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

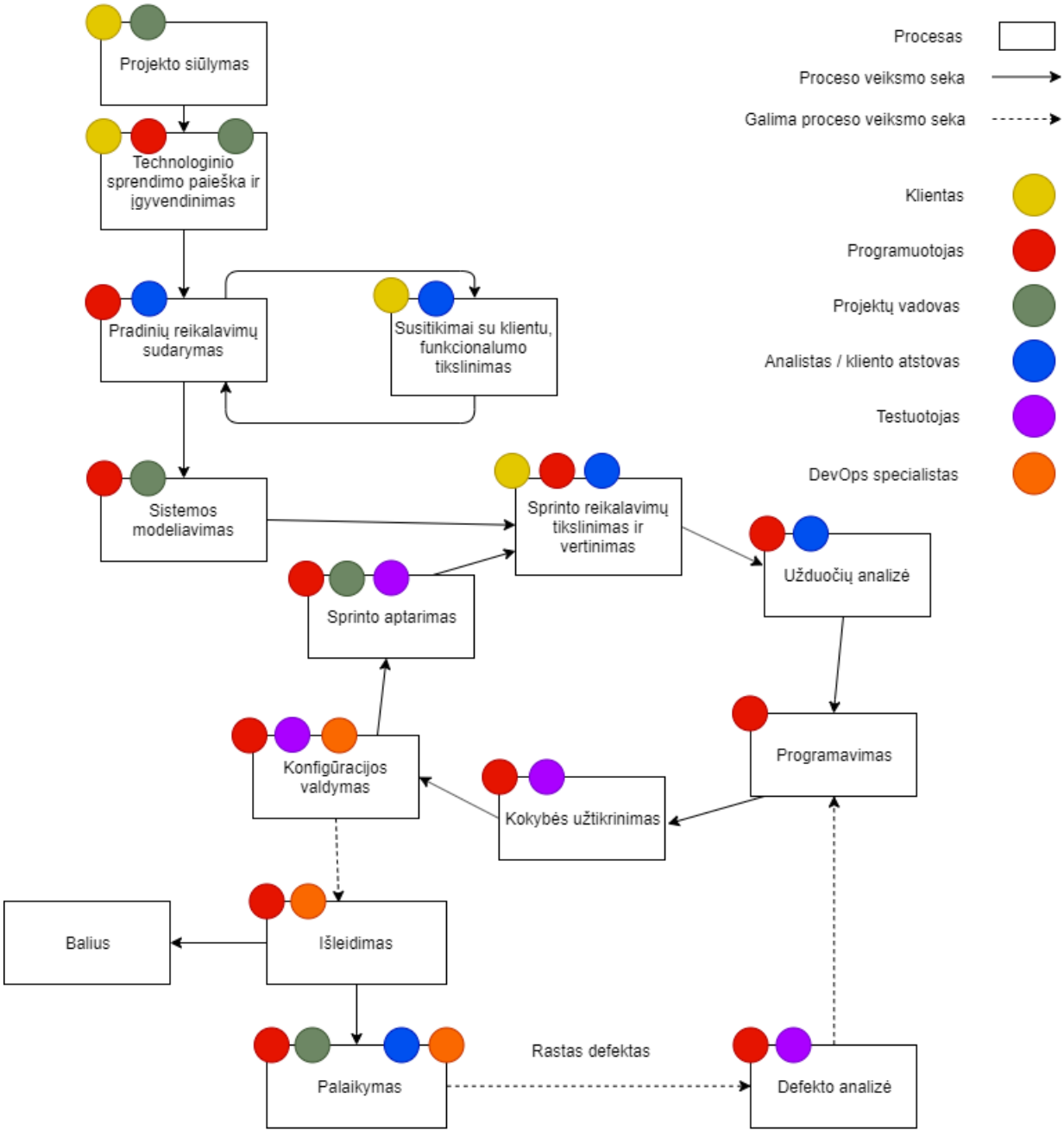
TURINYS

Įvadas

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.

