VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ KATEDRA

Įmonės „Mėnuliukų technologijos” programų kūrimo proceso aprašas (Pirmas laboratorinis darbas)

Description of the development process of the „Moon technologies” company (First laboratory work)

Programų kūrimo proceso laboratorinis darbas

Atliko: 4 kurso 3 grupės studentai
Matas Savickis, Justas Tvarijonas, Džiugas Mažulis (parašas)
Greta Pyrantaitė, Andrius Bentkus (parašas)

Darbo vadovas: Saulius Ragaišis, Doc., Dr. (parašas)

Vilnius – 2019

TURINYS

ĮVADAS 2

1. KŪRIMO PROCESAS 3
   1.1. Projekto siūlymas 3
   1.2. Technologinio sprendimo paieška ir įgyvendinimas 4
   1.3. Reikalavimų ciklas 5
   1.4. Sistemos modeliavimas 5
      1.4.1. Mikroservisai 6
   1.5. Sprintas 7
      1.5.1. Sprinto užduočių tikslinimas ir vertinimas 7
      1.5.2. Užduočių analizė 8
      1.5.3. Programavimas 8
      1.5.4. Kokybės užtikrinimas 9
      1.5.5. Konfigūracijos valdymas 10
      1.5.6. Sprinto aptarimas 10
      1.5.7. Defekto analizė 10
   1.6. Išleidimas 11
   1.7. Palaikymas 11
   1.8. Balius/Post-mortem 12

