Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Eva Lužnik Žnidaršič

Ocenjevanje uspešnosti razvoja programske opreme z upoštevanjem vpliva organizacijskih karakteristik

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: doc. dr. Tomaž Hovelja
Somentor: izr. prof. dr. Damjan Vavpotič

Ljubljana, 2018

Zahvala

Zahvaljujem se mentorju in somentorju za pomoč in nasvete pri izdelavi magistrskega dela. Zahvaljujem se družini, prijateljem in študijskim kolegom za vso podporo, pomoč in uspodbudo tekom študija. Zahvaljujem se vsem zaposlenim v podjetjih, vključenih v študijo, za aktivno sodelovanje.

Eva Lužnik Žnidaršič, 2018

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
FDD	Feature-Driven Development	funkcionalno voden razvoj
iOS	iPhone operating system	iPhone operacijski sistem
IS	information system	informacijski sistem
IT	information technology	informacijska tehnologija
PMI	Project Management Institute	Inštitut za projektno vodenje
RUP	Rational Unified Process	ogrodje objektnega procesnega modela
XP	Extreme programming	ekstremno programiranje

Povzetek

Naslov: Ocenjevanje uspešnosti razvoja programske opreme z upoštevanjem vpliva organizacijskih karakteristik

Pri razvoju programske opreme prihaja do zamud, neuspehov, zavrnitev in opuščanja projektov, zato se poraja težnja po izboljšanju tega procesa. V strokovni literaturi se modeli ocenjevanja uspešnosti razvoja programske opreme večinoma osredotočajo na tehnične vidike kakovosti, kar za celosten vpogled ne zadostuje. Cilj magistrskega dela je zapolniti vrzel na področju vrednotenja metodologij razvoja programske opreme z modelom, ki pri ocenjevanju uspešnosti vključuje tudi skladnost z organizacijskimi karakteristikami podjetja. Definirali smo ključne organizacijske karakteristike in razvili model, ki celostno ocenjuje uspešnost razvoja. Model smo preizkusili z empirično študijo in na podlagi analize rezultatov potrdili osnovno hipotezo vpliva organizacijskih karakteristik na proces razvoja programske opreme. Ugotovitve in ključne predloge za izboljšavo skladnosti organizacijskih karakteristik in procesa dela smo predstavili vodstvu podjetij, vključenih v študijo, in na podlagi njihovih odzivov potrdili koristnost razvitega modela.

Ključne besede

metodologije razvoja programske opreme, organizacijske karakteristike, merjenje uspešnosti, razvoj programske opreme

Abstract

Title: Evaluation of software development success considering the impact of organizational characteristics

In the software development process, there are delays, failures, rejections and abandoned projects, so there is a clear need for the improvement of this process. The models for assessing the success of software development focus primarily on technical aspects of quality, which is not sufficient for comprehensive insight. The goal of the master's thesis is to fill the gap in the field of software development methodologies evaluation with the model that in performance assessment considers also the organizational characteristics of the company. The model was tested with an empirical study resulting in the confirmation of the basic hypothesis regarding the impact of organizational characteristics on the software development process. The findings and key suggestions for improving the fit between organizational characteristics and the work process were presented to the management of the companies involved in the study. Their responses have confirmed the usefulness of the developed model.

Keywords

software development methodologies, organizational characteristics, measuring success, software development

Poglavje 1

Uvod

Kljub pomembnosti programske opreme v modernem svetu proces razvoja programske opreme ni popoln. Ker razvoj programske opreme ni dosledno uspešen proces, pogosto prihaja do zamud, neuspešnih, opuščenih ali zavrnjenih projektov [17]. Izziv, ki se pojavi, je, kako izboljšati uspešnost razvoja programske opreme. Z namenom izboljšanja procesa razvoja je bilo v zadnjih desetletjih razvitih veliko različnih metodologij. Uporaba ustrezne metodologije projektnega vodenja ima pri izboljšanju uspešnosti procesa razvoja vedno pomembnejšo vlogo [64]. Ključno za izbiro ustrezne metodologije projektnega vodenja za določen proces razvoja programske opreme je ovrednotenje metodologij. Ker se tradicionalne metodologije ne prilagajajo dovolj posamezni projektni skupini in specifičnim projektom, jih je težje vpeljati. Zaradi tega organizacije zelo redko uporabljajo metodologije v celoti in pogosto uporabijo le posamezni del [36]. Da bi to stanje izboljšali, so raziskovalci pričeli iskati načine za načrtovanje, sestavljanje in prilagajanje metod, tehnik in orodij za razvoj programske opreme, ki so ga poimenovali "Method engineering" [15]. Raziskovanje kakovosti razvoja programske opreme se v veliki meri osredotoča na tehnične vidike kakovosti. Ker tehnični vidiki ne dovoljujejo celostnega vpogleda, sta Vavpotič in Bajec raziskovanje kakovosti dopolnila še s sociološkimi vidiki, ki pomembno vplivajo na uspešnost metod [73]. Še vedno obstaja vrzel na področju raziskovanja kako-

