) pasiekti. Tai „projektų integravimo į organizacijos veiklą ir projektų rezultatų valdymas, laiko, sąnaudų, kokybės, personalo, ryšių, rizikos ir pirkimų valdymas“ [3].

Projektų valdymo vadove (PMBOK) išskiriami šie projekto valdymo procesai :

- Inicijavimas. Šio etapo metu generuojamos ir atrenkamos idėjos, pradedamas projektas. Svarstoma laukiama nauda, mėginama nustatyti, ar verta pradėti projektą, atsižvelgiant į organizacijos ir suinteresuotųjų šalių interesus, suformuojama projekto komanda, kuriai pavedama įgyvendinti kitą etapą – suplanuoti projektą.
- Planavimas. Rengiamas projekto planas, leidžiantis pasiekti projekto tikslus, įvertinamas projekto įgyvendinamumas, jo rizikos, sąveika su kitais organizacijos projektais, reikiami ištekliai, apibrėžiami projekto dalyvių vaidmenys ir atsakomybės, darbų apimtis, sudaromas tvarkaraštis, nustatomas biudžetas.
- Vykdymas. Vykdomas projekto planas, valdomi darbuotojų ir kiti ištekliai, kuriamas produktas.
- Analizė. Šio etapo metu stebima ir vertinama projekto įgyvendinimo eiga, matuojama projekto pažanga tikslo atžvilgiu, fiksuojami dėl pakeitimų atsiradę projekto nuokrypiai.
- Valdymas. Nustatomos ir vykdomos projektą koreguojančios veiklos, šalinami neigiamai įtakojantys veiksniai.
- Užbaigimas. Įsitikinama, kad visi planuoti darbai atlikti ir laukiami rezultatai pasiekti, įgyvendintas projekto tikslas. Taip pat siekiama gauti projekto suinteresuotųjų šalių gauto tinkamo galutinio rezultato patvirtinimą. Formaliai užbaigiamas projektas, įvertinami projekto rezultatai.

Atvaizdavus šiuos projekto valdymo procesus laike, gautume grafiką, kuris pavaizduotas 1 paveikslėlyje[20]:



1 pav. *Projekto valdymo procesai[21]*

Vykdant projektą šie procesai vykdomi paeiliui, tačiau šie projekto valdymo procesai yra persidengę ir integruojasi vienas į kitą. Galime pastebėti, kad projekto procesai prasideda praktiškai vienu metu (išskyrus užbaigimą), tik yra skirtingas jų vykdymo laikas. Procesai glaudžiai tarpusavyje susiję, o vieno proceso produktas dažnai tampa duomenimis, reikalingais kitam procesui prasidėti.

1. 3 Programinės įrangos projektų problematika

Programinės įrangos projektai nuo kitų sričių projektų išsiskiria dėl vien programinei įrangai būdingų savybių. Pagrindines PĮ projektų charakteristikos[21]:

- Nematomumas. Programinė įranga yra neapčiuopiama. Programos struktūra nėra akivaizdi, matomos tik jos galutinio veikimo pasekmės.
- Sudėtingumas. PĮ sprendžia labai sudėtingus uždavinius ir problemas. PĮ gali būti tokia sudėtinga, kad programuotojas gali nesuprasti visos sistemos veikimo ir koreguodamas vieną jos dalį, gali pakenkti kitos veikimui.
- Lankstumas. PĮ veikimą galima nesunkiai keisti iš esmės. Todėl užsakovai kartais paprašo atlikti pakeitimus likus mažai laiko iki projekto pabaigos. Šie pakeitimai labai rizikingi, nes gali turėti rimtų pasekmių.
- Sunkiai įvertinama kaina. IT projektus vertinti gali tik didelę patirtį turintys ekspertai. Prieinamų šaltinių, kur tokie ekspertai dalintųsi savo žiniomis nėra daug.
- Testavimo ypatumai. Testavimą apsunkina PĮ sudėtingumas, besikeičiantis veikimas, painios sąsajos su išorinėmis sistemomis.

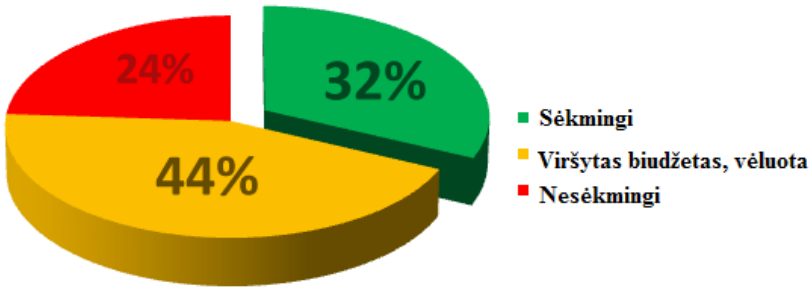
Šios PĮ projektų savybės daro juos unikaliais. Kitaip, nei kitose srityse, reikalavimai programinės įrangos funkcionalumui jos kūrimo metu gali kisti, todėl šių projektų valdymui reikalingi atitinkami metodai, kurie leistų projektą įvykdyti iki galo. Taip pat šie projektai išsiskiria tuo, kad lyginant su kitomis sritimis, čia projekto žlugimo tikimybė palyginti yra didelė.

Programinės įrangos projektų valdymas leidžia sumažinti nesėkmės tikimybę, numatyti ir planuoti biudžetą, sumažinti projekto išlaidas, sutrumpinti projekto trukmę, pagerinti programinės įrangos kokybę ir palengvinti jos palaikymą. Tai padidina programinės įrangos verslu užsiimančios įmonės pelną, padeda lengviau suformuluoti užduotis darbuotojams, kas padidina produktyvumą, padidina užsakovo pasitenkinimą gauta paslauga ar sukurtu produktu. Vidutinės ir didesnės apimties projektai (kai PĮ sudaro 5-50 ir daugiau tūkst. kodo eilučių[1]) gali tapti neįmanomi jeigu nepasitelkiami PĮ projektų valdymo modeliai. Vien šių modelių panaudojimas nereiškia, kad projektas bus įvykdytas sėkmingai, taip pat reikia ir kompetentingų specialistų valdant ir vykdant projektą. [2, 6, 9].

Yra atlikta daug tyrimų bandant įvertinti nesėkmingų PĮ projektų kiekį.

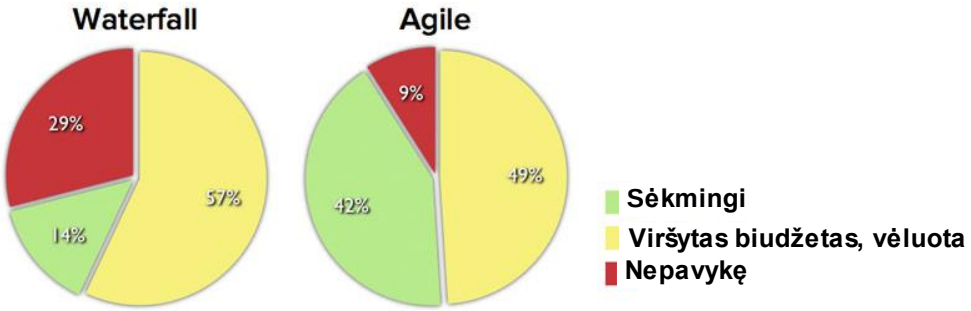
Pagal Standish Group Chaos atliktą tyrimą 2009 metais (žr. 2 pav.)[20]:

- 32% PĮ projektai buvo įvykdyti laiku neviršijant biudžeto (sėkmingai);
- 24% projektų nutraukti;
- 44% projektų būna atlikti viršijus numatytą biudžetą, ne laiku, nepilnai įgyvendinus numatytas PĮ funkcijas.



2 pav. *Sėkmingų ir nesėkmingų projektų pasiskirstymas[20].*

Ši organizacija taip pat atliko tyrimą, kurio metu tyrinėjo sėkmingų ir nesėkmingų projektų kiekį pagal taikytus modelius. Pagal šio tyrimo rezultatus, taikant vikriuosius programinės įrangos projektų valdymo modelius, 20% mažiau projektų žlunga, lyginant su krioklio programinės įrangos gyvavimo ciklo modelį taikiusiais projektais (žr. 3 pav.).



3 pav. *Sėkmingų ir nesėkmingų projektų pasiskirstymas pagal taikytus modelius [20]*

Tata Consultancy duomenimis 2007 metais[23]:

- 62% organizacijų IT projektai nesilaikė tvarkaraščių;
- 49% viršijo biudžetą;
- 47% buvo didesnės nei tikėtasi priežiūros išlaidos;
- 41% nesugebėjo pasiekti laukiamos vertės ir investicijų grąžos;

Apibendrinant, galima teigti, kad tik apie 30% PĮ projektų įvykdomi sėkmingai, 50% nespėja su nustatytais terminais, viršija biudžetą arba neišpildo visų funkcinių reikalavimų, apie 20% projektų patiria nesėkmę.

Pagrindinės IT projektų problemų priežastys[20]:

- Suinteresuotųjų šalių valdymo problema.
- Valdymo problema.
- Komandos valdymo problema.
- Komunikacijos valdymo problema.
- Išteklių valdymo problema.
- Rizikos valdymo problema.
- Pirkimų valdymas.

Tik sprendžiant visas aukščiau išdėstytas problemas galima padidinti tikimybę, kad PĮ projektas bus įvykdytas sėkmingai.

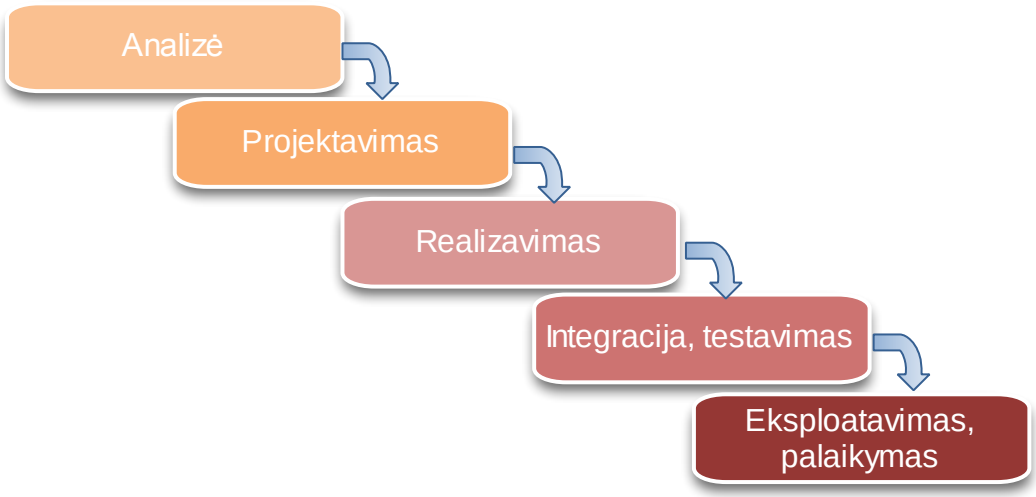
Šiame darbe bus nagrinėjami programinės įrangos gyvavimo ciklo modeliai (programinės įrangos projektų valdymo modeliai), kurie apima komandos, rizikos ir išteklių valdymo problematiką.

1. 4 Programinės įrangos projektų valdymo modeliai

1. 4. 1 Krioklio modelio analizė

Krioklio (dar vadinamo etapiniu, tradiciniu) PĮ gyvavimo ciklo modelis (žr. 4 pav.) pasižymi paprastumu ir taikymo aiškumu. Visi projekto etapai vykdomi nuosekliai, prie sekančio etapo pereinama tik visiškai pabaigus ankstesnį, nenumatomas atgalinis ryšys. Šis modelis susideda iš šių etapų[1]:

1. Analizė - surenkami užsakovo reikalavimai kuriami PĮ;
2. Projektavimas - suprojektuojama PĮ architektūra, numatoma kaip ji bus įgyvendinama, nusakomi ryšiai tarp komponentų, PĮ struktūra ir sudaromas realizavimo etapo planas;
3. Realizavimas- Toliau PĮ suprogramuojama, sukuriamas produktas, kurį būtų galima testuoti.
4. Integracija ir sistemos testavimas – atliekamas integracinis (patikrinti PĮ sistemą kaip visumą) ir priėmimo (tikrinama, ar PĮ tinkamai atlieka reikalavimuose numatytas užduotis, ar programinė įranga tinkama perduoti užsakovui).
5. Eksploatavimas ir palaikymas – apima klaidų taisymą, programinės įrangos tobulinimą bei pritaikymą prie naujos aplinkos.



4 pav. *Krioklio modelis[14]*

Krioklio modelio privalumai:

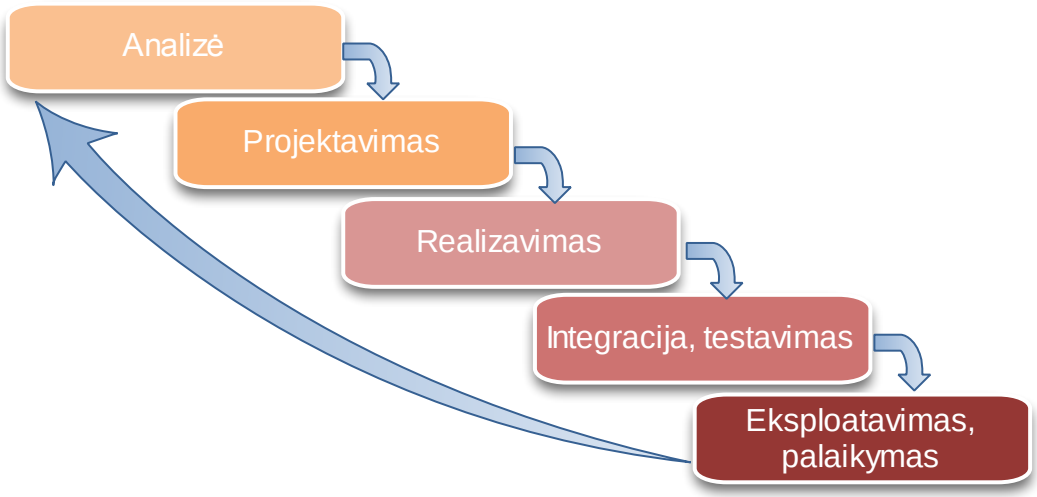
- aiškus ir paprastas taikymas;
- visiškai apibrėžus reikalavimus ir gerai suprojektavus PĮ architektūrą, PĮ realizavimas būna sklandus, kiekvieno etapo metu sukuriama dokumentacija yra tvirtas pamatas sekančiam etapui.

Šio modelio trūkumai:

- netinkamas, kai gali kisti reikalavimai;
- ribotas bendravimas su užsakovu, blogai ar nepilnai užsakovo nusakyti reikalavimai sąlygoja produktą, kuris neatitiks jo lūkesčių;
- kuriamo produkto prototipas egzistuoja tik paskutiniuose programinės įrangos kūrimo etapuose, todėl užsakovo nuomonė apie kuriamą produktą galima tik vėlyvose projekto stadijose.

1. 4. 2 Iteracinio modelio analizė

Iteracinio gyvavimo ciklo modelio esmė yra PĮ kūrimas iteracijomis (žr. 5 pav.) – besikartojančiais ciklais, kurie apima dalį arba visus krioklio modelio etapus. Šis modelis atsirado dėl krioklio modelio trūkumo – negalėjimo prisitaikyti prie besikeičiančių ar papildomų reikalavimų. Iteracijos yra pagrindinė iteratyvių PĮ projektų valdymo modelių dalis.