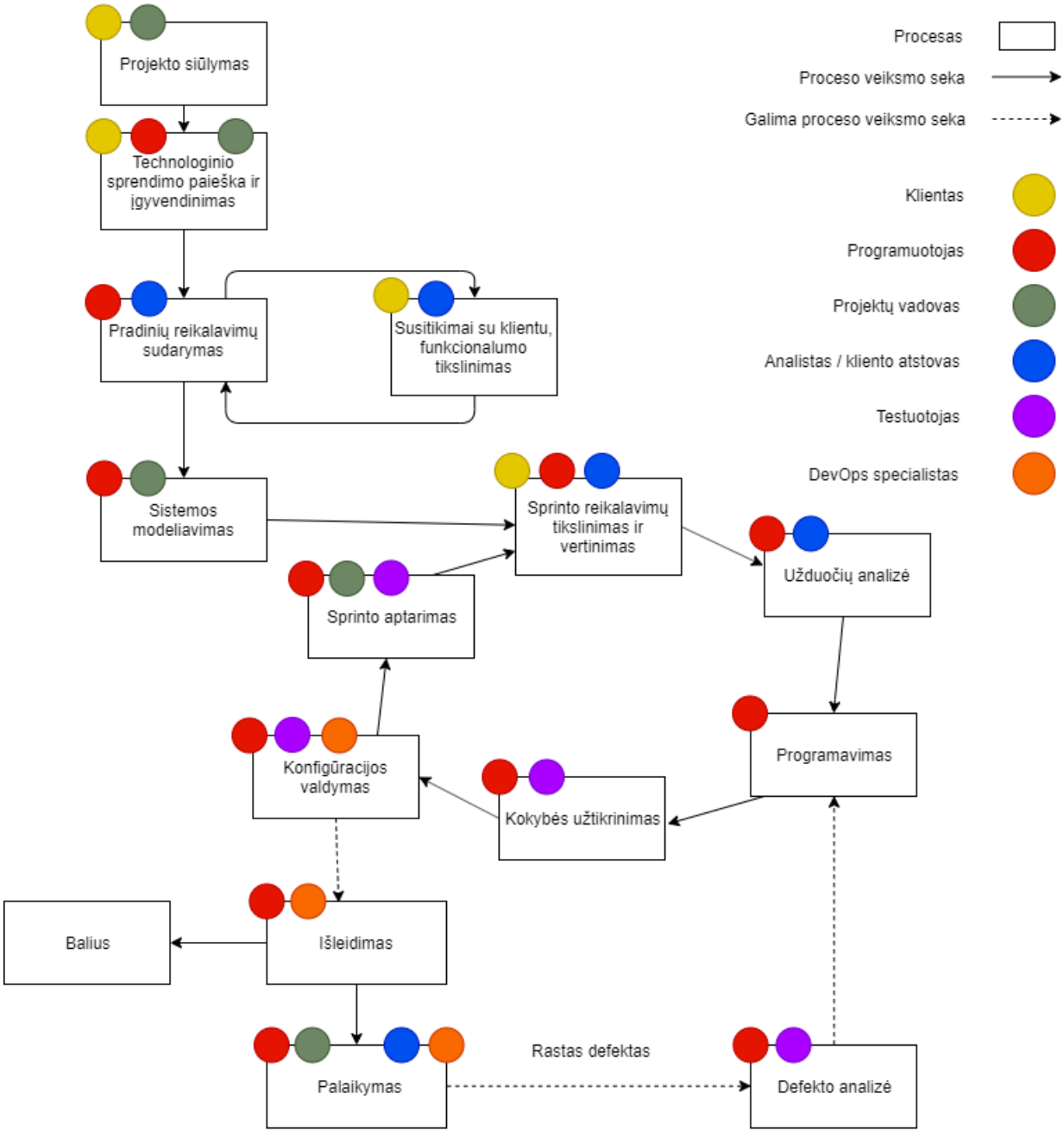
ŽODYNĖLIS 13

Įvadas

Šiame darbe bus pristatytas „Mėnuliukų technologijų" programų kūrimo procesas. Pats procesas yra paremtas Agile metodologija su minimaliais pakeitimais reikalavimų rinkime. Proceso pradžioje stengiamės su užsakovu išsiaiškinti norimus įgyvendinti funkcionalumus ir bendraujant kartu su užsakovu sudaryti reikalavimus. Sudarant reikalavimus yra diskutuojama ir sistemos ateities vizija siekiant susidaryti geresnę perspektyvą sistemos ateičiai ir darbartiniams reikalavimams. Įmonė įvertina kiek valandų užtruks kiekvieno funkcionalumo sukūrimas ir mokestis yra imamas už pradirbtas valandas. Klientui nesutikus su pateiktomis kainomis yra daromi susitikimai siekiant paaiškinti valandų vertinimą. Po susitikimų funkcionalumo įgyvendinimo valandos gali keistis, arba funkcionalumas bus atsisakytas. Kiekvieno sprinto pradžioje, po reikalavimų pasitikslinimo, yra sudaromi priėmimo testai, kuriuos praėjus kliento prašoma susimokėti už atliktus darbus.

1. Kūrimo procesas

Pridėti acceptance testing cikle su programavimu. Iš kokybės užtikrino proceso gali atsirasti naujas defektas.



1 pav. Sistemos kūrimo procesas

1.1. Projekto siūlymas

1. Pirmas susitikimas, kuriame aptariamas galimas projektas, kiekviena pusė išsako savo lūkesčius, pasidalinama įdėjomis.

1 lentelė. Projekto siūlymo procesas

Pavadinimas:	Projekto siūlymas.
Tikslas:	Aptarti galimą projektą su galimu klientu.
Vykdytojai:	Projekto vadovas ir klientas.
Veiklos:	V1 - Aptariama sistemos aprėptis. V2 - Nustatomos kainos ribos iki pirmo sistemos išleidimo. V3 - Pirminės sutarties pasirašymas
Naudojami produktai:	NP1 - Buvusių projektų dokumentai.
Sukuriami produktai:	SP1 - Pirminė sutartis sistemos projektavimui.

2. Po susitikimo įmonė paruošia pradininį pasiūlymą, į kurį įeina orientacinės finansų ribos, žmonių ištekliai, kurie galėtų būtų skiriami šiam projektui. Šis pasiūlymas aptariamas su klientu, kartu su juo dokumentuojami funkcionalumai, kurių klientas nori pirmame sistemos išleidime. Sėkmingai tesiantis tolesnėms deryboms nutariama dėl pradinio technologinio sprendimo pasiūlymo datos, bei finansavimo jam.