vosti, saj organizacijskim karakteristikam posvečamo dovolj pozornosti, kljub ugotovitvam v literaturi, da so trenutni izzivi, s katerimi se soočajo pri izboljševanju uspešnosti razvoja programske opreme, predvsem organizacijske in ne tehnične narave [59]. Obstoječe ocenjevanje uspešnosti metodologij razvoja programske opreme temelji torej predvsem na tehničnih, ekonomskih in sociooloških vidikih, znano pa je, da bi bilo potrebno upoštevati tudi organizacijske dimenzije podjetja. Že leta 1993 sta Henderson in Venkatraman [31] objavila model za ugotavljanje strateške skladnosti, kjer sta ugotovila, da morajo biti poslovna in informacijska strategija, organizacijska infrastruktura in procesi, ter informacijska infrastruktura in procesi med seboj usklajeni. Kljub ugotovljeni potrebi po tej uskladitvi so se raziskovalci zaenkrat osredotočili predvsem na strateški nivo in celostne usklajenosti niso dobro definirali.

Cilj magistrskega dela je zapolniti vrzel na področju vrednotenja metodologije razvoja programske opreme z modelom, ki pri ocenjevanju uspešnosti metodologije vključuje tudi organizacijske karakteristike podjetja. V ta namen je potreben pregled literature s področja organizacijskih karakteristik, projektne metodologije in ocenjevanja uspešnosti razvoja programske opreme na splošno. Potrebno je definirati organizacijske karakteristike in postaviti celosten model ocenjevanja uspešnosti razvoja programske opreme. Nato je potrebno s pomočjo empirične študije preveriti delovanje, uporabnost in koristnost razvitega modela za potrebe izboljšanja procesa razvoja programske opreme.

Magistrsko delo se začne s pregledom relevantne literature. V Poglavju 2 so predstavljene ključne smernice metodologij projektnega vodenja na področju razvoja programske opreme. Prav tako je narejen tudi pregled literature s področja organizacijskih karakteristik in definirane ključne organizacijske karakteristike, ki pomembno vplivajo na uspešnost uporabe informacijskih tehnologij v podjetjih. Predstavljena je literatura o najbolj razširjenih modelih ocenjevanja uspešnosti na področju informacijskih tehnologij.

Sledi Poglavje 3, kjer je podrobno predstavljen model, ki smo ga raz-

vili. Podrobno so opisane posamezne faze razvitega modela za ocenjevanje uspešnosti programske opreme. Definirane so organizacijske karakteristike, ki so uporabljene v modelu. Prav tako so predstavljene faze razvoja programske opreme, ki so uporabljene v razvitem modelu. Opisani so tudi kriteriji za ocenjevanje uspešnosti, ki jih razviti model uporablja.

Poglavje 4 se osredotoča na preizkus razvitega modela, ki smo ga opravili s pomočjo pluralne študije primera. Podjetji, v katerih je bila študija izvedena, sta podrobno predstavljeni. Pri tem so še posebej natančno predstavljeni procesi dela, postopek izvedbe študije in rezultati. V nadaljevanju so predstavljeni predlogi za izboljšavo procesa razvoja programske opreme pri izbranih podjetjih, ki so oblikovani na podlagi rezultatov študije. V zaključku poglavja so predstavljeni odzivi in povratne informacije, pridobljene s strani vodstva obeh podjetij.