5 pav. *Iteracinis modelis*

Iteracinio modelio privalumai:

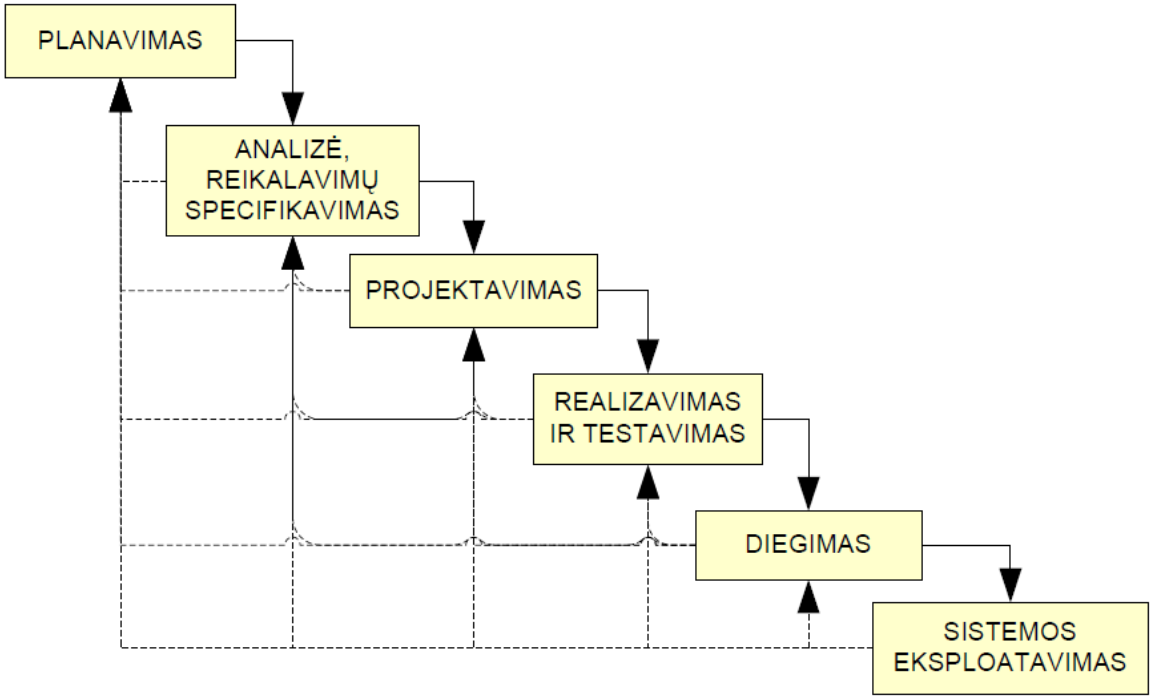
- aiškus ir paprastas taikymas;
- kiekvieno etapo metu sukuriama dokumentacija yra tvirtas pamatas sekančiam etapui;
- tinkamas taikyti, kai kinta užsakovo reikalavimai kuriamam produktui.

Iteracinio modelio trūkumai:

- dažnai vykdant pirmąsias iteracijas nesurenkami esminiai reikalavimai;
- sunku įvertinti projekto trukmę ir biudžetą.

1. 4. 3 Fontano modelio analizė

Fontano modelis - tai krioklio ir Iteracinio modelių modifikacija. Kaip ir krioklio modelyje čia iš eilės vykdomi visi PĮ kūrimo etapai, tačiau kitaip nei Iteraciniame, iš kiekvieno etapo egzistuoja grįžtamasis ryšys į bet kurį prieš tai buvusį etapą (žr. 6 pav.) – programinės įrangos kūrimas čia suprantamas kaip begalinis iteratyvus procesas.



6 pav. *Fontano modelis[32]*

Fontano modelio privalumai:

- aiškus ir paprastas taikymas;
- kiekvieno etapo metu sukuriama dokumentacija yra tvirtas pamatas sekančiam etapui;
- tinkamas taikyti, kai kinta užsakovo reikalavimai kuriamam produktui.

Fontano modelio trūkumai:

- dažnai vykdant pirmąsias iteracijas nesurenkami esminiai reikalavimai;
- galimybė grįžti į bet kurį ankstesnį etapą sąlygoja nepilnai ir netiksliai įgyvendintus ankstesniuosius;
- sunku įvertinti projekto trukmę ir biudžetą;
- projektas gali užstrigti nuolatiniame cikle.

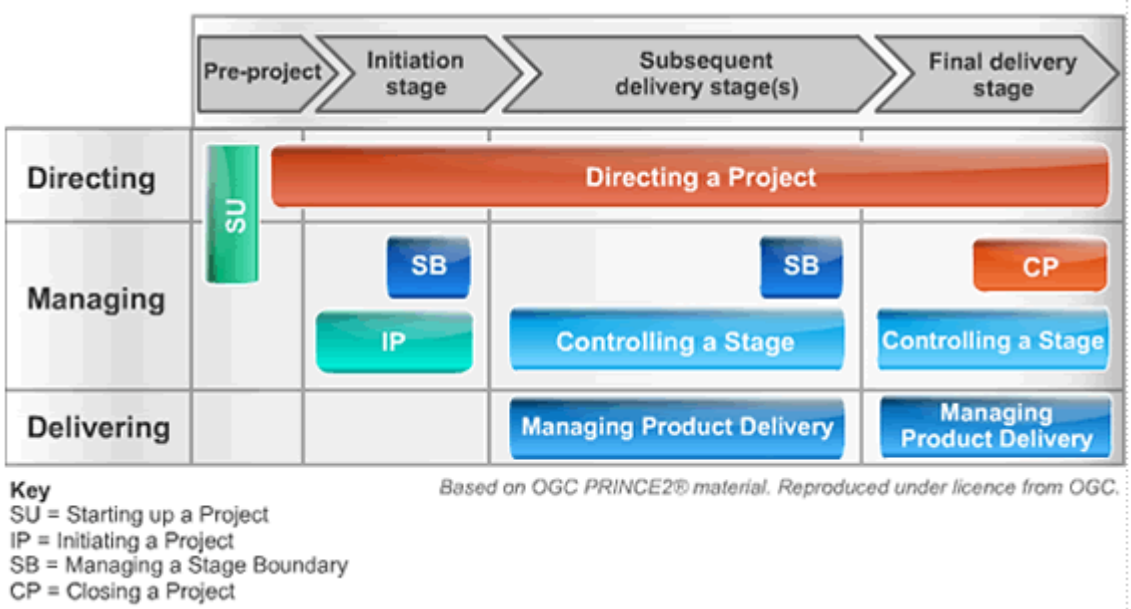
1. 4. 4 PRINCE2 modelio apžvalga

PRINCE2 (angl. PRojects IN Controlled Environments) tai procesais pagrįstas modelis, skirtas efektyviam projektų valdymui. Šis modelis yra bendros paskirties, taikytinas nepriklausomai nuo projekto mąsto, srities, organizacijos ar kultūros. Modelio taikymas aprašomas procesais, kurie vyksta šiuose etapuose:

1. Prieš projektinis;
2. Inicijavimas;
3. Tarpinis pridavimo etapas;
4. Galutinis pridavimo etapas.

Šį modelį sudaro 4 elementai[16]:

1. Principai- vadovaujamasi įsipareigojimas ir gera praktika. Yra 7 principai (tęstinumas versle, mokytis iš patirties, nusakyti roles ir atsakomybes, valdyti etapais, valdyti išimtimis, koncentruotis produktu ir pritaikyti projekto aplinkai).Jie turi būti pritaikyti, kitaip tai nebus PRICE2 projektas;
2. Objektai (ang. Themes) – apibūdina projekto valdymo aspektus, kurie turi būti vykdomi nuolat ir lygiagrečiai. 7 objektai (verslo planas, organizavimas, kokybė, planai, rizika, pokyčiai ir progresas) apibūdina specifinį elgesį reikalinga PRINCE2 įvairiuose projekto valdymo disciplinose ir kodėl jos yra reikalingos.
3. Procesai – apibūdina žingsnius, vykdomus projekto vykdymo metu. Kiekvienas procesas turi kontrolinį sąrašą sudarytą iš veiksmų, produktų ir atsakomybių. Yra 7 procesai.
4. Projekto aplinka – pritaikymas specifiniai sričiai ar projektui.



7 pav. *PRINCE2 modelis [16]*

PRINCE2 modelį sudaro šie procesai (žr. 7 pav.)[16]:

1. Vadovavimas projektui (angl. Directing a Project) – vadovavimas projektui pradėdamas nuo jo pradžios ir vykstomas iki jo pabaigos. Šį procesą vykdo projekto valdyba (angl. Project Board).
2. Projekto pradžia (angl. Starting up a Project) – tai pirminis procesas. Jis skirtas įsitikinti, ar viskas, kas reikalinga yra savo vietose, kad būtų galima pradėti projektą.

3. Inicijuojamas projektas (angl. Initiating a Project) – šio proceso tikslai yra įsitikinti, ar yra pakankamas pagrindas vykdyti projektą, nustatyti pradžiai tvirtą valdymo pagrindą, nustatyti ar yra pakankamai lėšų pradėti projektą, ar įvertintos rizikos ir kt.
4. Etapų ribų nustatymas (angl. Managing Stage Boundaries) – šiame procese projekto valdyba numato, kada verta tęsti projektą ir kada reikėtų jį nutraukti. Įsitikinama, kad visi siekiai, kurie buvo suplanuoti, yra įgyvendinti kaip ir numatyta, projekto valdybai suteikiama informacija kaip identifikuoti projekto eigą ir etapo užbaigtumą.
5. Etapo kontrolė (angl. Controlling a Stage) – šis procesas apibudina projekto valdymo veiklų stebėjimą ir kontrolę. Įsitikinama, kad neatsiliekama nuo plano, reaguojama į netikėtus įvykius. Per visą etapą vykdomos šios veiklos:

- Pasirūpinti, kad darbai būtų atlikti;
- Rinkti informaciją apie darbo progresą;
- Stebėti pokyčius;
- Peržiūrėti esamą būseną;
- Raportuoti;
- Imtis visų būtinų korekcijos veiksmų.

6. Produkto pristatymo valdymas (angl. Managing Product Delivery) – šio proceso tikslai: įsitikinti ar darbas yra pabaigtas, ar sukurtas produktas atitinka kokybės reikalavimus.
7. Projekto užbaigimas (angl. Closing a Project) – projekto valdybai suteikiama informacija apie atliktus darbus, kad ši galėtų patvirtinti, kad projektas gali būti pabaigtas. Parašomos rekomendacijos sekantiems projektams, paruošiamas projekto pabaigos raportas.

Rekomenduojama naudojant PRINCE2 modelį taikyti 3 lygių planavimą, kad būtų atspindimi skirtingų valdymo lygių poreikiai: projekto, etapo ir komandos[16].

Planavimas tai pasikartojantis procesas. Jo veiklos įeina į PRINCE2 septynis pagrindinius procesus. Planavimo veiklos yra šios[16]:

- Sukuriamas planas;
- Apibūdinami ir analizuojami produktai;
- Identifikuojamos veiklos ir priklausomybės;
- Paruošiamas įvertinimas;
- Paruošiamas grafikas;
- Analizuojamos rizikos;
- Dokumentuojamas planas.

PRINCE2 naudoja metodą žinomą kaip „Planavimas remiantis produktu“ (angl. Product based planning) kuriam reikalingos 4 veiklos[16]:

- Aprašomas projekto kuriamo produkto apibūdinimas;
- Sukuriama produkto paskirstyta struktūra;
- Aprašomas produkto apibūdinimas;
- Sukuriama produkto srautų diagrama.

Šios 4 veiklos yra atliekamos vietoj „Apibūdinami ir analizuojami produktai“ veiklos paminėtos aukščiau.

PRINCE2 modelio privalumai:

- palaikomas tokių organizacijų kaip APM, ISEB.
- palaikomas valdžios institucijų (Jungtinėje Karalystėje).
- gali būti taikomas bet kokioje organizacijoje bet kokios apimties projektui.

Trūkumai:

- sunkiai įsisavinimas;
- norint išnaudoti šio modelio privalumus (valdžios institucijų pripažinimą ir kt.) reikalingi sertifikatai;
- gali užtrukti daug laiko, kol šio modelio taikymas bus įdiegtas visoje organizacijoje;
- modelis orientuotas į biurokratiją, sukuriamas didelis dokumentų ir ataskaitų kiekis;

- praktikoje šis modelis netaikytinas projektams, kurių metu sunku nusakyti būsimo produkto savybes.

1. 4. 5 Rational Unified Process (RUP) modelio analizė

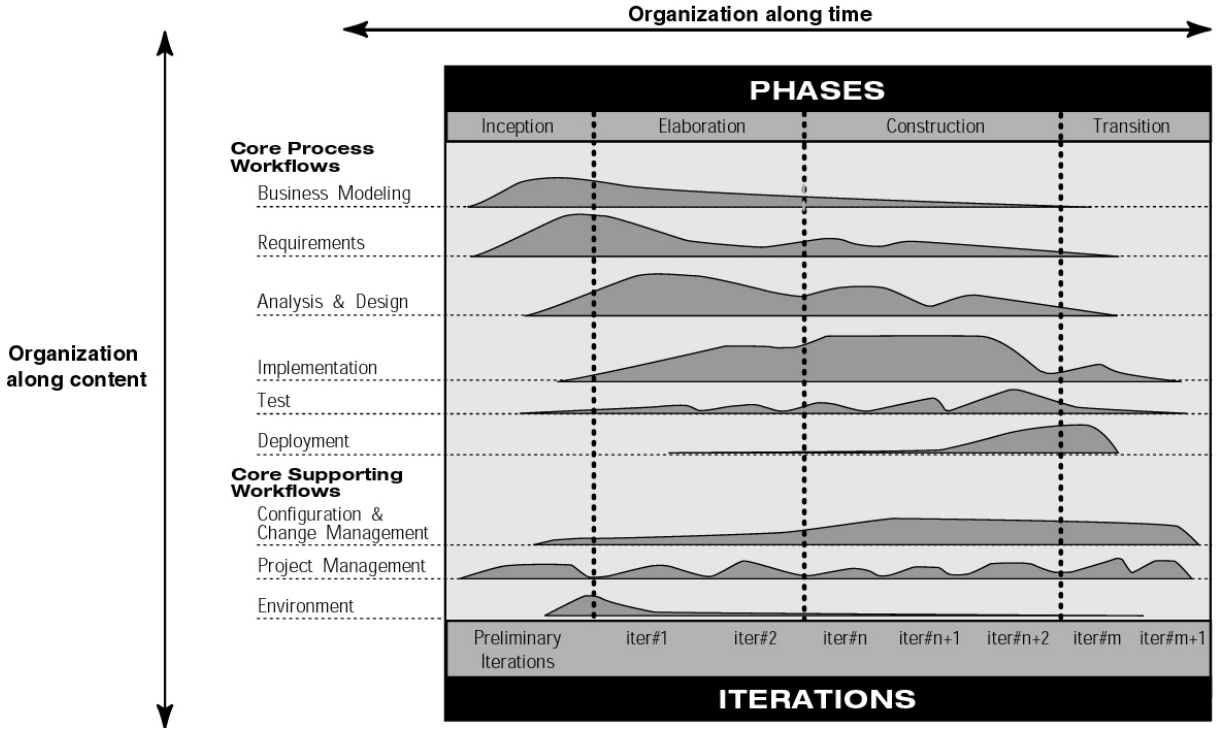
RUP (angl. Rational Unified Process) - programinės įrangos kūrimo modelis, sukurtas įmonės Rational Software, priklausančios IBM [4].

Yra šeši RUP raktiniai principai[13,8]:

1. Kurti programinę įrangą iteracijomis – taip rimti nesusipratimai yra nustatomi anksčiau, o reagavimas į juos dar nėra toks brangus. Tai paskatina vartotojo įtraukimą, padeda detaliau išsiaiškinti realius sistemos reikalavimus. Pakartotinai testuojant objektyviau įvertinama projekto būsena. Tolygiau paskirstomas projekto komandos apkrovimas. Projekto komanda gali mokintis iš praeitų iteracijų ir gerinti procesą.
2. Valdyti reikalavimus - reikalavimų valdymas tampa disciplinuota veikla. Reikalavimams suteikiami prioritetai, jie filtruojami ir sekami. Lengviau nustatomi netikslumai.
3. Naudoti komponentais pagrįstas architektūras – jos yra lankstesnės, moduliškumas leidžia lengviau paskirstyti darbus, efektyviau atlikti modulinį testavimą, atskirti elementus, kurie gali keistis. Skatina pakartotinį panaudojimą, remiantis standartizuotais karkasais (angl. frameworks) ir komerciniais komponentais.
4. Vizualiai modeliuoti programinę įrangą - panaudos atvejai ir scenarijai tiksliau nusako veikimą. Vizualūs modeliai labai patogūs analizavimui ir leidžia atrasti prastai suprojektuotas ir nelanksčias architektūros vietas. Yra daug gerų įrankių palaikančių UML modeliavimą.
5. Tikrinti programinės įrangos kokybę - projekto būsenos įvertinimas parodo nesuderinamumus reikalavimuose, projektavime ir realizacijoje. Anksčiau nustatyti defektai žymiai sumažina jų ištaisymo kainą.
6. Kontroliuoti programinės įrangos pasikeitimus - gali būti įvertinama pakeitimų įtaka. Pakeitimų kontroliavimui gali būti naudojamos patikimos, pritaikomos sistemos. Pakeitimų prašymų procedūra skatina aiškesnę komunikaciją.

Procesas gali būti apibūdintas dvejomis dimensijomis (žr. 8 pav.)[13, 5]:

- Horizontali ašis simbolizuoja laiką ir parodo dinaminę vykstančio proceso pusę ir išreiškiama ciklais, fazėmis, iteracijomis ir etapais.
- Vertikali ašis simbolizuoja statinę proceso pusę: veiklos, artefaktai (informacijos vienetas, sukuriamas, modifikuojamas arba naudojamas programinės įrangos kūrimo procese), darbuotojai, darbų sekos (angl. workflow).