3. Pasirašoma pradinė sutartis, kurioje dokumentuojama prieš tai aptarta informacija. Ši sutartis galioja iki pirmojo prototipo pasiūlymo, po kurio atnaujinamos derybės dėl tolesnio projekto vystymo.

1.2. Technologinio sprendimo paieška ir įgyvendinimas

2 lentelė. Technologinio sprendimo paieškos ir įgyvendinimo procesas.

Pavadinimas:	Technologinio sprendimo paieška ir įgyvendinimas
Tikslas:	Išskirti technologijas ir jų versijas kurios bus naudojamos projekte
Vykdytojai:	Projekto vadovas, klientas, programuotojas.
Veiklos:	V1 - Aptarti technologijas šiuo metu naudojamas projekte. V2 - Nustatomos technologijų kainos kurios bus naudojamos projekte. V3 - Nutariama dėl technologinių alternatyvų.
Naudojami produktai:	NP1 - Esamos sistemos dokumentacija, norimų naudoti technologijų dokumentacija ir kainynas.
Sukuriami produktai:	SP1 - Technologinių sprendimų ir jų alternatyvų dokumentas su preliminariomis technologijų licensijų kainomis.

1. Aptarti technologijas šiuo metu naudojamas projekte - išskiriami technologiniai karkasai, duomenų bazės, programavimo kalbų versijos ir kitos naudojamos technologijos ir jų versijos. Įvertinamas esamų technologijų saugumo lygis, greitaveika ir ateities palaikymas. Pasiūlomi technologiniai sprendimai pagrindžiant jų naudą sistemai.

2. Nustatomos technologijų kainos, kurios bus naudojamos projekte - paskaičiuojamas dabartinių technologijų kainos ir naujų siūlomų technologijų kainos.

3. Nutariama dėl technologinių alternatyvų - jeigu įmanoma klientui pasiūloma atviro kodo technologiniai sprendimai siekiant sutaupyti pinigų. Pateikiamas palyginimas tarp dabartinės sistemos technologijos, siūlomos technologijos ir atviro kodo technologijos sprendimų.

1.3. Reikalavimų ciklas

3 lentelė. Reikalavimų ciklo procesas.

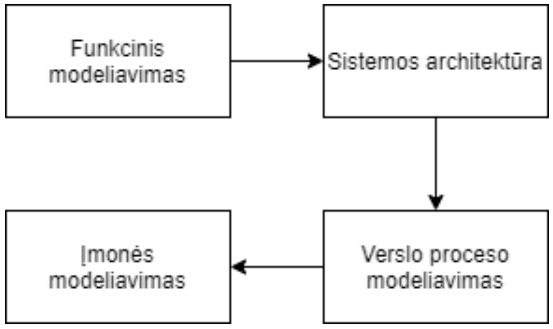
Pavadinimas:	Reikalavimų ciklas
Tikslas:	Suformuoti funkclinius ir nefunkcinius reikalavimus
Vykdytojai:	Programuotojas, analistas, klientas
Veiklos:	V1 - Iš kliento pateiktų verslo reikalavimų suformuojame funkcinius reikalavimus. V2 - Pristatome klientui sudarytus funkcinius reikalavimus ir tiksliname pateiktus verslo reikalavimus V3 - Siūlomi nefunkciniai reikalavimai
Naudojami produktai:	NP1 - Kliento pateiktas verslo reikalavimų dokumentas.
Sukuriami produktai:	SP1 - Funkcinių ir nefunkcinių reikalavimų dokumentas

1. Iš kliento pateiktų verslo reikalavimų suformuojame funkcinius reikalavimus - mūsų įmonės verslo analitikas dirbdamas kartu su programuotojais suformuoja atsekamus(su indentifikacijos kodu) funkcinius reikalavimus

2. Pristatome klientui sudarytus funkcinius reikalavimus ir tiksliname pateiktus verslo reikalavimus - suformavus funkcinius reikalavimus planuojami susitikimai su klientu siekiant jam pristatyti suformuotus reikalavimus, patikslinti pateiktus verslo reikalavimus ir toliau tikslinti reikalavimus iki kol reikalavimai tenkis klientą ir bus suprantami programuotojų komandai, kuri dirbs prie kliento projekto.