Zadnje, Poglavje 5, magistrsko nalogo zaključuje s sklepi. Na podlagi izvedene študije in odzivov vodstva podjetij, vključenega v raziskavo, sklepa o koristnosti razvitega modela za ocenjevanje uspešnosti razvoja programske opreme. Opisan je tudi razmislek o omejitvah modela in predlogi za nadaljnje študije.

Poglavje 2

Pregled literature

V tem poglavju smo pripravili pregled relevantne literature s področja metodologij, organizacijskih karakteristik in merjenja uspešnosti razvoja programske opreme. Namen magistrskega dela je ocenjevanje uspešnosti razvoja programske opreme z upoštevanjem vpliva organizacijskih karakteristik, tako da smo se posvetili predvsem literaturi o različnih metodologijah dela na področju razvoja programske opreme. Poseben poudarek smo namenili literaturi o organizacijskih karakteristikah podjetja in definiranju le-teh. Pregledali smo tudi literaturo o ocenjevanju uspešnosti razvoja programske opreme, ki nam je pomagala pri postavljanju modela.

2.1 Metodologije razvoja programske opreme

Že desetletja se razpravlja o tem, kako organizirati razvoj programske opreme, da bi zagotovili hitrejše, kvalitetnejše in cenejše rešitv [28]. Predlagane so bile številne izboljšave, od standardizacije in merjenja procesov programske opreme, do številnih konkretnih orodij, tehnik in praks [28]. Terminologija in smernice projektnega vodenja so utemeljene s standardom "Project Management Body of Knowledge", ki ga definira Project Management Institute (PMI) [60]. Izbira metodologije, ki jo organizacija uporablja, vpliva na njihove organizacijske karakteristike in vice versa. Metodologija vpliva

na prilagodljivost spremembam v okolju na eni strani in optimiziranost na drugi strani [21]. V tem poglavju bomo predstavili nekatere izmed najbolj uveljavljenih metodologij razvoja programske opreme.