8 pav. *Proceso struktūra per dvi dimensijas[13]*

RUP turi 4 fazes (dinaminė proceso pusė)[13]:

1. Pradžios fazė (angl. Inception) – šiame etape suprantama kokią PĮ norima sukurti, identifikuojamas sistemos funkcionalumas, nustatomas bent vienas galimas sprendimas, įvertinamas sistemos biudžetas, terminai, rizikos. Nusprendžiama kokius įrankius naudoti. Baigiama gyvavimo ciklo tikslų gaire.
2. Vystymo fazė (angl. Elaboration) - čia detalizuojami reikalavimai: suprojektuojama, realizuojama, patikrinama ir apibrėžiama architektūra. Įvertinamos rizikos ir tiksliau įvertinami tvarkaraščiai ir biudžetas. Išbandomas realizacijos procesas, jis patobulinamas ir paruošiama realizacijos technologijų aplinkai.
3. Konstravimo fazė (angl. Construction) – sumažinami realizacijos kaštai, darbai paskirstomi lygiagrečiai komandos nariams. Iteracijomis sukuriamas produktas, kurį būtų galima perduoti vartotojams. Pabaigiama operacinių galimybių gaire .
4. Perdavimo fazė (angl. Transition) – atliekamas beta testavimas, apmokomi vartotojai, paruošiama darbinė aplinka, pasiruošiama marketingui, paskirstymui ir pardavimui. Siekiama gauti užsakovo sutikimą, kad programinė įranga atitinka tikslus. Pabaiga – produkto išleidimo gairė.

Kiekviena RUP fazė procese gali būti papildomai suskaidyta į iteracijas. Lyginant su krioklio procesu, iteratyvus procesas turi šias teigiamas savybes[13]:

- Anksčiau sumažinamos rizikos;
- Lengviau valdomi pokyčiai;
- PĮ labiau tinka pakartotiniam panaudojimui (angl. reuse);
- Projekto komanda, vykstant šiam procesui, gali apsimokyti;
- Geresnė PĮ kokybė visais atžvilgiais.

Statinė proceso struktūra apibrėžia kas atlieka ką, kaip ir kada. RUP yra sudarytas iš 4 pirminių modeliavimo elementų[13]:

- Darbuotojų – kas;
- Veiklos – kaip;
- Artefaktų – ką;
- Darbų sekos –kada.

Aukščiau paminėti elementai naudojami modeliuoti darbų sekas (angl. workflow) UML diagramomis. Darbų seka (angl. workflow) yra veiklų rinkinys, kuris duoda apčiuopiamą rezultatą. RUP apibrėžia 6 inžinerines darbų sekas[13]:

1. Verslo modeliavimas
2. Reikalavimų apibrėžimas
3. Analizė ir projektavimas
4. Realizacija
5. Testavimas
6. Įdiegimas

Taip pat RUP apibrėžia 3 pagalbines darbų sekas[13]:

1. Projekto valdymas
2. Konfigūracijos ir pasikeitimų valdymas
3. Aplinkos formavimas

RUP modelio privalumai:

- pilnai ir plačiai dokumentuota metodologija;
- išskiriamos disciplinos rizikos valdymui;
- tinkamas, kai gali kisti reikalavimai projekto eigoje;
- greitesnis produkto kūrimas, kai pakartotinai naudojami komponentai;
- gausus metodinės medžiagos kiekis apie šio modelio taikymo principus.

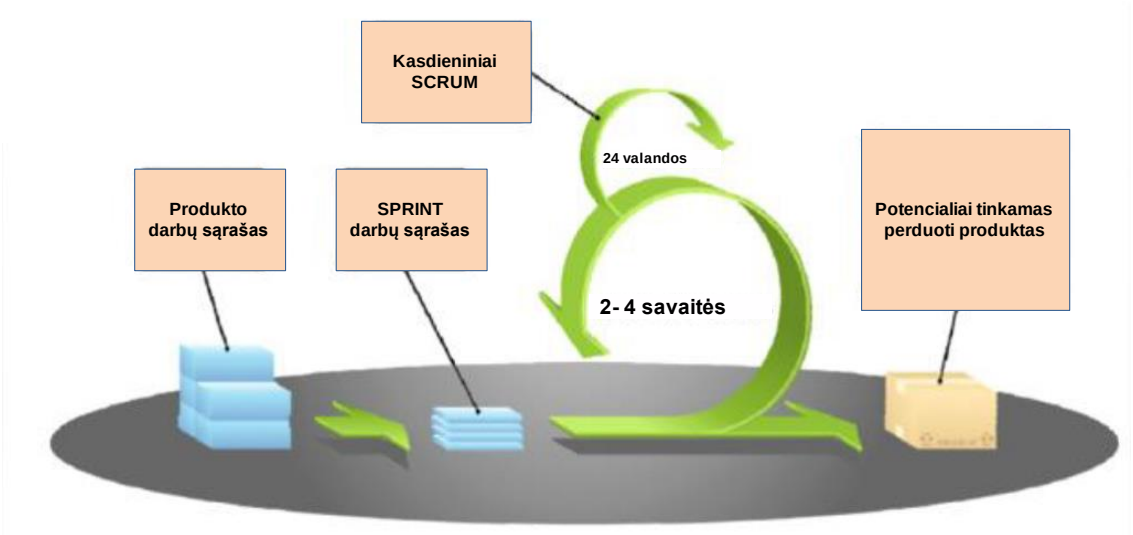
RUP modelio trūkumai:

- komandos nariai turi puikiai išmanyti savo sritį;
- modelio taikymas yra painus ir komplikuotas;
- projektuose, kuriuose naudojamos naujos technologijos, pakartotinis komponentų panaudojimas gali būti neįmanomas, todėl išauga projekto trukmė ir biudžetas;
- taikant RUP modelį, projekto eigoje atsiranda didelis dokumentacijos kiekis.

1. 4. 6 Scrum modelio analizė

Scrum yra lanksčiojo programavimo projektų valdymo modelis. Šio modelio gyvavimo ciklas susidaro iš keturių etapų (žr. 9 pav.)[12]:

1. Planavimo - aprašoma vizija, pradinis darbų sąrašas (angl. backlog), sukuriamas pradinis modelis ir prototipas.
2. Konstrukcijos – planavimas, sukuriamas pradinis modelis ir prototipas.
3. Kūrimo – sprintais kuriama PĮ. Vyksta savaitiniai iteracijų planavimo susirinkimai, kasdieniai Scrum susirinkimai, iteracijų demonstravimas.
4. Pridavimo – įdiegiama veikianti PĮ. Rašoma dokumentacija, apmokomi vartotojai.



9 pav. *Scrum modelis[10]*

Scrum modeliui būdinga[12]:

- Trumpomis iteracijomis užbaigiami dideli darbai;
- kasdieniai scrumai – trumpi 15 min. susirinkimai, kurių metu kiekvienas grupės narys pasako ką per dieną nuveikė ir ką planuoja daryti toliau;
- planavimo sesijos, kurių metu sudaromos užduotys sekančiam sprintui;
- atliktų darbų pristatymas, per kurį visi komandos nariai apžvelgia praėjusią iteraciją;
- Scrum vedlys, kuris yra iteracijų planuotojas. Jo pirminis darbas – padėti komandai įvykdyti iteracijų užduotis ir pašalinti visus iškylančius trukdžius.

Scrum modelio praktikos[12]:

- klientas tampa programinės įrangos kūrimo komandos dalimi;
- kaip ir kiti lanksčiojo programavimo PĮ kūrimo procesai, Scrum turi tarpinius, klientui rodomus sukurtos PĮ pristatymus;
- rizikos planai, kontrolė ir valdymas (rizikų analizė) yra visose projekto etapuose;
- planavimo ir produkto kūrimo aiškumas – kiekvienas žino, už ką atsakingas ir iki kada turi atlikti pavestas užduotis;
- dažni tarpiniai komandos narių susitikimai padeda matyti projekto progresą, anksti numatyti vėlavimus ir nuokrypius nuo tvarkaraščio;
- nei viena problema nėra nuslepiama ar paliekama neišspręsta.

Scrum modelis skatina savarankiškumą ir betarpišką komandos narių bendravimą. Šis modelis akcentuoja anksti sukurtą programinę įrangą ir mažą dokumentacijos kiekį arba visišką jos nebuvimą.

SCRUM modelio privalumai:

- padeda taupyti laiką ir pinigus;
- gerai sudarius darbų sąrašą ir gerai nustačius užduotims skiriamą laiką, galima gana tiksliai numatyti projekto įvykdymo datą, nustatyti kada projektas gali vėluoti;
- tinkamas, kai gali kisti reikalavimai projekto eigoje;
- aiški esama projekto būsena;
- nuolat žinoma užsakovo nuomonė apie kuriamą produktą;
- galima racionaliau paskirstyti darbus ir taip pasiekti didesnį komandos produktyvumą;
- lengviau pristatyti kokybišką produktą laiku.

SCRUM modelio trūkumai:

- netiksliai esminiai reikalavimai įtakoja netikslią projekto pabaigos datą ir biudžetą;
- komandos nariai turi būti motyvuoti ir atsidavę darbui;
- labiau tinkamas mažiems dinamiškiems projektams, kai juos vykdo mažos komandos;
- komandos nariams reikia tūrėti patirties dirbant pagal šį modelį;
- stipri individuali priklausomybė gali įtakoti projekto baigtį.

1. 4. 7 Analizuotų modelių palyginimas

Nagrinėtuose modeliuose skiriasi etapų ir fazių pavadinimai, bet veiklos jų metu yra panašios. Žemiau pateiktoje lentelėje pateikiamas modelių etapų ir fazių palyginimas suklasifikuojant juos į planavimą ir reikalavimų surinkimą, architektūros specifikavimą, programavimą, testavimą, perdavimą vartotojams (palaikymą).

1 lentelė. Fazių ir etapų palyginimas

Apibendrinti projekto etapai	**Krioklio, Fontano modelis**	**SCRUM**	**RUP**		**PRINCE2**
			Fazės	**Darbų sekos**	
Planavimas ir reikalavimų surinkimas	Analizė	Planavimas	Pradžios	Reikalavimu apibrėžimas; Projekto valdymas; Konfigūracijos ir pasikeitimu valdymas;	Prieš projektinis Inicijavimas;

				Aplinkos formavimas.	
Architektūros specifikavimas	Projektavimas		Vystymas	Verslo modeliavimas Analizė ir projektavimas	Tarpinis perdavimo etapas
Programavimas	Realizavimas	Konstrukcijos etapas Kūrimas	Konstravimas	Realizacija	Tarpinis perdavimo etapas
Testavimas	Integracija ir sistemos testavimas	Konstrukcijos etapas Kūrimas	Konstravimas	Testavimas	Tarpinis perdavimo etapas
Perdavimas	Eksploatavimas ir palaikymas	Pridavimas	Perdavimas	Įdiegimas	Galutinis perdavimo etapas

Iš 1 lentelės matyti, kad dauguma etapų nagrinėtuose modeliuose pagal juose vykdomas veiklos yra panašūs. RUP modelyje šiems etapams būtų galima priskirti ir fazes ir darbų sekas, nes skirtingose fazėse dominuoja skirtingos darbų sekos.

2 lentelė. Modelių palyginimas.

Savybės		**SCRUM**	**RUP**	**Krioklio modelis**	**Fontano, Iteracinis modelis**	**PRINCE2**
Projekto dydis	Mažas	+	+	+	+	+
	Vidutinis	+	+	+	+	+
	Didelis	+	+	+	-	+
Kūrimas iteracijomis		+	+	-	+	-
Komandos narių skaičius		5-9	7-∞	1-∞	1-∞	1-∞
Kokybės valdymas		+	+	+	+	+
Rizikos valdymas		+	+	+	+	+
Išlaidų valdymas		+	+	+	+	+
Projekto metu sukuriam detali dokumentacija, ataskaitos		-	+	+	+	+
Paprastas taikymas		+	-	+	+	-
Būtini pilnai apibrėžti visi reikalavai projekto pradžioje		-	-	+	-	+
Prototipo buvimas projekto eigoje		+	+	-	+	+
Užsakovas dalyvauja projekto eigoje		+	+	-	-	+
Etapai (fazės)		4	4	5	5	4
Iteracijos (ciklai)		1-∞	8-∞	1	1-∞	1

Visi šie nagrinėti modeliai pagal literatūroje aprašytą taikymą, išskyrus Iteracinį ir Fontano, teoriškai yra tinkami įvairaus dydžio projektams. Tačiau praktikoje SCRUM nėra taikytinas dideliems projektams, nes dėl stiprios priklausomybės nuo komandos narių, yra rizika, kad projektas gali būti neįvykdytas.

Visus šiuos modelius galima įsivaizduoti kaip ciklus, kurių metu atliekami etapas po etapo, tačiau iteraciniuose modeliuose šie ciklai projekto eigoje kartojasi, kol pasiekiamas tikslas – sukuriama programinė įranga. Iteracinis, Fontano, RUP, Scrum yra judrieji modeliai, taikant šiuos modelius projektas vykdomas iteracijomis, pakartotinai nusakomi ir realizuojami nauji reikalavimai kuriamam

produktui. Krioklio ir PRINCE2 yra nejudrieji modeliai, kurie taikomi projektams, kuriuose yra aiškūs reikalavimai, jų pasikeitimo tikimybė yra labai mažai tikėtina.

SCRUM ir RUP modeliai dėl jose išskirtų rolių yra taikytini tik komandoms, kiti nagrinėti modeliai gali būti taikomi dirbant ir vienam asmeniui.

Visi nagrinėti modeliai vienokiu ar kitokiu aspektu apima rizikos, kokybės ir išlaidų valdymą. Pavyzdžiui RUP per tam skirtas darbų sekas, PRINCE2 tam skirtus procesus, SCRUM – per etapuose atliekamas veiklas, kiti modeliai – per etapų metu sukuriamą dokumentaciją ir t.t. Literatūroje teoriškai nusakoma, kada tai turėtų būti daroma, praktikoje galima pasitelkti detaliau šias veiklas nusakančius standartus ir modelius.

Scrum yra vikrusis modelis, jis akcentuoja, kad geriau anksčiau egzistuojantis produktas nei išsamesnė dokumentacija. Kiti modeliai yra sunkiasvoriai, gausiai dokumentuoti, jų metu sukuriamas didelis ataskaitų, dokumentų kiekis. Šių modelių savybių palyginimai pateikti 2 lentelėje.

2. PROGRAMINĖS ĮRANGOS PROJEKTŲ VALDYMO MODELIŲ TYRIMAS

2. 1 Programinės įrangos projekto valdymo modelių teorinio tyrimo metodologija

Atlikus projektų valdymo programinės įrangos analizę, su pasirinktu įrankiu suplanuojami vidutinio dydžio projektai pagal krioklio, SCRUM ir RUP modelius. Pagal krioklio modelį modeliuojami 2 variantai: kai projekto reikalavimai pakinta projekto eigoje, ir kai jie nekinta, modeliuojant pagal SCRUM ir RUP modelius laikoma, kad prie pasikeitusių reikalavimų prisitaikoma vykdant iteracijas. Modeliuojant naudojami fazių, etapų, darbų sekų pavadinimai, nevardijant konkrečių jų metu vykdomų darbų, nes tai eksperimento rezultatams įtakos neturės.

Laikoma, kad vidutinio dydžio projektas tai projektas, kurio [1, 21]:

- trukmė 1-2 metai;
- produktą sudaro 50 400 kodo eilučių.

Modeliuojant, bus laikoma, kad projektą vykdo 6 programuotojai ir 1 projektų vadovas.

Laikoma, kad 1 programuotojas per darbo dieną parašo 80 kodo eilučių [1, 24, 25]. Jeigu programavimo darbus vykdys 6 programuotojai, tai pagal turimą kodo kiekį gauname, kad programavimo darbai sudarys 50400/6/80=105 projekto darbo dienų. Remiantis programavimo trukme bus apskaičiuojama projektų trukmė pagal šio etapo procentinę dalį, skirtinguose modeliuose.

Programavimas atliekamas šiuose modelių etapuose/fazėse (žr. 3 lentelę):

3 lentelė. modelių etapai/fazės, kuriuose atliekamas programavimas.

Modelis	Krioklio modelis	Scrum	RUP
Etapas arba fazė	Programavimo etapas	Kūrimas (SCRUM iteracijos)	Vystymo fazė; Konstravimo fazė; Perdavimo.