3. Siūlomi nefunkciniai reikalavimai - jeigu klientas pats nepateikė nefunkcinių reikalavimų įmonė pati pateikia nefunkcinių reikalavimų siūlymus pagal esamo projekto apimtį ir biudžetą. Pateikti nefunkciniai reikalavimai yra patariami ir tikslinami su klientu.

1.4. Sistemos modeliavimas



2 pav. Sistemos modeliavimas

Modeliavimas vykdomas Ad-hock principu, pasitelkiant praeities žinias arba tiesiog susėdus ekspertams ir bandant išsiaiškinti sprendimą. Programų kūrimo proceso modeliavimą sudaro keturios dalys:

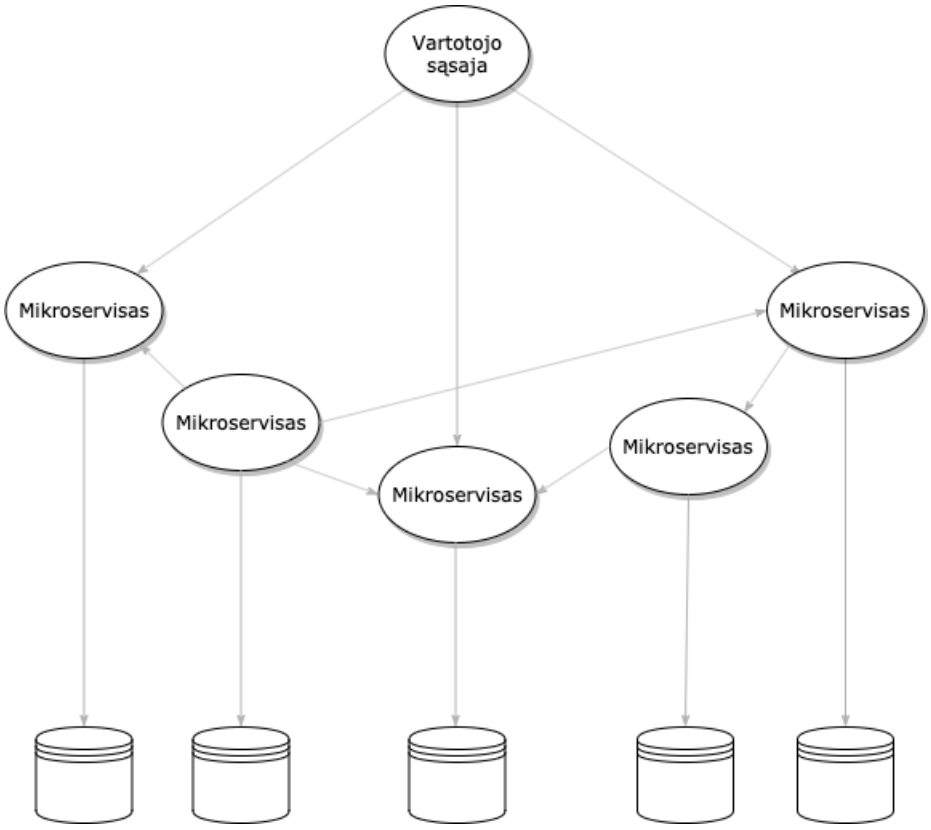
1. Sistemos architektūra - šiame žingsnyje apibrėžiama sąveiką tarp skirtingų sistemos komponentų(duomenų bazių, servisų ir kitą) bei išorinių sistemų. Šiam modeliavimui pasitelkta UML komponentų diagramos. Sistema yra išskaidoma į skirtingus mikroservizus

2. Verslo proceso modeliavimas - šiam modeliavimui naudojami BPMN grafifai apibrėžentys verslo procesus. Sukurti procesai analizuojami, tobulinami. Taip pat išskiriamos verslo proceso dalys kurias galima automatizuoti.