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

update

Pasirašius sutartį su klientu pradedama ieškoti konkretaus technologinio sprendimo tinkamo projekto vystimui. Sutatomos konrečios karkasų, duomenų bazių arba betkokios kitos technologijos versijos kurios bus naudojamos. Jeigu projektas yra jau egzistuojantis ir klientas perka tolimesnį projekto vystimą yra išsiaiškinama ar nereikės pakelti projekte naudojamų technologijų versijos. Projektuojant sistema siekiama patenkinti arba siekti išvardintas charakteristikas:

- Intersuotų asmenų įvairovė - sistema turis patenkinti įvairių interesuotų asmenų norus ir poreikius, pavyzdžiui vadovai, savininkai, vartotojai ir valdytojai.

- Interesų atskyrimas - bandoma atskirti interesai taip, kad lengviau būtų realizuoti galutinėje sistemoje.

- Remiamasi kokybės užtikrinimu - Bandoma taip pat siekti nefunkcinių reikalavimų užtikrinimu pavyzdžiui užtikrinti greitaveiką.

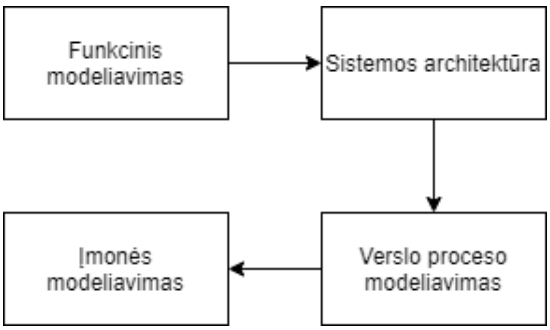
1.3. Reikalavimų ciklas

update

Nutarus dėl konkrečių technologijų, kurios bus naudojamos pradedame funkcinių ir nefunkcinių reikalavimų sudarymas. Jų sudarymas vyksta ciklišKai, pirma mūsų įmonės verslo analitikas

išanalizuoja verslo poreikius ir kartu su architektu sudaro pirminius reikalavimus, jie yra pristatomi klientui kartu su klausimais, įvyksta suformuotų reikalavimų aptarimas. Aptarime dalyvauja architektas, verslo analistas ir verslo žmogus. Jeigu klientas sutinka su sudarytais reikalavimais pradedamas sistemos modeliavimas, jeigu kyla neaiškumų dėl reikalavimų tarp kliento ir mūsų įmonės grįžtama prie poreikių sudarimo įmonės viduje. Ciklas vyksta iki tol kol pasiekiamas sutarimas tarp mūsų įmonės ir kliento.