Projektai modeliuojami taikant literatūroje aprašytus esminius modelių taikymo principus.

Projektų kainos ir trukmės apskaičiavimui bus naudojamos MS Project projektų ataskaitos. Dėl projekto kainos vaizdumo, resursams priskiriamos užmokestis, kokį galėtų gauti šių profesijų atstovai šiuo metu Lietuvoje[33, 34]:

- Projekto vadovas - 20,00 Lt/val.;
- Programuotojas - 10,00 Lt/val.

2. 2 Projektų valdymo programinė įranga

Projektams planuoti dažnai naudojama tam skirta programinė įranga. Jį suteikia galimybę sudaryti projekto planus, suplanuoti užduočių atlikimo laiką. Priklausomai nuo pasirinktos programinės įrangos, ji gali suteikti galimybę užduotims priskirti resursus (darbuotojus, projekte naudojamus resursus), preliminariai nustatyti projekto pabaigos datą ir jo vykdymo kainą. Programinė įranga pasirinkta analizei aprašyta žemiau.

LeadingProject - suteikia galimybę kurti projekto planus, priskirti resursus projekto veikloms, stebėti vykdymo eigą, valdyti kaštus ir analizuoti darbo apkrovas. Programa leidžia vartotojams efektyviai valdyti projektus, padalinti veiklas, resursus ir kurti projektus neriboto dydžio ir ilgio. LeadingProject padeda vaizdžiai planuoti ir stebėti kelių lygiagrečių projektų eigą.

Microsoft Project - viena didžiausių ir labiausiai naudojama programinė įranga projektų planavimui. Planai gali būti vertinti pagal resursų lygį ir yra vaizduojami Gantt diagramoje. Gali atvaizduoti projekto kritinį kelią. Programa taip pat gali skirti vartotojus skirtingas grupes. Šios skirtingos vartotojų grupės gali turėti skirtingas priėjimo prie projekto duomenų teises. Suteikia galimybę kurti įvairias projekto ataskaitas (eigos, kainos, resursų panaudojimo ir kitas).

- Primavera - tai projekto planavimo ir valdymo sprendimų rinkinys, kurį galima pritaikyti, naudojant tik reikalingus jo modulius. Moduliai apima:
- Primavera Contractor - sinchronizuoja rangovus ir subrangovus.
- Expedition Profesional - suteikia rangovų valdymą per internetą.
- PrimeContract - automatizuoja verslo procesus.
- Cost Management - kaštų valdymas ir prognozė.

- Progress Reporting - sujungia atskirus projekto vienetus bendrą planą.
- Project Planer Profesional - didelių projektų valdymas.
- SureTask - skirta mažiems ir vidutinio dydžio projektams.

Asta Powerproject - tai projektų valdymo sprendimas skirtas valdyti projektus, optimizuoti resursus ir kontroliuoti projekto kaštus. Ši programa apima projekto valdymą, resursų ir kaštų valdymo funkcijas, grafikus ir pranešimus.

OpenProj - tai atviro kodo projektų valdymo programa, kurioje mėginama išpildyti MS Project suteikiamas galimybes. Ši programa naudoja JAVA platformą, o tai suteikia galimybę programą naudoti skirtingose operacinėse sistemose.

Pasirinkta projektų planavimo programinė įranga buvo lyginama šiais aspektais:

- Galimybė naudoti projektų šablonus – tai galimybė susikurti karkasą, pagal kurį būtų galima sudaryti kelis skirtingus projektus;
- Veiklų išskirstymas į detalesnes – veiklų grupavimas;
- Veiklų ryšiai – galimybė nustatyti veiklų tarpusavio pasiskirstymą laike (gali vykti lygiagrečiai, viena pasibaigus kitai, vienai prasidėjus pradedama kita, abi baigiamos vienu metu);
- Biudžeto planavimas – galimybė priskirtiems resursams (veiklų dalyviams, vykdytojams) priskirti valandinį ar kitą atlygį ir taip apskaičiuoti projekto kainą;
- Gantt (vizualizus veiklų pasiskirstymas laike) - diagramos pateikimas pagal sudarytą projekto planą;
- Kritinio kelio metodas – nustatymas kokiai veiklai vėluojant, vėluoja visas projektas;
- Integracija su MS Project – galimybė panaudoti jau esamus projektų planus, arba eksportuoti į MS Project palaikomą bylos formatą;
- Perspėjimai – įspėjimai, jeigu projekto plane vienu metu panaudojamas vienas resursas, kitų galimų kolizijų nustatymas;
- Resursų diagramos – resursų panaudojimo laike grafinis atvaizdavimas;
- Resursų lygiavimas – projekto pradžios ir pabaigos nustatymas pagal projekto veiklose naudojamų resursų prieinamumą;
- Galimybė importuoti resursus – galimybė importuoti arba eksportuoti jau aprašytus resursus.

4 lentelė. Programinės įrangos palyginimas

Funkcijos	**Programinė įranga**				
	MS project	**Leading Project**	**Primavera**	**Asta Powerproject**	**OpenProj**
Projektų šablonai	+	-	-	+	-
Veiklų išskirstymas į detalesnes	+	+	+	+	+
Veiklų ryšiai	+	-	+	+	+
Biudžeto planavimas	+	+	+	+	+
Gantt diagramos	+	+	+	+	+
Kritinio kelio metodas	+	-	-	-	+
Integracija su MS Project	+	-	-	+	+
Perspėjimai	+	+	+	+	-
Resursų diagramos	+	-	+	+	+
Resursų lygiavimas	+	+	+	+	+
Galimybė importuoti resursus	+	-	-	-	-

Atlikus programinės įrangos, skirtos projektų valdymui (žr. 4 lentelę), galima daryti išvadą, kad MS Project yra labiausiai išbaigta ir turinti daugiausiai funkcijų projektų valdyme programinė įranga, todėl ji yra pasirenkama projektų planų sudarymui pagal pasirinktus nagrinėtus modelius.

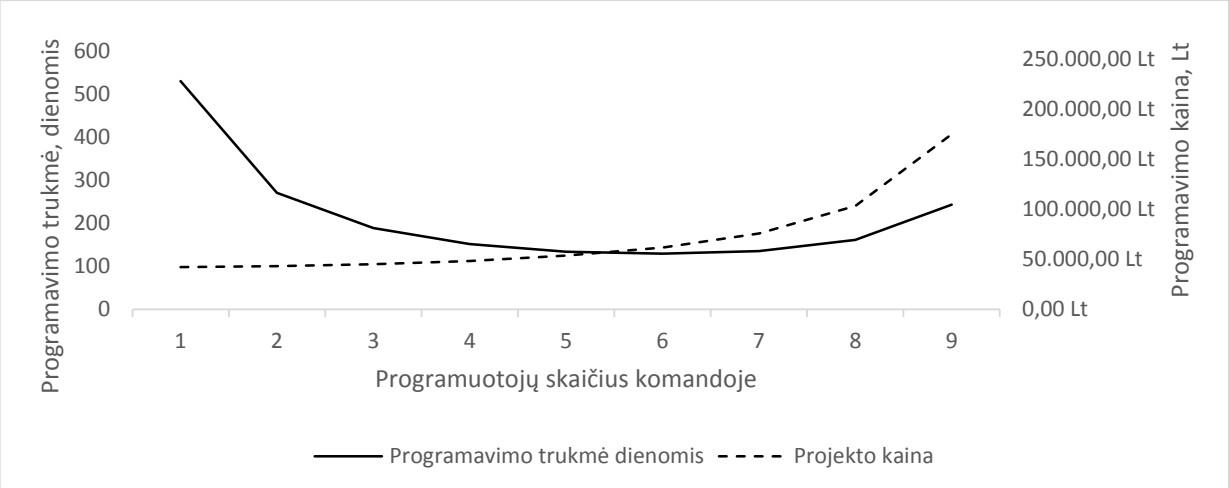
2. 3 Darbo grupės produktyvumas

Kuriant didesnės apimties programinę įrangą, dėl didesnės kodo apimties ir riboto darbo atlikimui skirto laiko, ją dažnai kuria ne pavieniai programuotojai, o programuotojų grupės.

Didėjant komandos narių skaičiui, didėja asmenų, su kuriais reikia bendradarbiauti (ryšių), kiekis. Jeigu grupę sudaro n programuotojų, o kiekvienas jų turėtų tarpusavyje bendrauti (tinklinė grupės struktūra), tai ryšių skaičių būtų galima apskaičiuoti pagal formulę[1]:

$$k = n \times \frac{n-1}{2} \quad (1)$$

Kiekvienas ryšis mažina programuotojo našumą dėl poreikio bendrauti. Jeigu laikysime, kad dirbdami programuotojai pavieniui per dieną parašo 95 kodo eilutes, viena jo darbo diena kainuoja 80 Lt, o kuriamą programinę įrangą sudaro 50400 kodo eilutės, laikant, kad vienas ryšys sumažina programuotojo našumą 2 kodo eilutėmis per dieną, tai priklausomybę tarp komandos narių skaičiaus ir projekto trukmės bei kainos būtų tokia, kaip 10 paveikslėlyje pavaizduotoje diagramoje.



10 pav. *Komandos narių skaičiaus įtaka projekto kainai ir trukmei.*

Teoriškai atrodytų, kad didesnis programuotojų skaičius galėtų atlikti programavimo darbus anksčiau, tačiau dėl mažėjančio jų našumo didėjant komandos narių skaičiui, kai komandą sudaro daugiau nei 6 asmenys (nagrinėjant pastarąjį atvejį), programavimo darbai būtų atliekami lėčiau, nei jei šiuos darbus atliktų mažesnė komanda. Taip pat galima pastebėti, kad mažėjant programavimo darbų trukmei, didėja jų kaina.

Panašios tendencijos išlieka projektus vykdant nepriklausomai nuo modelio. Komandos narių produktyvumo mažėjimą, didėjant narių skaičiui galima sumažinti naudojant skirtingas grupės narių struktūras (hierarchinę, žvaigždės), tačiau visiškai ryšių grupėse atsisakyti negalima ir priklausomybė tarp komandos narių skaičiaus ir jų našumo išlieka[1].

2. 4 Projektų modeliavimas

2. 4. 1 Krioklio modelis

Literatūroje aprašytas projekto laiko pasiskirstymas skirtingiems projekto etapams pagal teorinius modelius aprašomas skirtingai. Taip pat varijuoja laiko pasiskirstymas pagal laiko pasiskirstymą realiai vykdytuose projektuose. Projektavimui bus naudojami Ye Yang, Mei He, Mingshu Li, Q ing Wang ir Barry Boehm atlikto tyrimo rezultatuose pateikto projekto laiko pasiskirstymas etapams (modeliavime bus naudojami tyrime aprašyti etapų pavadinimai) taikant krioklio modelį. Šis laikas pasiskirsto sekančiai[26, 27, 35]:

- Planavimas ir reikalavimų surinkimas 18%;
- Architektūros specifikavimas 18%;
- Programavimas 38%;
- Testavimas 21%;
- Perdavimas 5%.

Pagal programavimo trukmę (105 darbo dienos), laikas projekto etapams pasiskirstytų kaip pateikta 5 lentelėje:

5 lentelė. Laiko pasiskirstymas krioklio modelyje

Etapas	Projekto dalis %	Darbo dienos
Planavimas ir reikalavimų surinkimas	18	50
Architektūros specifikavimas	18	50
Programavimas	38	105
Testavimas	21	58
Perdavimas	5	14

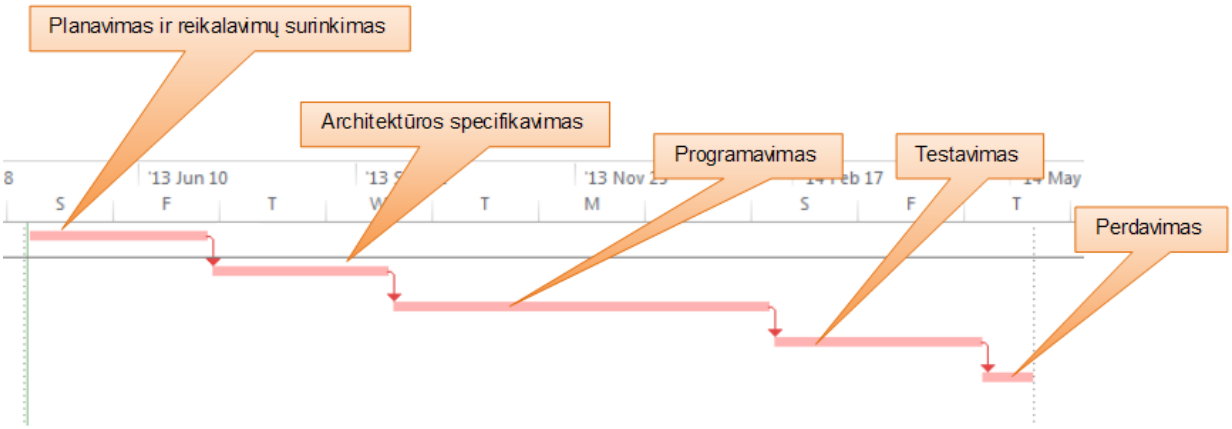
Modeliuojant projektus skaičiavimai bus atlikti naudojant šiuos resursus, kurie bus priskirti etapams nurodytiems 6 lentelėje. Dėl projekto kainos vaizdumo, šiems resursams priskiriamos užmokestis, kokį galėtų gauti šių profesijų atstovai šiuo metu Lietuvoje[33, 34].

6 lentelė. Etapuose naudojami resursai

Resursas	Etapai
Projekto vadovas	Planavimas ir reikalavimų surinkimas; Architektūros specifikavimas.
Programuotojas 1	Architektūros specifikavimas; Programavimas; Testavimas; Perdavimas.
Programuotojas 2	Architektūros specifikavimas; Programavimas; Testavimas; Perdavimas.
Programuotojas 3	Architektūros specifikavimas; Programavimas; Testavimas; Perdavimas.
Programuotojas 4	Architektūros specifikavimas; Programavimas; Testavimas; Perdavimas.
Programuotojas 5	Architektūros specifikavimas; Programavimas; Testavimas; Perdavimas.
Programuotojas 6	Architektūros specifikavimas; Programavimas; Testavimas; Perdavimas.

Suprojektavus krioklio modelį, kai reikalavimai nekinta, gaunama Gantt diagrama (žr. 11 pav.) matyti, kad etapai vyksta vienas po kito iki projekto pabaigos. Gaunami šie rezultatai:

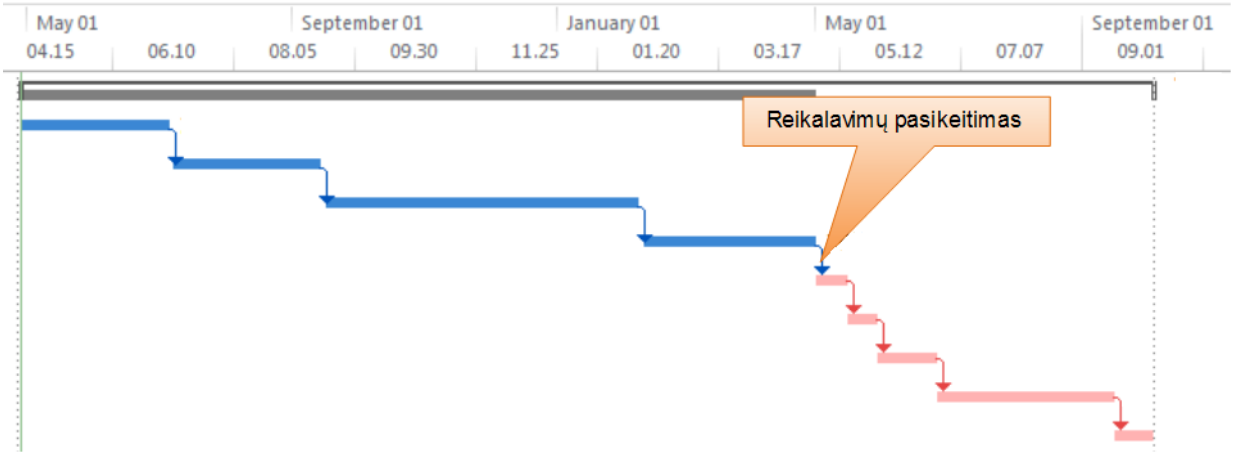
- Projekto kaina: 153280 Lt;
- Projekto trukmė: 277 dienos.