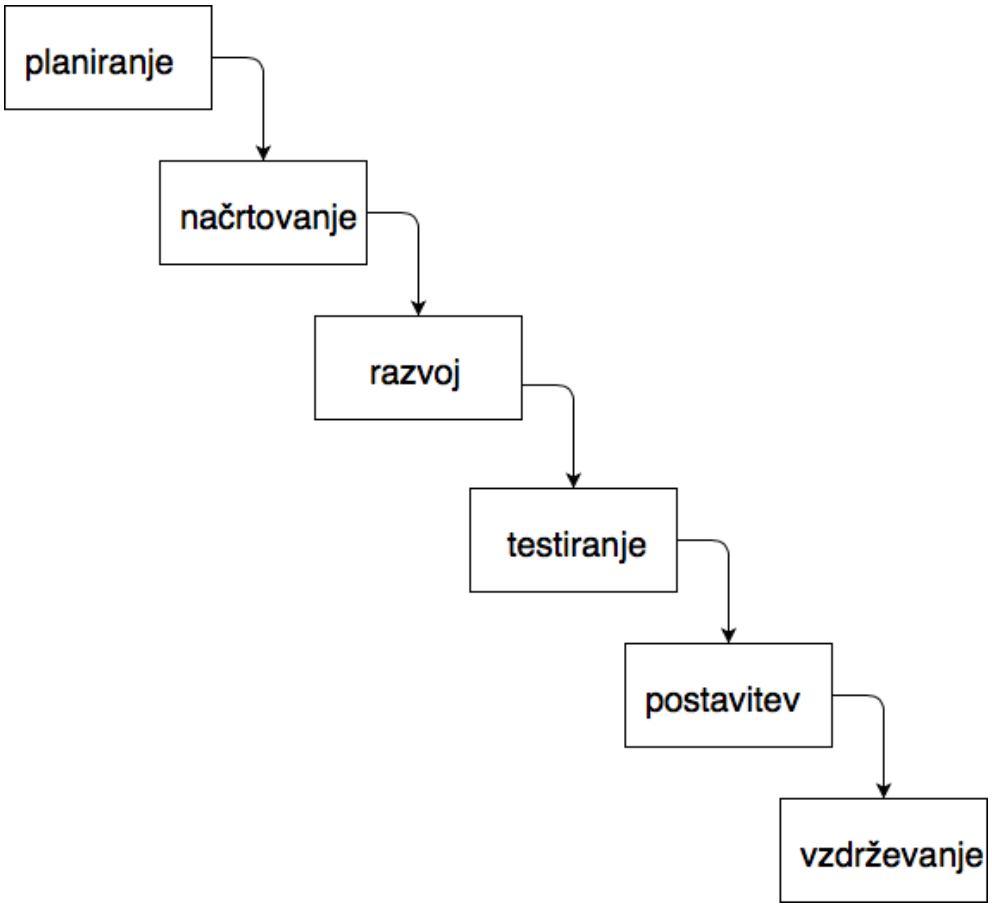
2.1.1 Slapovni model

Leta 1970 je Winston W. Royce [61] opisal metodologijo za razvoj programske opreme, ki so jo kasneje poimenovali slapovni model oziroma "Waterfall". Gre za tradicionalno metodologijo razvoja programske opreme, kjer so vse procesne faze (planiranje, načrtovanje, razvoj, testiranje, postavitev in vzdrževanje) izvedene v zaporednem vrstnem redu [20]. Procesne faze slapovnega modela so predstavljene na Sliki 2.1. Vsaka nadaljnja faza se začne šele, ko se je prejšnja zaključila. Glavna pomanjkljivost slapovnega modela je, da se ni možno vrniti nazaj v proces, da bi na primer uvedli spremembo razvojnih zahtev. Metodologija narekuje, da so zahteve jasno definirane že v fazi definicije sistemskih zahtev. Povratne informacije glede prejšnje faze procesa je tako zelo težko upoštevati.

Kljub mnogim pomanjkljivostim je slapovna metodologija še vedno zelo pogosto uporabljena v podjetjih, ki razvijajo programsko opremo [44]. Glavni problemi metodologije so med drugim:

- Priprava dokumentacije za vsako fazo je zelo časovno in finančno potratna [47]. Spremembam se je zelo težko prilagoditi.
- Če je potrebno narediti novo iteracijo določene faze, to zahteva veliko ponovnega dela [38].
- Ker se testira šele na koncu projekta, lahko pride do nepredvidenih težav s kvaliteto, visokimi stroški in časovnimi prekoračitvami.
- Pomanjkljivosti določene faze se prenesejo dalje in jih je potrebno reševati v kasnejših fazah [68].

Zaradi linearnosti modela je to metodologijo preprosto vpeljati v podjetje in posledično potrebuje manj sredstev za izvajanje kot druge metode [8].



Slika 2.1: Slapovni model [8].

Dobra stran metodologije je tudi, da so zahteve jasne pred začetkom razvoja in da se pri vsaki fazi ustvari ustrezna dokumentacija.