3. Įmonės modeliavimas - abstrakčiai sumodeliuojama kaip įmonės procesai įtakos tolimesnį programos vystymą. Kokie procesų modeliai reikalingi sukurti pseudo kodui. Iš procesų modelių ir duomenų modelių kyla reikalavimai.

1.4.1. Mikroservisai

”Mėnuliukų technologijos” bando atskirti bendrą sistemą į daug mažesnių sistemų remdamasi dabartine populiaria mikroservisu architektūra. Kiekviena atskirta sistema turi būti kuo savarankiškesnė ir atlikti būtent vieną funkcionalumą, tačiau jį turės atlikti kuo efektyviau. Mikroservisai turi itin aiškiai apibrėžtas programų sąsajas bei dokumentaciją, kuri akivaizdžiai apibrėžia funkcionalumą. Taip užtikrinama, jog programų sistemos tikslingai atlieka reikalaujamus verslo siekius.



3 pav. Mikroservisų architektūra

Mikroservisų architektūros teikiami privalumai:

- atskiros sistemos dalys yra lengviau prižiūrimos ir lengviau testuojamos
- silpna sistemos sąsajo - neglaudus sąryšis tarp komponentų bei programų leidžia daryti pakeitimus, kurie nepropaguoja per visą sistemą
- nepriklausomas servisų dalis galima atskirai diegti bei atlikti atnaujinimus nediegiant visos sistemos iš naujo
- Mažos komandos efektyviau prižiūri sistemą, reikia mažiau komunikacijos
- Servisai kuriami pagal įmonės gebėjimus

1.5. Sprintas

Sprintu skaitome dvi savaites, kurių pabaigoje yra įvykdytas verslo analitiko ar komandos parinktas užduočių skaičius ir matomas apčiuopiamas rezultatas - veikiantis funkcionalumas. Sprinto ilgis - 2 savaitės - pasirinktas taip, kad nebūtų sunku numatyti ir suplanuoti užduočių tam laiko tarpui ir taip, kad būtų pakankamai laiko jas įvykdyti iki galo. Sprintas turi kelias veiklas, kurios skirtos palaikyti efektyvią programos kūrimo eigą ir užtikrinti, kad rezultatas būtų pasiektas laiku.

1.5.1. Sprinto užduočių tikslinimas ir vertinimas

4 lentelė. Sprinto užduočių tikslinimo ir vertinimo procesas

Pavadinimas:	Sprinto užduočių tikslinimas ir vertinimas.
Tikslas:	Įvertinti ir paanalizuoti užduotis.
Vykdytojai:	Programuotojas bei Įmonės analistas arba užsakovo atstovas, klientas
Veiklos:	V1 - Analistas paaiškina užduotis iš verslo perspektyvos. V2 - Užduočių vertinimo sesija tarp programuotojų. V3 - Įvertintų užduočių aptarimas su analistu
Naudojami produktai:	NP1 - Užduočių sąrašas.
Sukuriami produktai:	SP1 - Sprinto užduočių sąrašas. SP2 - Įvertintos užduotys backloge(vertimas?)

1. Pradinis užduočių aptarimas su analistu arba užsakovo atstovu, šiame aptarime paaiškinami kiekvienos užduoties scenarijai iš dalykinės srities pusės, paaiškinami reikalavimai, kurie turi būti įgyvendinami prieš pabaigiant užduotį. Diskusijos su programuotojais metu galimi užduočių ar jų priėmimo kriterijų pakeitimai arba, jeigu jų negalima atlikti nepasitarus su klientu, užduotis yra atnaujinama vėliau, pasitarus su klientu.