11 pav. *Krioklio modelio Gantt diagrama*

Modeliuojamas projektas pagal krioklio modelį, kai reikalavimai pakinta testavimo etapo pabaigoje. Šis etapas pasirenkamas todėl, nes tik tada užsakovas gali pamatyti sukurtą PĮ. Laikoma, kad kartojant etapus reikėjo skirti 20% anksčiau jau skirto laiko ir resursų, išskyrus testavimui, laikoma, kad reikės 100%, nes reikės patikrinti visos kuriamos PĮ suderinamumą su naujai nusakytais, pakeistais reikalavimais. Gaunama Gantt diagrama, kurioje matyti (žr. 12 pav.), kad etapai vykę iki reikalavimų pasikeitimo atkartojami mažesnėmis dalimis, išskyrus testavimą. Pagal MS Project projekto ataskaitą, gaunami šie rezultatai:

- Projekto kaina: 211,200 Lt;
- Projekto trukmė: 375 dienos.



12 pav. *Krioklio modelis, kai pakinta reikalavimai testavimo etapo pabaigoje*

2. 4. 2 SCRUM modelis

Literatūroje projekto laiko pasiskirstymas pagal SCRUM modelį nerastas, daugiausia akcentuojama, kad taikant šį modelį didžiausias prioritetas skiriamas programavimui (produkto kūrimui), rašoma minimali dokumentacija arba visiškai nerašoma. Modeliavimui, laiko pasiskirstymui bus naudojamas vidutinis panašių iteracinių modelių projekto etapų laiko pasiskirstymas pagal Ye Yang, Mei He, Mingshu Li, Q ing Wang ir Barry Boehm atliktą tyrimą. Konstrukcijos ir kūrimo etapai modeliuojant bus apjungti į vieną - kūrimo, nes jų metu atliekami tie patys darbai (žr. 7 lentelę).

7 lentelė. SCRUM modelio laiko pasiskirstymas ir etapų atitikimas

Projekto laiko sąnaudos %	Modelių etapai	Scrum
14	Planavimas ir reikalavimų surinkimas	Planavimas
13	Architektūros specifikavimas	-
45	Programavimas	Kūrimas
19	Testavimas	Kūrimas
9	Perdavimas	Pridavimas

Pagal programavimo trukmę ir etapų kiekį, laikas ir projekto dalys tenkančios etapams pasiskirstytų sekančiai (žr. 8 lentelę):

8 lentelė. SCRUM modelio laiko pasiskirstymas

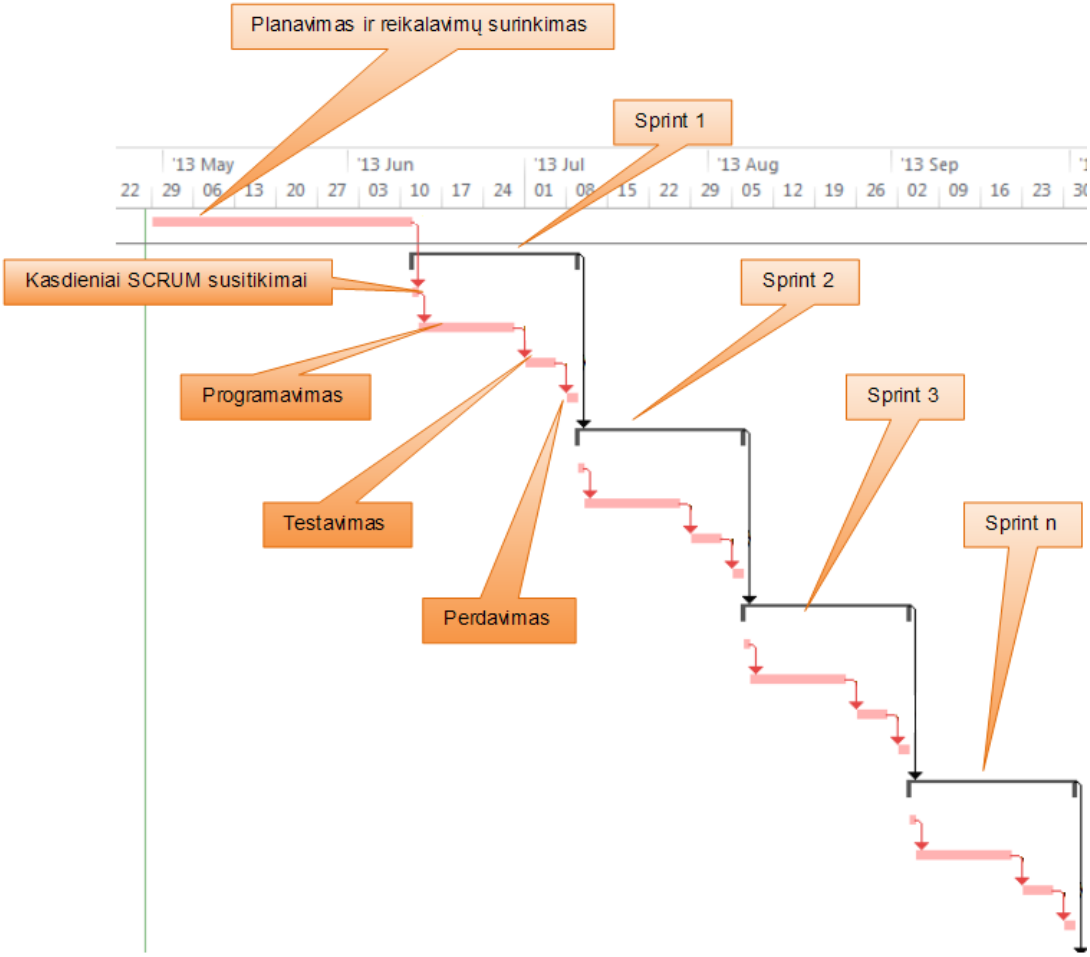
Etapas	Projekto dalis %	Darbo dienos
Planavimas ir reikalavimų surinkimas	16	32
Programavimas	52	105
Testavimas	22	44
Perdavimas	10	20

Modeliuojant programavimas, testavimas ir perdavimas tolygiai suskirstomi sprintams. Taip pat prie kiekvienos iteracijos pridedama 1 (20 dienų po 15 minučių, suapvalinant į didžiąją pusę gaunama 1 darbo diena) darbo diena, skirta kasdieniams SCRUM susitikimams.

Modeliuojant skaičiavimai bus atlikti naudojant šiuos resursus, kurie bus priskirti etapams nurodytiems 9 lentelėje.

9 lentelė. Etapuose naudojami resursai

Resursas	Etapai
Projekto vadovas (SCRUM lyderis)	Planavimas ir reikalavimų surinkimas Programavimas Testavimas Perdavimas
Programuotojas 1	Programavimas Testavimas Perdavimas
Programuotojas 2	Programavimas Testavimas Perdavimas
Programuotojas 3	Programavimas Testavimas Perdavimas
Programuotojas 4	Programavimas Testavimas Perdavimas
Programuotojas 5	Programavimas Testavimas Perdavimas
Programuotojas 6	Programavimas Testavimas Perdavimas



13 pav. *SCRUM modelio Gantt diagrama*

Pamodeliavus projektą su MS Project, gaunama Gantt diagrama matoma 13 paveikslėlyje. Pagal MS Project projekto ataskaitą, gaunama, kad:

- Projekto kaina: 120 320 Lt;
- Projekto trukmė: 212 dienos.

2. 4. 3 RUP modelis

Projektui vykstant pagal RUP modelį, projekto laikas pasiskirstymas fazėms yra toks [28]:

- Pradžios fazė 10%;
- Vystymo fazė 30%;
- Konstravimo fazė 50%;
- Perdavimo fazė 10%.

Programavimo darbai bus vykdomi vystymo, konstravimo ir perdavimo fazėse. Pagal modelio taikymo principus (žr. 8 pav.) tai sudarytų apytikriai 33% vystymo, 60% konstravimo ir 7% perdavimo fazėse. Kiekvienoje šių fazių vykdomos darbų sekos (angl. workflows) , kurių apytikrė fazės laiko dalis procentais pateikiama 10 lentelėje.

10 lentelė. Darbų sekų trukmės pasiskirstymas fazėse

	Darbų sekų trukmės pasiskirstymas fazėse(%)			
Darbų seka	Pradžios	Vystymas	Konstravimas	Perdavimas
Verslo modeliavimas	36	14	2	2
Reikalavimu apibrėžimas	48	21	5	5
Analizė ir projektavimas	12	34	10	7
Realizacija	2	21	34	24

Testavimas	2	7	24	37
Įdiegimas	0	3	25	25

Pagal procentinę programavimo dalį fazėse apskaičiuojama darbo dienų skaičius fazėse, kuriose vykdomas programavimas, o pagal atliktus skaičiavimus apskaičiuojama laiko pasiskirstymas pradžios fazėje. Skaičiavimų rezultatai pateikiami 11 lentelėje.

11 lentelė. Skaičiavimų rezultatai

	Darbų sekų pasiskirstymas fazėse išreikštas darbo dienomis			
Darbų seka	Pradžios	Vystymas	Konstravimas	Perdavimas
Verslo modeliavimas	15	23	4	1
Reikalavimu apibrėžimas	20	35	9	2
Analizė ir projektavimas	5	56	19	2
Realizacija	1	35	63	7
Testavimas	1	12	44	11
Įdiegimas	0	5	46	8

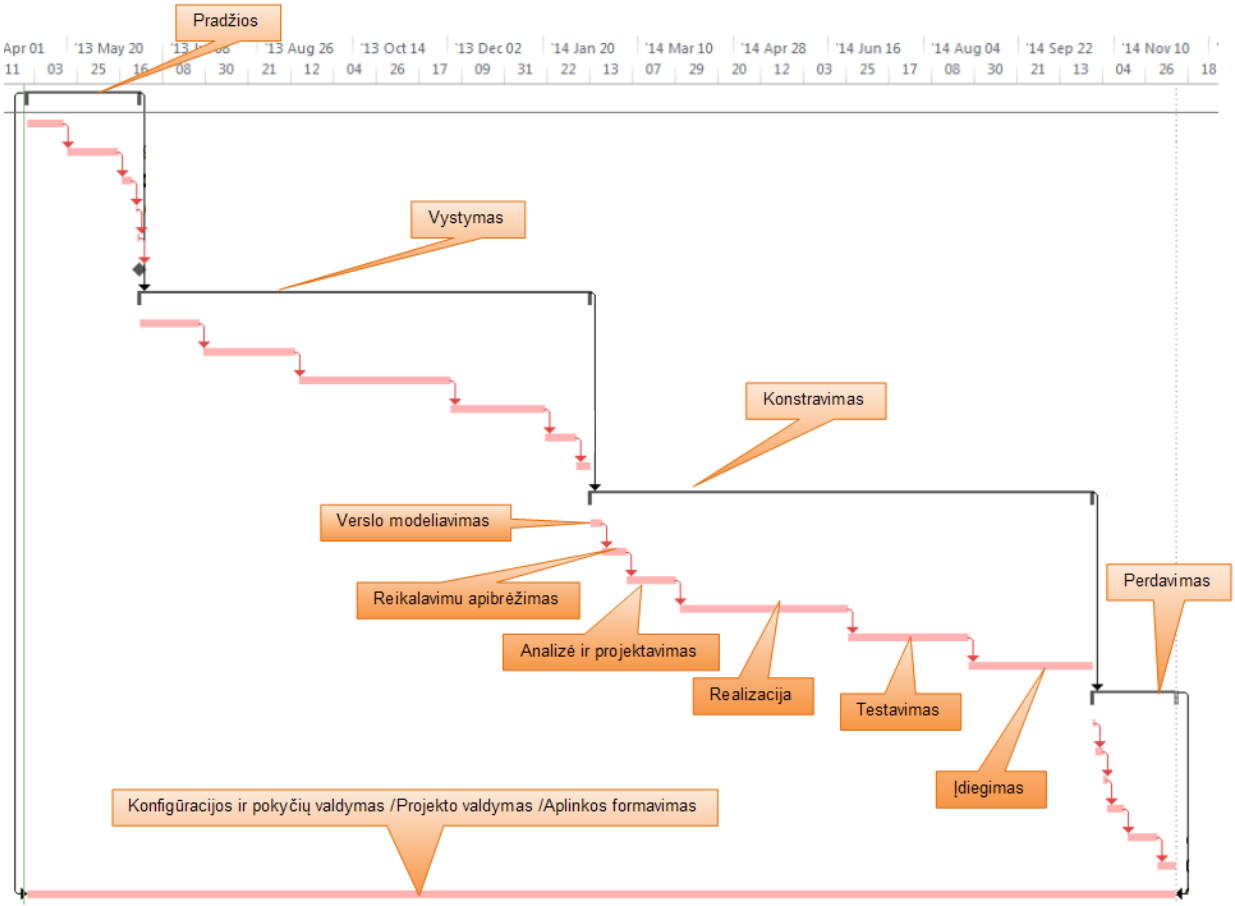
Modeliuojant laikoma, kad programuotojai dalyvauja inžinierinėse darbų sekose, projekto vadovas pagalbinėse. Resursų priklausymas darbų sekoms pateiktas 12 lentelėje.

12 lentelė. Darbų sekose naudojami resursai

Fazė	Darbų seka	Resursai
Pradžios	Verslo modeliavimas	Programuotojas 1-6
	Reikalavimu apibrėžimas	
	Analizė ir projektavimas	
	Realizacija (programavimas)	
	Testavimas	
	Įdiegimas	
	Konfigūracijos ir pokyčių valdymas	Projekto vadovas
	Projekto valdymas	
	Aplinkos formavimas	
Vystymo	Verslo modeliavimas	Programuotojas 1-6, užsakovas
	Reikalavimu apibrėžimas	
	Analizė ir projektavimas	
	Realizacija (programavimas)	
	Testavimas	
	Įdiegimas	
	Konfigūracijos ir pokyčių valdymas	Projekto vadovas
	Projekto valdymas	
	Aplinkos formavimas	
Konstravimo	Verslo modeliavimas	Programuotojas 1-6, užsakovas
	Reikalavimu apibrėžimas	
	Analizė ir projektavimas	
	Realizacija (programavimas)	
	Testavimas	
	Įdiegimas	
	Konfigūracijos ir pokyčių valdymas	Projekto vadovas
	Projekto valdymas	
	Aplinkos formavimas	
Perdavimo	Verslo modeliavimas	Programuotojas 1-6, užsakovas
	Reikalavimu apibrėžimas	
	Analizė ir projektavimas	

	Realizacija (programavimas)	
	Testavimas	
	Įdiegimas	
	Konfigūracijos ir pokyčių valdymas	Projekto vadovas
	Projekto valdymas	
	Aplinkos formavimas	

Modeliuojant apsiribojama, kad bus 4 iteracijos, kiekvienai fazei po vieną. Atlikus modeliavimą, gautas projekto vaizdas pateiktas 14 paveikslėlyje.



14 pav. *RUP modelio Gantt diagrama*

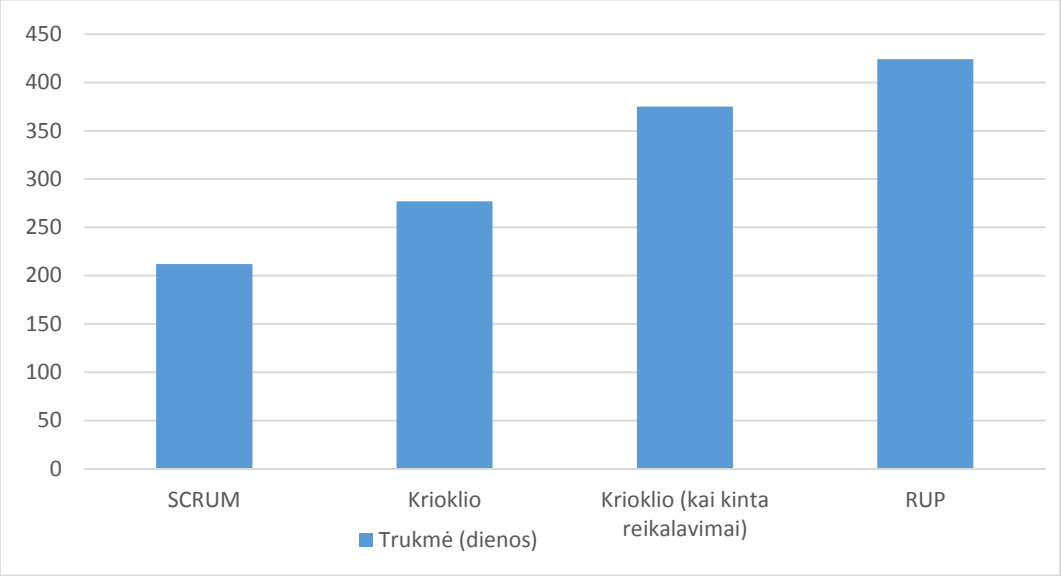
Pagal MS Project projekto ataskaitą, gaunami šie rezultatai:

- Projekto kaina: 271.360 Lt;
- Projekto trukmė: 424 dienos.