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

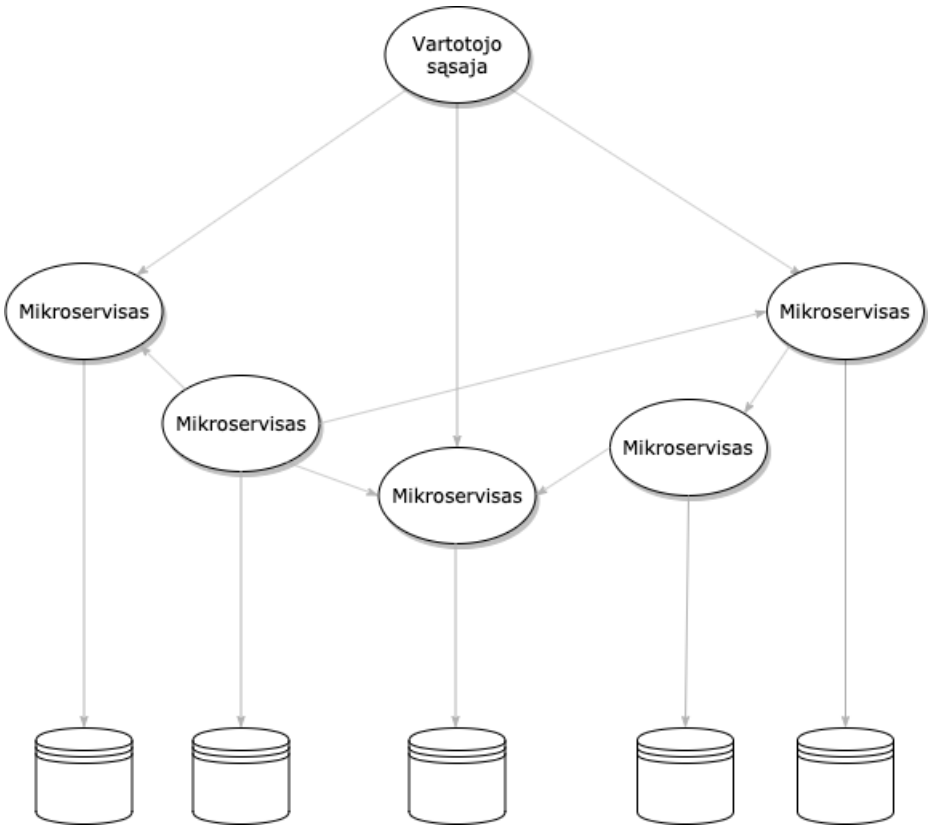
2. Verslo proceso modeliavimas - šiam modeliavimui naudojami BPMN grafifai apibrėžentys verslo procesus. Sukurti procesai analizuojami, tobulinami. Taip pat išskiriamos verslo proceso dalys kurias galima automatizuoti.

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

Šitą matyt šalinam?

”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.

2 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?)

1.5.1. Sprinto užduočių tikslinimas ir vertinimas

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 atveju, 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).

Ką darom su šitu tekstu? Tekstas gražus, bet nežinau kaip įkomponuot.

Tai viena svarbiausių fazių, kadangi jos kokybiškas įvykdymas padeda pagrindą visam spintui - jei reikalavimai netikslūs ar užduotys prastai aprašytos ir jų sudėtingumas atmestinai įvertintas - sprintas pasmerktas nesusipratimams ir laiko gaišimui aiškinantis, koks sprinto tikslas ir koks jo sėkmingo baigimo vertinimas. Sprinto reikalavimų tikslinime ir vertinime dalyvauja komanda ir verslo analitikas. Verslo analitikas pateikia reikalavimus ir išskirtas užduotis, kartu su komanda tikslina priėmimo kriterijus tol, kol susidaro pilnos užduotys su tiksliais užduoties sėkmingo įvykdymo reikalavimais ir klausimais klientui jei iškyla neaiškumų. Komanda taip pat įvertina aptariamų užduočių sudėtingumą ir kartu su verslo analitiku nusprendžia, kokias ir kiek užduočių įdėti į sekantį sprintą.

1.5.2. Užduočių analizė

Update

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

Šis žingsnis glaudžiai susijęs su prieš jį einančiomis fazėmis, jame vykdomas suprojektuotos sistemos dalies įgyvendinimas, kuris apima kodo rašymą, automatizuotų testų kūrimą, konfigūracijos pakeitimus ir duombazės kūrimą. Ši fazė įgyvendinama tik gerai išsiaiškinus ir susidokumentavus reikalavimus ir jau turint prototipą, kad kuriama programa atitiktų kliento lūkesčius ir reikalingų pakeitimų būtų mažiau.

1.5.4. Kokybės užtikrinimas

Papildyt aprašymą po lentele, pridėt automated testing.

3 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. NP2 - ?
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

Update. Kas dalyvauja?

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. Šios veiklos 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

Update

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 ir įgyvendina priežasčiai išspręsti reikalingą veiksmų seką. Įgyvendinus šiuos žingsnius belieka įsitikinti, kad procesas atnešė pageidaujamą rezultatą - defektas nurodytomis sąlygomis neegzistuoja.

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

Update

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.

Rezultatai ir išvados

Ar šito skyriaus čia iš vis reikia?

Šarūnas minėjo, kad Ragaišis visai glossary nori.