2. Atlikus pradinį aptarimą kartu su analistu arba verslo atstovu rengiama diskusija tarp programuotojų, kuriame kiekviena užduotis yra įvertinama taškais, vertinama fibonačio sekos skaičiais, o skaičiaus reikšmė yra viena programuotojo darbo diena. Šioje diskusijoje programuotojai diskutuoja apie galimą kiekvienos užduoties implementaciją, bei jos sudėtingumą, tada balsuojama dėl šiai užduočiai skiriamo balų skaičiaus. Taip aptariamos visos dar neįvertintos užduodys, tuo atvėju, jeigu gilinantis į implementaciją atsiranda neaiškumų dėl dalykinės srities, užduotis yra blokuojama paliekant komentarą, kad jį radęs analistas galėtų patikslinti užduotį.

3. Atlikus užduočių vertinimą analistas arba verslo atstovas sudeda sekančio sprinto struktūra pagal galimą talpą (talpa lygi programuotojų skaičiui padauginta iš 8).

1.5.2. Užduočių analizė

5 lentelė. Uzduociu analizė

Pavadinimas:	Užduočių analizė
Tikslas:	Surinkti visą reikalingą informaciją užduočiai įvykdyti
Vykdytojai:	Programuotojas
Veiklos:	V1 - Klausimų, į kuriuos reikia atsakymo, iškėlimas V2 - Informacijos rinkimas, atsakymas į klausimus V3 - Įgyvendinimo alternatyvų aprašymas V4 - Jei reikia, įrodymas, kad alternatyva veiks ar neveiks
Naudojami produktai:	NP1 - Užduočių sąrašas
Sukuriami produktai:	SP1 - Atsakymai į iškeltus klausimus SP2 - Implementacijos alternatyvų sąrašas ir aprašymas SP3 - Jei reikia, pasirinktos alternatyvos supaprastinta implementacija SP4 - Jei reikia, papildytas, patobulintas užduočių sarašas

Turint pilnai aprašytas užduotis vyksta užduočių analizė, jei yra tam poreikis. Jei kyla daugiau neiškumų dėl projekto vykdymo ateities planų gali būti sukuriamos atskiros užduoties tam tikros srities išsiaiškinimui tam, kad geriau išsiaiškinti galimas implementacijos alternatyvas. Tokios analizės rezultatas - dokumentas, pateikiantis klausimus ir atsakymus, implementacijos alternatyvas ir kilusius kitus pastebėjimus. Po analizės turi būti aišku, kaip ir su kokiomis technologijomis užduotis bus įvykdoma ir, jei reikalinga, tikslinamos užduotys.

Jei analizė reikalinga mažesniu mastu, bet dar negalima iškart imti ir programuoti - asmeniškai, jau pasiėmus užduotį, aiškinamasi, kokių žinių trūksta ir kas jas galėtų suteikti. Tai gali būt pasikalbėjimas su kolegomis ar panašių užduočių implementacijos pavyzdžių ieškojimas. Šios veiklos pabaigoje jau galima pradėti progamuoti.

1.5.3. Programavimas

Update

6 lentelė. Programavimo procesas

Pavadinimas:	Programavimas.
Tikslas:	Suprogramuoti sprinto užduotis.
Vykdytojai:	Programuotojai.
Veiklos:	V1 - Kodo rašymas. V2 - Rankinis testavimas. V3 - Automatinių testų rašymas.
Naudojami produktai:	NP1 - Užduočių aprašymas.
Sukuriami produktai:	SP1 - Užduoties kodas. SP2 - Modulių testai. SP3 - Integraciniai testai.

1. Atlikus užduočių analizę ir išsiaiškinus kodo implementacijos kryptį, pradedama rašyti užduoties implementacija. Atlikus implementaciją kodas yra peržiūrymas komandus narių, bet pagal jų rekomendacijas pamodifikuojamas.