3. HIBRIDINIO MODELIO SUDARYMAS IR TYRIMAS

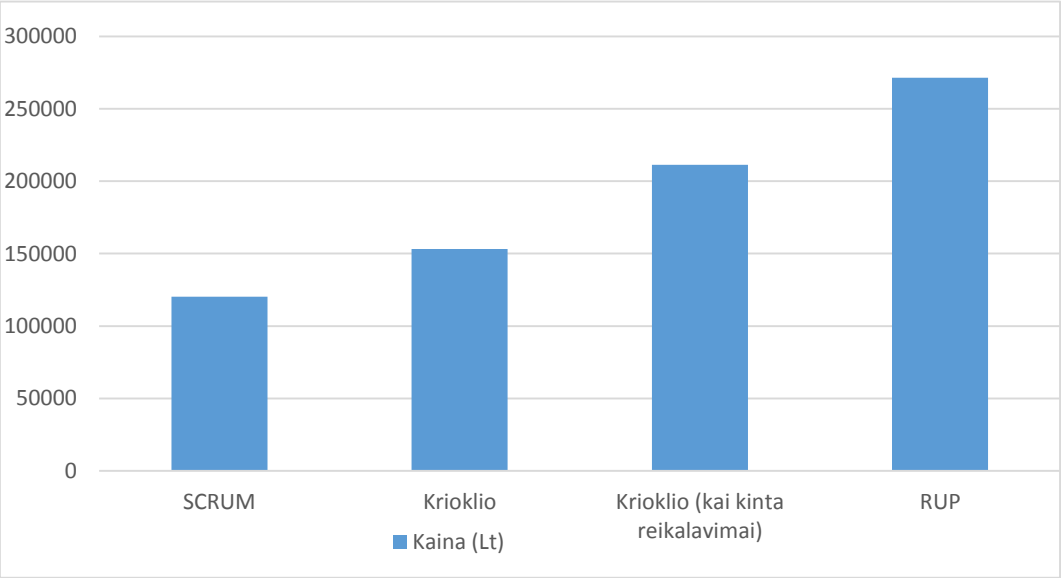
3. 1 Hibridinio modelio sudarymo prielaidos

Hibridinis modelis sudaromas Atlikus krioklio, RUP ir SCRUM modelių analizę ir teoriškai pamodeliavus projektų eigą pagal šiuos modelius nustatyta, kad projekto trukmė būtų mažiausia naudojant SCRUM modelį. Programinę įrangą kuriant pagal krioklio modelį, jeigu pakistų produktui skirti reikalavimai testavimo etape, projektas trukmė padidėtų 35%. Didžiausia projekto trukmė būtų ji vykdant pagal RUP modelį (žr. 15 pav.).



15 pav. *Modeliuotų projektų trukmės palyginimas*

Atitinkamai pagal projekto trukmę, pasiskirsto ir kainos: Pigiausia būtų projektą įvykdyti pagal SCRUM modelį, brangiausiai kainuotų projekto vykdymas pagal RUP modelį. Vykdant projektą pagal krioklio modelį, pakitus reikalavimams, keliamiems kuriamai programinei įrangai, projekto kaina padidėtų 38% (žr. 16 pav.).



16 pav. *Modeliuotų projektų trukmės palyginimas*

Negalima teigti, kad SCRUM modelis yra geriausias, nes pagal jį vykdant projektą nėra kuriama išsami produkto dokumentacija (taip kaip krioklio ir RUP atveju), o tai gali tapti projekto nesėkmės priežastimi, jeigu projekto komandą paliktų vienas ar daugiau narių. Praktikoje taikant šį modelį svarbūs komandos narių santykiai, jų motyvacija [29].

Vykdant projektą pagal krioklio modelį, sukuriamas pakankamas dokumentacijos kiekis (reikalavimų, architektūros specifikacija), kuris padėtų išvengti neigiamų pasekmių, jeigu projekto komandą paliktų vienas ar daugiau komandos narių. Tačiau pakitus ar nepilnai užsakovui nusakius esminius reikalavimus paskutiniuose projekto etapuose, projekto kaina ir trukmė išauga, o tokio įvykio rizika yra didžiausia programinės įrangos projektuose [30].

Projekto trukmė taikant RUP modelį yra ilgiausia, o kaštai didžiausi, lyginant su pagal kitus modelius modeliuotais projektais. Dėl dokumentacijos sukūrimo šio modelio taikymas gali padėti išvengti neigiamų pasekmių, jeigu projekto komandą paliktų vienas ar daugiau komandos narių. Šis modelis numato procesus ir priemones(konfigūracijos ir pokyčių valdymas, projekto valdymas, aplinkos formavimas), juos išskiria kaip atskiras disciplinas, kurios turėtų eliminuoti programinės įrangos projekto žlugimą. Iteracinis projekto vystymas padeda prisitaikyti prie kintančių reikalavimų. Visa tai šį modelį

daro griozdišką. Nors literatūroje nurodoma, kad šis modelis tinkamas taikyti mažiems, vidutiniams ir dideliems projektams, neadaptuoto šio modelio taikymas mažiems ir vidutiniams programinės įrangos projektams gali ženkliai pailginti trukmę ir padidinti kaštus[31].

Taip pat galima pastebėti, kad kuo mažesnė projekto laiko dalis skiriama programavimo etapui, tuo projekto kaina tampa didesnė, jo trukmė tampa ilgesnė.

3. 2 Hibridinio modelio sudarymas

Kuriamas hibridinis modelis remsis anksčiau atlikto teorinio tyrimo rezultatais. Modelis naudos šias nagrinėtų modelių savybes:

- Krioklio modelio etapus ir juose vykstančias veiklas, panašius jų prioritetus;
- SCRUM darbų sąrašą (angl. Backlog) – jis bus naudojamas kaip reikalavimų specifikavimo dalis, tinkamas projekto būsenai nustatyti;
- RUP modelio darbų sekų prioritetus juos pritaikant atitinkamiems krioklio modelio etapams. Kaip ir RUP, bus išskirtas reikalavimų pokyčių valdymas kaip atskira disciplina.

Kuriamo modelio etapų trukmė pasirenkama pagal krioklio modelį, nes šis modelis numato dokumentacijos rašymą ir o atlikus teorinį jo tyrimą nustatyta, kad projektas truko trumpiau nei RUP. Šių etapų laiko pasiskirstymas perskirstomas kuriamam modeliui (žr. 13 lentelę). Planavimo ir reikalavimų surinkimo etapas suskaidomas ir jo laikas paskirstomas šiems etapams:

- Darbų sąrašo sudarymas (angl. backlog) – naudojamas reikalavimų specifikacijos formavimui ir projekto būsenos nustatymui, pagal atliktų darbų dalį;
- Pasikeitimų ir projekto valdymas – RUP modelio reagavimo į reikalavimų pokyčius disciplina;
- Reikalavimų specifikavimas – krioklio modelyje naudojamas etapas, kurio metu aprašomi ir patvirtinami reikalavimai kuriamam produktui.

Projekto laikas perskirstomas naujiems etapams:

- 11% - darbų sąrašo sudarymui, pasikeitimų ir projekto valdymui
- 89% - iteracijose vykdomiems etapams.

13 lentelė. Modelio etapų pasiskirstymas projekto laike

Kuriamo modelio etapai	Projekto laikas %	Krioklio modelio etapai	Projekto laikas %
Darbų sąrašo sudarymas	5	Planavimas ir reikalavimų surinkimas	18
Pasikeitimų ir projekto valdymas	6		
Reikalavimų specifikavimas	10		
Architektūros specifikavimas	18	Architektūros specifikavimas	18
Programavimas	35	Programavimas	38
Testavimas	21	Testavimas	21
Perdavimas	5	Perdavimas	5

Iteracijoms pagal juose vyraujančius etapus suteikiami šie pavadinimai:

- Reikalavimų specifikavimas;
- Architektūros specifikavimas;
- Vystymas;
- Perdavimas.

Pagal RUP modelį, krioklio modelio etapams iteracijose parenkamas panašus etapų pasiskirstymas per iteracijas ir iteracijų savybes – per pirmas iteracijas didesnis prioritetas skiriamas reikalavimų specifikavimui, per vėlesnes – architektūros specifikavimui, programavimui, testavimui ir pridavimui (žr. 14 lentelę) . Taip per pirmąsias iteracijas siekiama daugiau laiko skirti reikalavimų surinkimui, per vėlesnes prisitaikyti prie atsiradusių naujų, pakitusių reikalavimų.

Pasikeitimų ir projekto valdymo dalis išskiriama po 25% kiekvienai iteracijai, darbų sąrašo sudarymo etapas atliekamas prieš prasidedant vykdyti projektą iteracijomis.

14 lentelė. Etapų pasiskirstymas iteracijose (iš kairės į dešinę) procentais

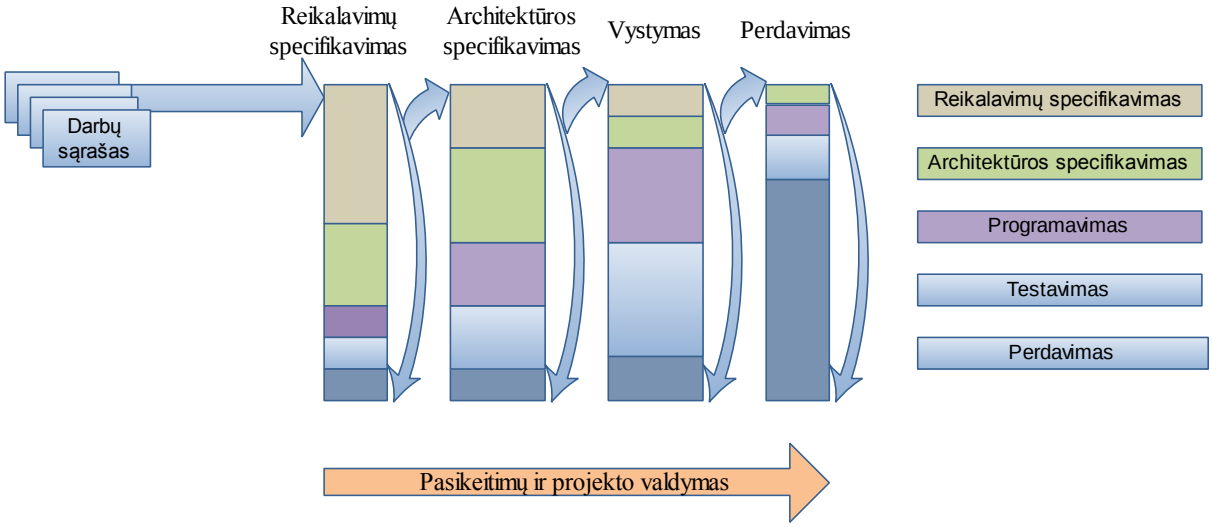
Iteracija / Etapas	Reikalavimų specifikavimas	Architektūros specifikavimas	Vystymas	Perdavimas
Reikalavimų specifikavimas	50	30	20	0
Architektūros specifikavimas	30	50	15	5
Programavimas	10	30	50	10
Testavimas	10	30	50	10
Perdavimas	10	10	20	60

Perskaičiuojami etapų pasiskirstymas kiekvienoje iteracijoje. Skaičiavimų rezultatai pateikti 15 lentelėje.

15 lentelė. Etapų pasiskirstymas kiekvienoje iteracijoje ir kiekvienos iteracijos pasiskirstymas iteracijoms skirtame laike procentais

Iteracija / Etapas	Reikalavimų specifikavimas (20)	Architektūros specifikavimas (30)	Vystymas (30)	Perdavimas (20)
Reikalavimų specifikavimas	45	20	13	0
Architektūros specifikavimas	27	33	10	6
Programavimas	9	20	32	12
Testavimas	9	20	32	12
Perdavimas	9	7	13	71

Pagal paskaičiuotas dalis sudaromas projekto valdymo modelio vaizdas (žr. 17 pav.), kuriame galima vizualiai matyti, kaip kinta etapų pasiskirstymas iteracijose (aukštesnis kvadratas – didesnė dalis projekto laiko skiriama etapui) ir iteracijų pasiskirstymą projekto metu (platesnis iteracijos plotis – daugiau projekto laiko jai skiriama).



17 pav. *Siūlomas modelis*

Pagal gautą projekto dalį procentais apskaičiuojama kiekvieno etapo trukmė darbo dienomis. Šie dydžiai bus reikalingi modeliuojant projektą pagal kuriamą modelį. Skaičiavimų rezultatai pateikiami 16 lentelėje.

16 lentelė. Modeliuojamo projekto apskaičiuoti laikai kiekvienoje iteracijoje darbo dienomis

Iteracija / Etapas	Reikalavimų specifikavimas	Architektūros specifikavimas	Vystymas	Perdavimas
Reikalavimų specifikavimas	15	9	6	0
Architektūros specifikavimas	16	27	8	3
Programavimas	11	32	53	11
Testavimas	6	19	32	6
Perdavimas	2	2	3	9

Taip pat apskaičiuojami projekto etapų, neįeinančių į iteracijas trukmė darbo dienomis. Darbų sąrašo sudarymas (angl. backlog) truks 15 dienų, pasikeitimų valdymui reikės 20 darbo dienų (žr. 17 lentelėje pateiktus dydžius).

Visi skaičiavimai atliekami remiantis, kad 35% projekto laiko skiriama programavimui, kuris trunka 105 dienas.

Pagal etapuose vykstančias veiklas, jiems paskirstomi resursai. Etapų ir resursų sąsajos pateikiamos 17 lentelėje.

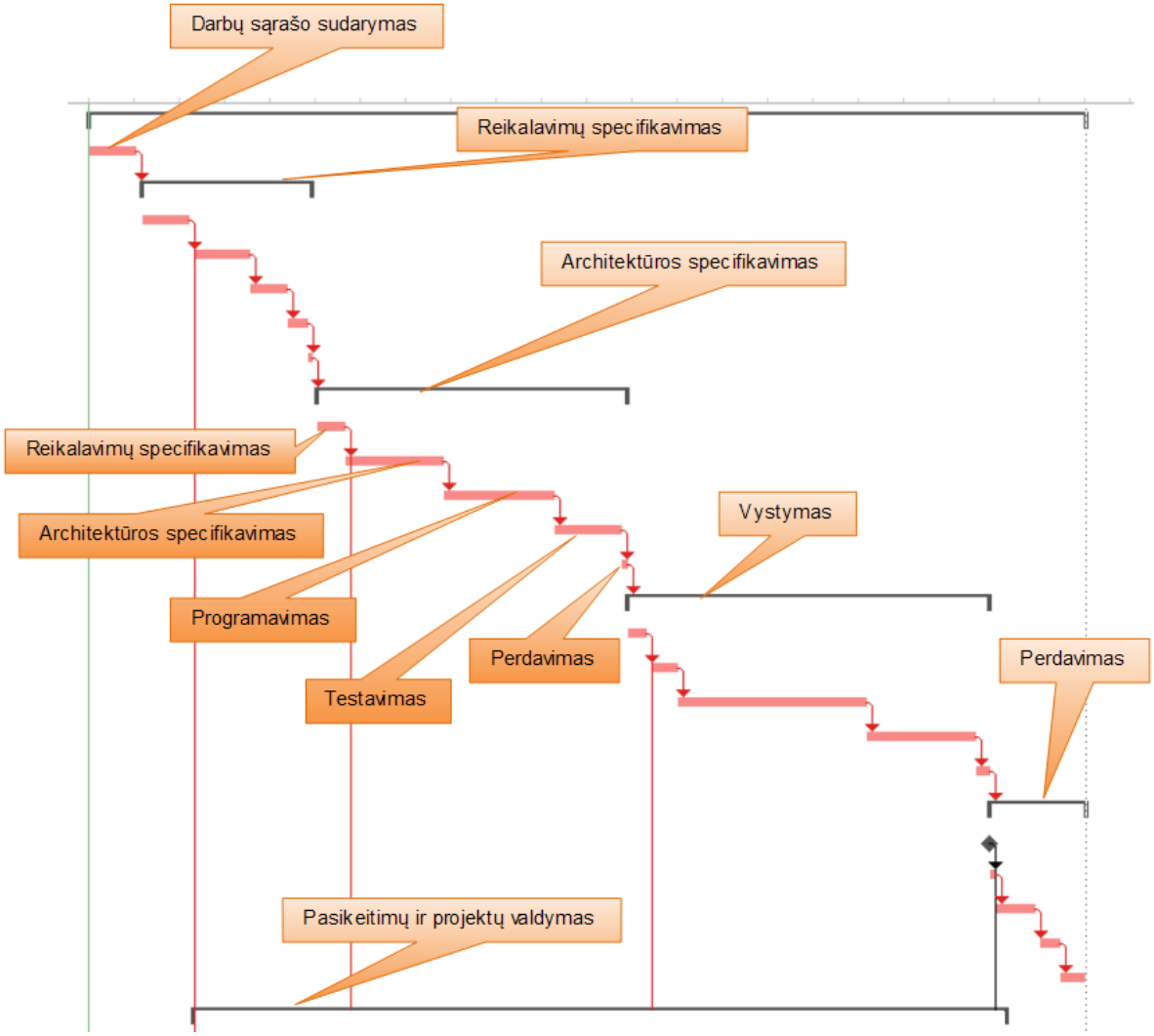
17 lentelė. Resursų pasiskirstymas kuriamo modelio etapuose

Iteracija	Etapas	Resursai
	Darbų sąrašo sudarymas	Projekto vadovas, užsakovas
	Pasikeitimų ir projekto valdymas	
Pradžios	Reikalavimų specifikavimas	Projekto vadovas
	Architektūros specifikavimas	Programuotojas 1-6, užsakovas
	Programavimas	
	Testavimas	
	Perdavimas	
Architektūros specifikavimas	Reikalavimų specifikavimas	Projekto vadovas
	Architektūros specifikavimas	Programuotojas 1-6, užsakovas
	Programavimas	
	Testavimas	
	Perdavimas	
Vystymo	Reikalavimų specifikavimas	Projekto vadovas
	Architektūros specifikavimas	Programuotojas 1-6, užsakovas
	Programavimas	
	Testavimas	
	Perdavimas	
Perdavimo	Reikalavimų specifikavimas	Projekto vadovas
	Architektūros specifikavimas	Programuotojas 1-6, užsakovas
	Programavimas	
	Testavimas	
	Perdavimas	

Atlikus projekto modeliavimą naudojant MS Project gaunami šie rezultatai:

- Projekto kaina: 125600 Lt;
- Projekto trukmė: 285 dienos.