2.1.2 Rational Unified Process

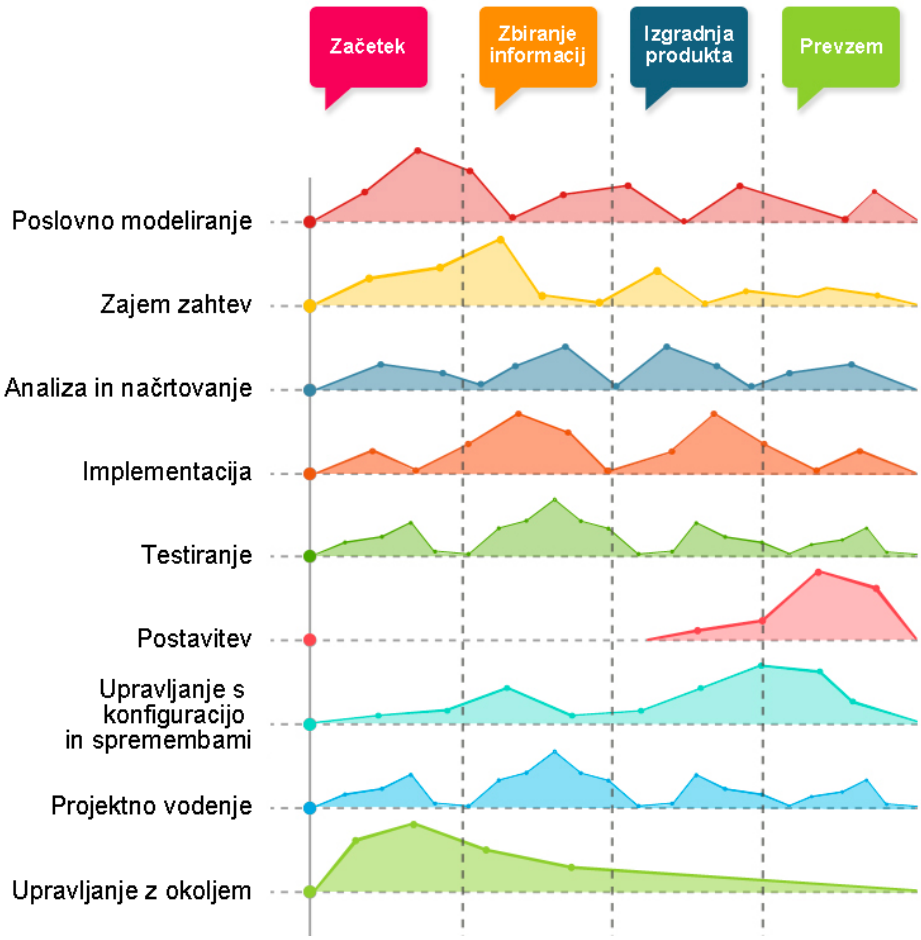
Rational Unified Process (RUP), ki ga je leta 1999 izdelalo podjetje Rational Software Corporation [56], del podjetja IBM, je prilagodljiva metodologija razvoja programske opreme. Organizacije redko uporabljajo metodologijo v celoti, pogosto uporabijo le posamezni del. Metodologija RUP razvojni cikel razdeli v iterativne cikle, kjer vsak cikel dela na novi verziji produkta. Vsak cikel je razdeljen na 4 faze, kjer ima vsaka faza vsaj en cilj oziroma tako imenovani mejnik [57]. Te faze so začetek (angl. Inception), zbiranje informacij (angl. elaboration), izgradnja produkta (angl. construction) in prevzem (angl. transition). Potek dela je razdeljen v 6 glavnih delov: poslovno modeliranje, zajem zahtev, analiza in načrtovanje, implementacija, testiranje in postavitev. Poleg glavnih so še 3 podporni deli: upravljanje s konfiguracijo in spremembami, projektno vodenje in upravljanje z okoljem. Model je predstavljen na Sliki 2.2.

Ključne deli razvojnega procesa so [71]:

1. **Poslovno modeliranje:** Cilj faze je dokumentirati poslovne procese z uporabo tako imenovanih poslovnih primerov uporabe, ki se nato analizirajo za boljše razumevanje poslovnih procesov. Veliko projektov se odloči izpustiti to fazo.

2. **Zajem zahtev:** Cilj faze je opisati, kaj mora sistem početi. V ta namen se pogovarja s stranko, dogovarja o željah, organizira in dokumentira zahtevane funkcionalnosti in omejitve. Sledi in zapisuje se odločitve in kompromise.

3. **Načrtovanje arhitekture:** Cilj faze je izdelati načrt, kako bo sistem realiziran v fazi implementacije. Namen je načrtovati sistem, ki bo v določenem razvojnem okolju izvajal naloge in funkcionalnosti določene



Slika 2.2: RUP model [43].

v fazi zajema zahtev, se držal vseh zahtev, bo stabilen in fleksibilen (če pride do spremembe zahtev, ga je možno enostavno spremeniti).

4. **Implementacija:** Cilj faze je realizirati sistem po načrtih narejenih v fazi načrtovanja arhitekture. Sistem se realizira z implementacijo posameznih sestavnih delov.

5. **Testiranje:** Cilj faze je preveriti medsebojno vplivanje objektov, pravilno vključevanje vseh sestavnih delov programske opreme in pravilno implementacijo vseh zahtev iz faze zajema zahtev. Cilj faze je identificiranje in naslavljanje napak pred fazo postavitve.

6. **Postavitev (angl. deployment):** Cilj faze je uspešna objava produkta in izročitev končnim uporabnikom. Vključuje dejavnosti kot so objava, predstavitev, distribucija, nameščanje programske opreme in nudenje pomoči uporabnikom.