2. Atlikus užuodoties implementaciją programuotojas iš pradžių rankiniu būdu pratestuoja ar jo sukurtas funkcionalumas veikia tinkamai, o radęs netikslumų juos ištaiso.

3. Rankinio testavimo būdu pratestavus funkcionaluma yra rašomi automatiniai testai skirti patikrinti sudėtingesnius scenarijus, bei ateities regresiniam testavimui užtikrinti. Į šiuos testus įeina modulių, aptarnavimo bei integraciniai testai.

1.5.4. Kokybės užtikrinimas

Papildyt aprašymą po lentele, pridėt automated testing.

7 lentelė. Kokybės užtikrinimo procesas

Pavadinimas:	Kokybės užtikrinimas.
Tikslas:	Patikrinti sistema su atnaujinta kodo baze veikia teisingai.
Vykdytojai:	Testuotojai ir programuotojai.
Veiklos:	V1 - Leidžiami integraciniai testai. V2 - Testų klaidų analizė ir defektų sūkurimas. V3 - Testų rezultatų apibendrinimas. V4 - Performance testavimas.
Naudojami produktai:	NP1 - Integraciniai testai.
Sukuriami produktai:	SP1 - Nauji defektai. SP2 - Testų rezultatų dokumentas SP3 - Performance analizės dokumentas.

Kokybės užtikrinimas skirtingas kiekvienam projektui. Tuose projektuose, kuriuose kuriamoje sistemoje egzistuoja vartotojo sąsaja, atliekamas rankinis testavimas kartu su automatizuotu regresiniu

testavimu. Į kokybės užtikrinimą yra įtraukiami ir programuotojai, kurie yra atsakingi už automatizuotų testų teisingumą. Testuotojai atsakingi už rankinį testavimą bei regresinių testų rezultatų apibendrinimą. Jeigu projektas yra iteracinis ir jau išleistas plačiam naudojimui, po sėkmingo testavimo kodas yra sudedamas į aukštesnę aplinką, kurioje yra papildomai pravaliduojamas prieš jį išleidžiant į produkciją.

1.5.5. Konfigūracijos valdymas

Update. Nutart, kas pas mus yra konfigūracijos valdymas.

Užtikrinant didesnį sistemos patikimumą reikalingos skirtingos aplinkos skirtos testuoti sistemą, šioms aplinkoms palaikyti atliekamas konfigūracijos valdymas - kiekvieno sprinto metu konfigūracijoje ar duomenų bazėje atliktus pakeitimus programuotojas pažymi tam skirtoje vietoje, pagal kurią po sprinto devOps specialistai atlieka konfigūracinius pakeitimus aplinkose, kuriose šie pakeitimai reikalingi. Taip pat po sprinto kodas yra sudedamas į aukštenę aplinką tolimesniam testavimui.

1.5.6. Sprinto aptarimas

8 lentelė. Sprinto aptarimas

Pavadinimas:	Sprinto aptarimas
Tikslas:	Užtikrinti, kad sprintai būtų efektyvūs
Vykdytojai:	Komanda
Veiklos:	V1 - Problemų iškėlimas V2 - Problemų aptarimas V3 - Iškeliami pasiūlymai sekančio sprinto našumui gerinti
Naudojami produktai:	NP1 - Sprinto užduočių sąrašas
Sukuriami produktai:	SP1 - Užduotys komandai sekančio sprinto efektyvumui gerinti.

Komandos efektyvumui labai svarbu aptarti praėjusį sprintą: kas trukdė efektyviam darbui, kas gerai sekės, kokias programavimo veiklas tęsti, kokias nutraukti, kokios komandos nuotaikos ir to priežastys. Šioje veikloje dažniausiai dalyvauja tik komanda, o jos rezultatas - pagal išreikštus pastebėjimus išskirti tobulėjimo punktai, kurių laikomasi būsimame sprinte siekiant geresnės ir greitesnės projekto implementacijos. Tai gali būti aktyvus kažkokių procesų vykdymo trukdžių sprendimas, komunikacija su kitomis komandomis ar klientu dėl iškilusių problemų.