Siūlomo projekto modelio Gantt diagrama pateikta 20 paveikslėlyje.

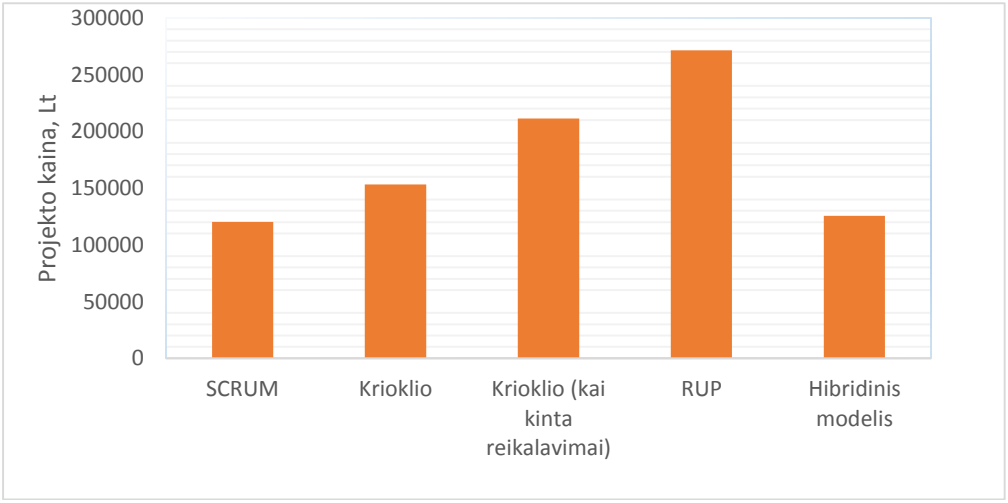


18 pav. *Hibridinio modelio projekto Gantt diagrama*

18 paveikslėlyje matome, kad didžiausia projekto darbo dalis atliekama vykstant architektūros ir vystymo iteracijoms. Šiose iteracijose didžiausias prioritetas skiriamas architektūros specifikavimui ir programavimui.

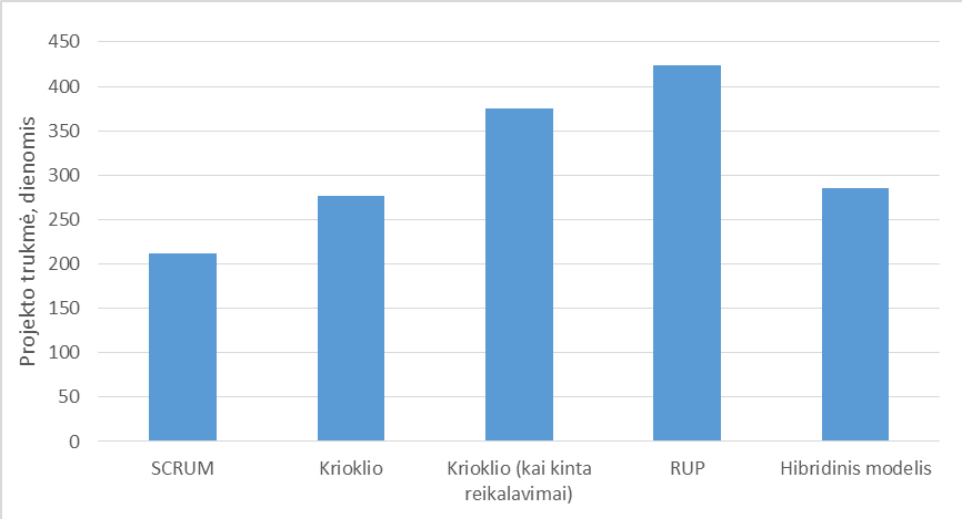
3. 3 Hibridinio modelio tyrimo išvados

Atlikus 3 projektų modeliavimą pagal skirtingus modelius, remiantis gautais rezultatais buvo suprojektuotas naujas modelis. Projekto (žr. 19 pav.), pagal šį modelį trukmė yra mažesnė nei RUP ir krioklio su pakitusiais reikalavimais.



19 pav. *Teoriškai tirtų modelių kainų palyginimas*

Projekto kaina yra nežymiai didesnė lyginant su SCRUM, tačiau siūlomas modelis numato produkto specifikacijos kūrimą, o tai su suteikia projekto eigai stabilumo, tuo atveju, jeigu projektą paliktų 1 ar daugiau komandos narių.



20 pav. *Modeliuotų projektų trukmės palyginimas*

Projekto trukmė (žr. 20 pav.) pagal siūlomą modelį yra artimiausia krioklio modeliui. Darbų kiekiai projekte pagal siūlomą modelį mažesni dėl atsisakytų verslo modeliavimo ir kitų etapų pagal RUP. Taip pat perimta 1 iš 3 darbo sekų, taip sumažinant darbų kiekį, bet išlaikant modelyje esminius bruožus, kurie jį daro pranašesnį už SCRUM (architektūros specifikavimas, pokyčių valdymas ir kt.)

4. PĮ PROJEKTŲ VALDYMO IR MODELIŲ TAIKYMO PRAKTIKOJE TYRIMAS

4. 1 Tyrimo tikslai:

Analizuojant PĮ projektų valdymo modelius be teorinių tiriamos srities mokslinių darbų analizės buvo iškeltas tikslas atlikti praktinį tyrimą, kurio tikslas ištirti PĮ projektų valdymo modelių taikymą Lietuvos įmonėse. Atliekant tyrimą buvo siekiama nustatyti:

- kokius programinės įrangos valdymo modelius praktikoje taiko Lietuvos IT įmonėse dirbantys projektų vadovai;
- kokie prioritetai išskiriami programinės įrangos projekto laiko etapams praktikoje;
- kokios priemonės naudojamos projektų planavimui;
- nustatyti, kaip dažnai užsakovai keičia reikalavimus kuriamai PĮ.

4. 2 Tyrimo metodologija

Tyrimas buvo atliekamas šiais etapais:

- Respondentų paieška;
- Apklausos anketos sudarymas;
- Anketos perdavimas respondentams;
- Surinktų duomenų apibendrinimas.

Siekiant surasti respondentų buvo išsiųsti užklausimai dėl dalyvavimo apklausoje šioms įmonėms: UAB "BALTIC DATA CENTER", UAB ”ATEA“, , UAB "METASITE BUSINESS SOLUTIONS", "BITĖ Lietuva“, AB "Lietuvos paštas", UAB "ETRONIKA", UAB "Alna Business Solutions", UAB "Algoritmų sistemos", UAB "BLUE BRIDGE CODE", Barclays, UAB "Tieto Lietuva", UAB "Prewise", iTree Lietuva, UAB "BULL Baltija", UAB "BT Grupė", UAB "VIP Solutions", UAB "NRD”, UAB "Baifoteka".

Taip pat buvo susitikta su Klaipėdos miesto savivaldybės ir Klaipėdos teritorinė ligonių kasos IT skyriaus vadovais, siekiant išsiaiškinti, ar jie galėtų dalyvauti apklausoje. Buvo gautas atsakymas, kad šiose valstybinėse įstaigose IT skyrių darbuotojai jokių programinės įrangos projektų nevykdo, todėl apklausoje dalyvauti negali.

Apklausoje sutiko dalyvauti UAB "METASITE BUSINESS SOLUTIONS", UAB "Baifoteka" ,UAB ”ATEA“. Buvo susitarta, kad perdavus anketas šių įmonių atstovams, jie jas paplatins projektų vadovams. Taip pat buvo gauti 2 laisvai samdomų projektų vadovų kontaktiniai duomenys, kurie sutiko dalyvauti apklausoje.

Tymui atlikti buvo nuspęsta sukurti anketą, kuria būtų galimą užpildyti internetu. Tam tikslui pasiekti pasirinktas apklausa.lt apklausų puslapis, dėl to, kad jame galima nemokamai sukurti ir patalpinti anketas, patogus anketų pildymas ir vaizdus atsakymų pateikimas.

Sudarant klausimus buvo laikomasi prielaidos, kad projektų vadovai gali projektus vykdyti krioklio etapais, tačiau tokio jo vykdymo neišskirti kaip modelį. Taip pat sudarant klausimus, buvo išskirtas laukelis, kuriame buvo galima parašyti taikomo modelio pavadinimą, jeigu tarp atsakymų variantų jo nebūtų. Panašiomis prielaidomis buvo sudaryti ir kiti klausimai. Siekiant surinkti kuo daugiau duomenų, buvo numatyta, kad į visus klausimus buvo neprivaloma atsakyti, norint patvirtinti atsakymus.

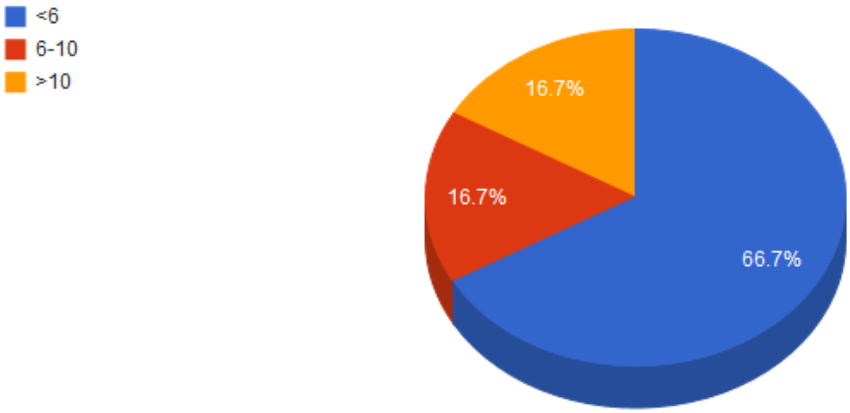
Anketos nuoroda buvo išsiųsta apklausoje sutikusių dalyvauti įmonių atstovams. Į anketos klausimus buvo atsakyta 10 kartų.

4. 3 Tyrimo rezultatai

Tyrime dalyvavo 10 projektų vadovų. Pagal apklausos rezultatus, 8 respondentai buvo jaunesni nei 30 metų amžiaus, 2 respondentai vyresni nei 40 metų. Visi apklausos dalyviai buvo įgiję aukštąjį išsilavinimą. Respondentų patirtis programinės įrangos projektų valdyme varijavo nuo 1 iki 20 metų. Respondentai nurodė, kad turi patirties nuo 2 iki 15 projektų vadovavime.

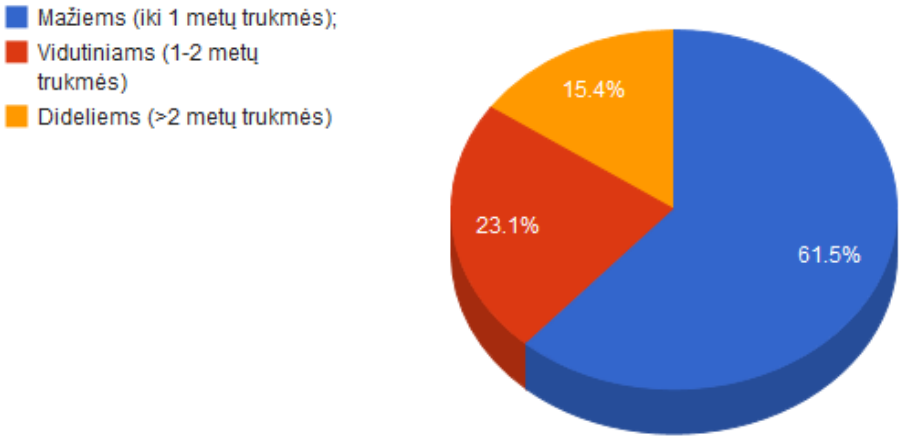
Visi respondentai atsakė teigiamai į klausimą, ar jie įtraukia užsakovą į projekto eigą (demonstruoja projekto eigą, klausiate jo nuomonės apie kuriamą produktą). Taip pat dauguma apklausoje dalyvavusių projektų vadovų (9 iš 10), atsakė, kad projekto metu perduoda sukurtą produkto prototipą užsakovui, o tai leidžia daryti prielaida, kad produktas projekto metu yra vystomas iteracijomis.

Dauguma apklausoje dalyvavusių respondentų nurodė, kad vadovauja mažesnėms nei 6 asmenų darbuotojų grupėms (žr. 21 pav.). 2 projektų vadovai vadovauja grupėms, kurias sudaro nuo 6 iki 10 darbuotojų, taip pat 2 projektų vadovai nurodė, kad jų vadovaujamoje darbo grupėje yra daugiau kaip 10 darbuotojų.



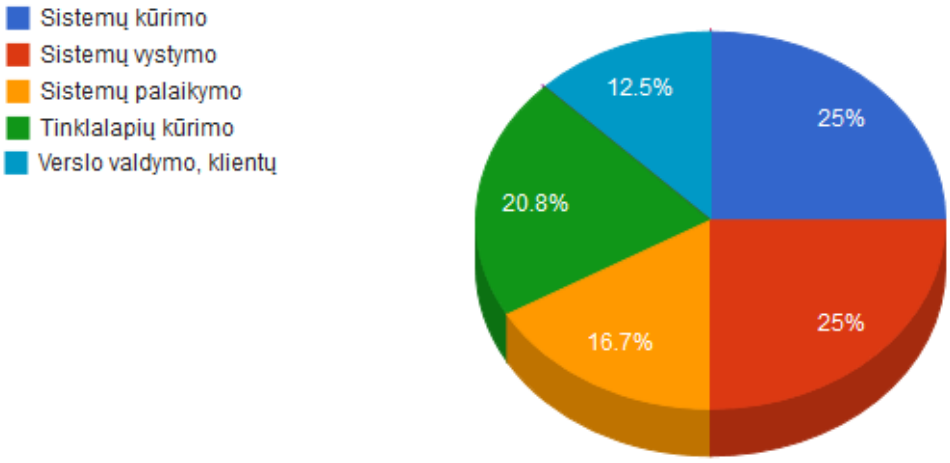
21 pav. *Komandų dydžiai*

Dauguma projektų vadovų (žr. 22 pav.) nurodė, kad vadovauja mažiems projektams (iki 1 metų trukmės). Mažiausiai respondentų nurodė, kad vadovauja dideliems projektas, kurių trukmė ilgesnė nei 2 metai.



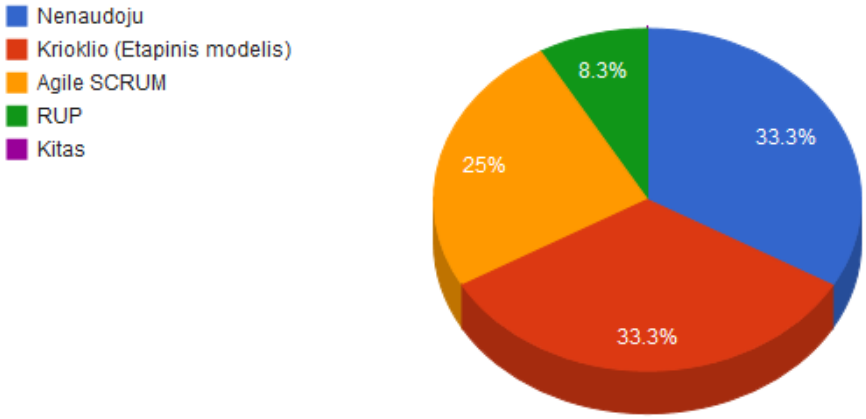
22 pav. *Projektų dydžiai*

Pagal vykdomų projektų tipą, dominuoja sistemų kūrimas, sistemų vystymas, ir tinklalapių kūrimas. Respondentų vadovaujamų projektų įvairovė pateikta 23 paveikslėlyje.



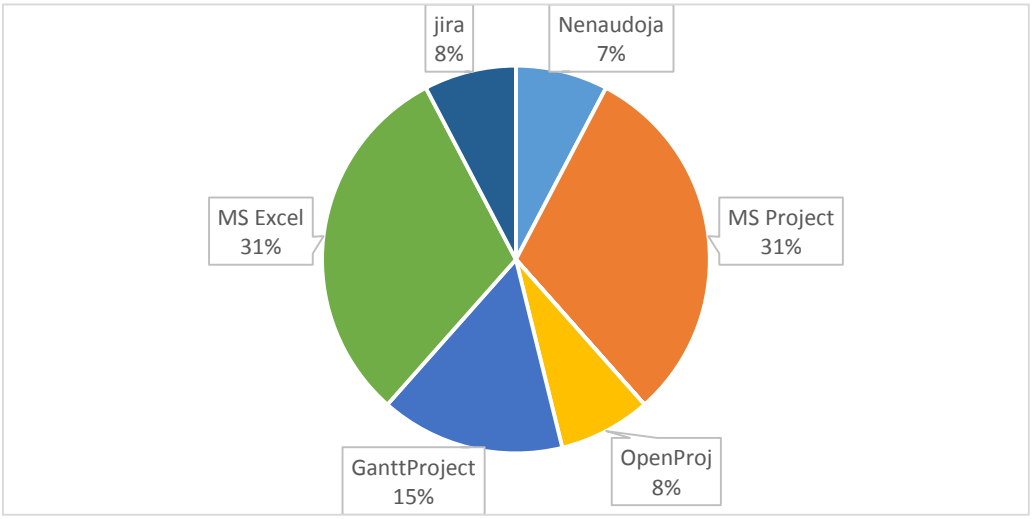
23 pav. *Projektų tipai*

Tyrimo metu nustatyta, kad populiariausi yra krioklio ir SCRUM modeliai. 4 respondentai nurodė, kad PĮ projektų valdymo modelių nenaudoja (žr. 24 pav.).



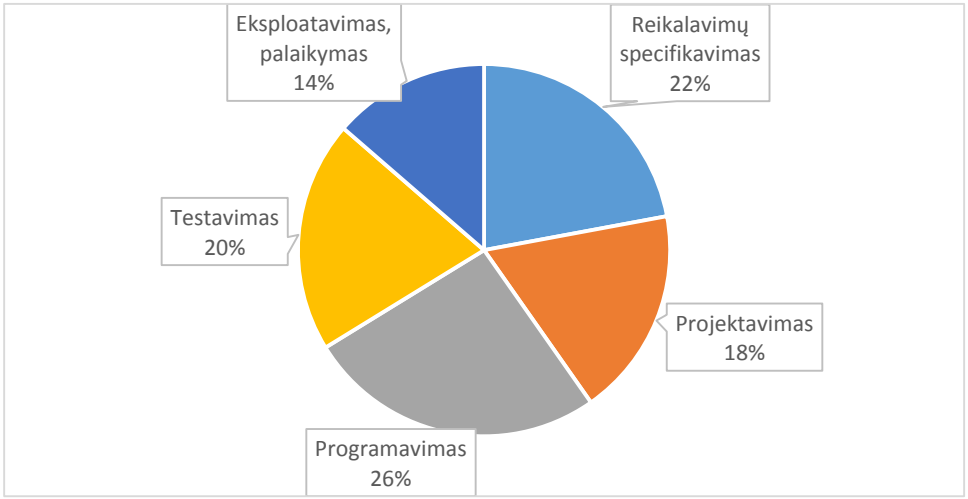
24 pav. *PĮ projektams taikomi modeliai*

Į klausimą apie projektų planavimui naudojamą programinę įrangą, dauguma (žr. 25 pav.) respondentų nurodė, kad naudoja MS Project. Taip pat buvo palikta galimybė nurodyti kitą, jeigu tarp atsakymų variantų nebūtų. Pasinaudodami šia galimybę, 4 iš 10 respondentų nurodė, kad naudoja MS Excel, 1 respondentas nurodė, kad naudoja JIRA projektų planavimo sprendimą.



25 pav. *Projektų planavimui naudojama programinė įranga*

Apibendrinus projekto laiko skyrimo prioritetus projekto etapams, buvo sudaryta skritulinė diagrama pavaizduota 27 paveikslėlyje.



26 pav. *Projekto etapams vidutiniškai skiriamo projekto laiko dalis, procentais.*

Tyrime dalyvavę respondentai kaip projektų nesėkmės priežastis nurodė užsakovo reikalavimų pakeitimas arba neaiškus jų nusakymas, užsakovo darbuotojų sabotažas, trečiųjų šalių vėlavimas, užsakovo bankrotas arba nenoras investuoti.

4. 4 Tyrimo išvados

Tyrimo metu nustatyta, kad apklausoje dalyvavę respondentai dažniausiai projektus vykdo taikant krioklio modelį. 4 respondentai nurodė, kad projektų valdymui netaiko modelių, tačiau jie vien nurodė prioritetus projekto laiko pasiskirstymui etapuose, todėl galima daryti išvada, kad šie projektų vadovai taip pat taiko krioklio (etapinį) modelį.

Pagal tyrimo rezultatus dažniausiai projektų planavimui naudojama Microsoft programinė įranga. 4 respondentai nurodė kad naudoja MS Excel, toks pat respondentų skaičius nurodė kad naudoja MS Project. Šį faktą būtų galima paaiškinti tuo, kad iš esmės MS Project ir Excel skaičiavimams naudoja eilutėse ir stulpeliuose įrašytus duomenis, bet MS Project yra įdiegtos priemonės, kurios pritaikytos būtent projektų valdymui (projekto laiko paskirstymas laike pagal resursų prieinamumą, resursų kolizijos nustatymas, projekto kritinio kelio nustatymas ir daug kitų).

5. IŠVADOS

Išanalizavus PĮ projektų valdymo modelius buvo pastebėta, kad nors jų taikymas aprašytas gana skirtingai, tačiau iš esmės pagal juos projektai plėtojami panašiais etapais. Atsižvelgus į jų tarpusavio panašumą ir išteklių panaudojimo skirtingumą teorinio eksperimento atlikimui buvo pasirinkti krioklio, SCRUM ir RUP modeliai.

Palyginus kelis projektų valdymo įrankius nustatyta, kad MS Project išsiskiria funkcionalumu. Naudojant šią programinę įrangą, darbe teoriškai modeliuoti vidutinio dydžio programinės įrangos projektai taikant visus tris nagrinėjamus modelius.

Atlikus PĮ modelių teorinį tyrimą nustatyta, kad taikant Krioklio modelį, pakitus reikalavimams, projekto trukmė ir kaštai išauga daugiau nei 30%. Mažiausiai resursų ir laiko reikėjo projektui modeliuotam pagal SCRUM modelį, tačiau stipri individuali priklausomybė gali įtakoti projekto baigtį. Taikant RUP modelį, projektui reikėjo daugiausiai resursų ir laiko, tačiau šis modelis įgyvendina įvairias disciplinas eliminuojančias projekto rizikas. Tad galima teigti, kad jeigu modelio taikymas optimalus laiko ir resursų panaudojimo atžvilgiu, jis gali būti neatsparus reikalavimų pasikeitimai.

Remiantis teorinio tyrimo rezultatais buvo sukurtas hibridinis modelis, susiejantis vikriųjų ir tradicinių modelių struktūrą ir išlaikantis dalį jų savybių. Atlikus jo tyrimą pastebėta, nors hibridinio modelio taikymas nebūtų optimalesnis laiko ir finansinių išteklių panaudojimo atžvilgiu lyginant su SCRUM, tačiau sudarytas modelis eliminuoja šio modelio stiprią individualią priklausomybę ir jo taikymas būtų optimalesnis nei kitų nagrinėtų alternatyvų.

Atlikus praktinį tyrimą, nustatyta, kad dauguma apklausoje dalyvavusių Lietuvos IT įmonėse dirbančių projektų vadovų praktikoje projektus vykdo pagal krioklio modelio etapus, juos vykdant iteracijomis. Tai pagrindžia, kad nei įmonės netaiko vieno iš teoriškai nagrinėtų modelių, o adaptuoja juos pagal projekto specifiką.

LITERATŪRA

1. Bareiška E., Krivickas J., Motiejūnas K., Keršienė V., Ambrazas A. Programinės įrangos projektų valdymas K.: Technologija, 2003
2. R. K. Wysocki, Effective Project Managment: Traditional, Agile, Extreme, Fifth Edition,Wiley Publiching, 2009. [žiūrėta 2011-12-12]. Prieiga per internetą: < http://books.google.lt/books?id=AQifszBiJe8C&printsec=frontcover&hl=lt#v=onepage&q&f=true >
3. Project Management Institute (PMI), A Guide To The Project Management Body Of Knowledge. Third Edition. Project Management Institute, 2004.
4. C. Péraire, M. Edwards, A. Fernandes, E. Mancin, K. Carroll, The IBM Rational Unified Process for System z, 2007 [žiūrėta 2011-12-12]. Prieiga per internetą: < http://www.redbooks.ibm.com/redbooks/pdfs/sg247362.pdf>
5. RUP data sheet, 2007 [žiūrėta 2011-12-12]. Prieiga per internetą: < ftp://public.dhe.ibm.com/software/rational/web/datasheets/RUP_DS.pdf>
6. C. Jones, Team Software Process (TSP) In Context, 2011 [žiūrėta 2011-12-12]. Prieiga per internetą: < http://www.sei.cmu.edu/tspsymposium/past-proceedings/2011/Keynote_Jones.pdf>
7. V. Golubeva. Projektų valdymo sistemos modelis ir jo eksperimentinis tyrimas. Magistro darbas.2006 [žiūrėta 2011-12-12]. Prieiga per internetą: <http://vddb.library.lt/fedora/get/LT-eLABa-0001:E.02~2006~D_20060530_234023-59756/DS.005.0.02.ETD>
8. Improving project performance with proven adaptable processes. IBM Rational Unified Process, IBM Corporation, 2007 [žiūrėta 2011-12-12]. Prieiga per internetą: < ftp://public.dhe.ibm.com/software/rational/web/datasheets/RUP_DS.pdf>>
9. H.Kerzner. Project Management– A Systems Approach to Planning, Scheduling, and Controlling. 2009 [žiūrėta 2011-12-12]. Prieiga per internetą: <http://www.google.lt/books?hl=lt&lr=&id=4CqvpWwMLVEC&oi=fnd&pg=PR21&dq=project+management+project+success&ots=LNlOtuxw4r&sig=A01D_bMQj8yx2jyeoJJSGavk8a4&redir_esc=y#v=onepage>
10. R.Pannone. The World of Agile/Lean Product Development and Delivery with Scrum Made Easy. 2009 [žiūrėta 2011-12-12]. Prieiga per internetą:< http://www.slideshare.net/rpannone/the-world-of-agile-and-lean-product-development-with-scrum>
11. David I. Cleland, Roland Gareis (2006). Global project management handbook. McGraw-Hill Professional, 2006. ISBN 0-07-146045-4. p.1-4
12. Laima Miliutė. SCRUM METODO TAIKYMAS KURIANT TARPTAUTINĘ PROGRAMINĘ ĮRANGĄ Laima Miliutė. IšInformacinės technologijos. 2085, Nr. 2(31), p. 214, 215, 216, 217
13. Rational Unified Process. Best Practices for Software Development Teams. Rational Software White Paper. 2003 [žiūrėta 2011-12-12]. Prieiga per internetą:http://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf
14. The Smart Method Limited [žiūrėta 2011-12-12]. Prieiga per internetą:< http://www.learnaccessvba.com/application_development/waterfall_method.htm>
15. Jay M. Siegelaub. Comparing PRINCE2® with PMBoK®. Impact Strategies LLC. 2002. http://impstrat.com/images/Siegelaub_-_Prince2_PMBOK_v2.1_.pdf
16. Managing Successful Projects with PRINCE2 (2009 ed.). The Stationery Office. 2009. ISBN 9780113310593
17. Cobit 4.1. IT Governance Institute. 2011
18. What Is ISO 20000. ISO 20000 Central 2005 Prieiga per internetą:http://20000.fwtk.org/iso-20000.htm
19. ISO/IEC 20000-1:2011(E) Second edition. INTERNATIONAL STANDARD. 2011
20. The Curious Case of the CHAOS Report 2009 [žiūrėta 2012-12-12]. Prieiga per internetą:< http://www.projectsmart.co.uk/the-curious-case-of-the-chaos-report-2009.html>

21. LAUČIUS, J.; VASILECAS, O. Informacinių technologijų projektų ir kokybės valdymas. Vilnius. 2007. 224 p. ISBN 978-9955-28-146-7.
22. BŪDA, V.; CHMIELIAUSKAS, A. Projektų valdymas. Kaunas. 2006. ISBN 978-9955-25-287-0
23. Tata Consultancy: new IT failure stats and COO interview [žiūrėta 2012-12-12]. Prieiga per internetą:<http://www.zdnet.com/blog/projectfailures/tata-consultancy-new-it-failure-stats-and-coo-interview/531>
24. Codebetter.com: software development community [žiūrėta 2012-12-12]. Prieiga per internetą:<http://codebetter.com/patricksmacchia/2012/01/23/mythical-man-month-10-lines-per-developer-day/>
25. CodeForNothing: Jose Simas Blog [žiūrėta 2012-12-12]. Prieiga per internetą:<http://codefornothing.wordpress.com/2009/06/14/the-mythical-one-hundred-lines-of-code-per-day/>
26. Sarah Afzal Safavi, Maqbool Uddin Shaikh. Effort Estimation Model for Each Phase of Software Development Life Cycle. COMSATS Institute of Information Technology. 2011
27. Ye Yang, Mei He, Mingshu Li, Q ing Wang ir Barry Boehm. Phase Distribution of Software Development Effort. Chinese Academy of Sciences, University of Southern California. 2008.
28. David West. Planning a Project with the Rational Unified Process. Rational Software Corporation. 2002.
29. Pete Deemer, Gabrielle Benefield, Craig Larman, Bas Vodde. THE SCRUM PRIMER. Scrum Training Institute. 2010.
30. Mohd Hairul Nizam Nasir, Shamsul Sahibuddin. Critical success factors for software projects: A comparative study. Faculty of Computer Science and Information Technology, University of Malaya. 2011. DOI: 10.5897/SRE10.1171
31. Per Kroll, Philippe Kruchten. The Rational Unified Process Made Easy– A Practitioner's Guide to the RUP. 2003. ISBN-10: 0-321-16609-4
32. L. Tutkutė. MDA IR PROGRAMŲ KODO GENERAVIMAS. Referatas. 2008.
33. cv.lt: sistema skirta specialistų, administracinių darbuotojų, valstybės tarnautojų, profesionalių darbininkų ir papildomo darbo ieškančių žmonių bei darbų paieškai. [žiūrėta 2012-12-12]. Prieiga per internetą:<http://manokarjera.cv.lt/Default4.aspx?ArticleId=f7b77b38-f9fc-4823-94f6-94af7a22f018/>
34. cv.lt: sistema skirta specialistų, administracinių darbuotojų, valstybės tarnautojų, profesionalių darbininkų ir papildomo darbo ieškančių žmonių bei darbų paieškai. [žiūrėta 2012-12-12]. Prieiga per internetą:<http://manokarjera.cv.lt/Default4.aspx?ArticleId=1196f8dd-e980-4f69-973e-4fe636ace882/>
35. W. Heijstek, M. R. V. Chaudron. Effort Distribution in Model-Based Development. Leiden University, Technische Universiteit Eindhoven. The Netherlands. 2008.

PRIEDAI

1 priedas. Kompaktinis diskas