1.5.7. Defekto analizė

Defekto analizės procesas pradedamas problemos identifikavimu. Vartotojo arba testuotojo aptikas defektas yra užregistruojamas aprašant sąlygas, reikalingas atkurti defektą, bei priskiriamas atitinkamai defektų klasei. Programuotojas išanalizuoja defektą, suranda defekto priežastį bei apibrėžia

9 lentelė. Defekto analizė

Pavadinimas:	Defekto analizė
Tikslas:	Pašalinti defektą
Vykdytojai:	Programuotojai ir testuotojai
Veiklos:	V1 - Defekto registracija V2 - Defekto analizė V3 - Defekto pašalinimas V4 - Patikrinimas ar defektas pašalintas
Naudojami produktai:	NP1 - Esama sistema
Sukuriami produktai:	SP1 - Defektų sąrašas

ir įgyvendina defektui pašalinti reikalingą veiksmų seką. Įvykdžius šiuos žingsnius belieka įsitikinti, kad procesas atnešė pageidaujamą rezultatą - defektas nurodytomis sąlygomis nepasikartoja.

1.6. Išleidimas

Update

Išleidimo stadiją galima skaidyti į 2 skirtingus tipus - galimas naujas projekto išleidimas, kurio metu vartotojui pateikiama nauja sistema, kuri buvo tam tikrą laiką kuriama. Taip pat egzistuoja kitas variantas, kai yra veikianti sistema, kuri yra atnaujinama kas tam tikrą laiką (tai priklauso nuo sprinto ilgi). Išleidimo metu įmonėje yra vykdomi darbuotojų budėjimai, kad sistemai veikiant ne tiksliai pagal planą, sistemos trikdžiai būtų kuo greičiau pašalinti.

1.7. Palaikymas

10 lentelė. Palaikymo procesas

Pavadinimas:	Palaikymas
Tikslas:	Užtikrinti korektišką sistemos veikimą po sistemos paleidimo
Vykdytojai:	Projektų vadovas, programuotojas, analistas, DevOps specialistas
Veiklos:	V1 - Analizuoti kliento pateiktus palaikymo darbus V2 - Registruoti palaikymo darbus V3 - Perduoti darbus defekto analizės procesui
Naudojami produktai:	NP1 - Esama sistema NP2 - Sistemos palaikymo sutartis NP3 - Užregistruoto defekto informacija NP4 - Vartotojo užregistruota palaikymo užduotis
Sukuriami produktai:	SP1 - Defekto aprašymas SP2 - Palaikymo darbo aprašymas

Palaikymo procesui kuriamas palaikymo planas, susidedantis iš programos paruošimo, problemos identifikavimo bei produkto konfigūracijos valdymo. Problemos identifikavimas vykdomas tikrinant programos validumą, sukuriant problemos sprendinį bei išskiriant resursus modifikacijai

įgyvendinti. Proceso patvirtinimas įgyvendinamas gavus patvirtinimą dėl kėtinamų įgyvendinti pakeitimų iš užklausos autoriaus. Įmonė teikia dviejų tipų palaikymą: taisomąjį bei adaptacinį. Taisomasis palaikymas orientuotas į problemų, atrastų vartotojų arba vartotojų klaidų reportų analizės metu, taisymą. Adaptacinis palaikymas skirtas nūdienos standartų programose palaikymui. Įmonė laikosi „Boehm“ modelio, kuris pasižymi atitinkamai pakeitimų pasiūlymu, patvirtinimu bei įgyvendinimu.

1.8. Balius/Post-mortem

Procesas užbaigiamas komandos bei prie projekto prisidėjusių žmonių švente, kurioje aptariamas bei įvertinamas proceso pasisekimas ir daromos išvados.

Žodynėlis

- Klientas