Metodologija RUP omogoča, da razvojna ekipa prej in bolj pogosto pride do rešitev, kar je posledica iterativnega razvojnega pristopa [51]. Tak način dela tudi pozitivno vpliva na količino povratnih informacij, ki jih razvojna ekipa prejme od stranke, kar pomaga prilagoditi programsko opremo pričakovanjem strank. Faze in mejniki pozitivno vplivajo na nadzor izvajanja projekta. Dejavnosti v fazi začetka in zbiranja informacij povečujejo predvidljivost izvajanja projekta, ker obravnavajo poslovna in tehnična tveganja. Kljub temu ima metodologija tudi nekaj pomanjkljivosti. Ugotovljeno je bilo, da obstaja omejitev pri merjenju in poročanju o poteku izvajanja projekta. Zaradi velikega obsega povratnih informacij ima razvojna ekipa povečan obseg dela, kar podraži izvedbo in vodenje projektov. Uvedba RUP zahteva spremembo miselnosti vodij, saj se morajo navaditi na iterativno planiranje in že na začetku določiti merila za končanje faz. RUP je obširna metodologija, ki jo je potrebno prilagoditi za učinkovito uporabo. Prilagajanje RUP metodologije pa je drag in zapleten proces, ki zahteva zaposlene z izkušnjami in znanjem o RUP.

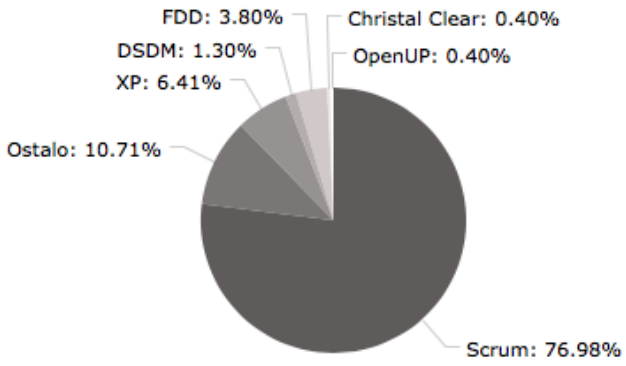
2.1.3 Scrum

Izraz Scrum izhaja iz študije, ki sta jo leta 1986 v Harvard Bussiness Review objavila Takeuchi in Nonaka [70]. Leta 1993 je Jeff Sutherland s pomočjo omenjene študije razvil proces Scrum v podjetju Easel Corporation [69]. Leta 1995 je Ken Schwaber še formaliziral razvoj programske opreme po metodi Scrum [62].

Scrum je preprosto agilno ogrodje, ki se prilagaja okoliščinam, tudi takšnim, ki niso povezane z razvojem programske opreme. Definira tri vloge, to so lastnik izdelka (angl. product owner), skrbnik metodologije (angl. scrum master) in razvojna ekipa, tri aktivnosti, ki so načrtovanje sprinta (angl. sprint planning), revizija sprinta (angl. sprint review) in dnevni scrum sestanek, in tri artefakte, ki so seznam zahtev izdelka (angl. product backlog), seznam zahtev sprinta (angl. sprint backlog) in graf preostalega časa (angl. burndown chart). Uvajanje Scruma pomeni vpeljavo časovno opredeljenih iteracij in razdelitev dela v manjše cilje razvrščene po prioriteti. Spremembe zahtev med iteracijo niso sprejete, drugače pa so dobrodošle. Metodologija predpisuje dnevne Scrum sestanke, ki pravilno trajajo 15 minut in mesečne sprinte oziroma iteracije. Na koncu vsake iteracije ima ekipa retrospektiven sestanek, kjer išče tudi načine za izboljšavo naslednjih iteracij [20].

Scrum je najbolj razširjena med agilnimi metodologijami [13]. Leta 2004 sta Kapitsaki in Christou v študiji [39], ki je vključevala več kot 126 podjetij, ugotovila, da med organizacijami, ki uporabljajo agilne metode kar 76.9% organizacij uporablja Scrum. Rezultati študije o najbolj uporabljenih agilnih metodah so predstavljeni na Sliki 2.3.

Metodologija je zasnovana tako, da je celoten proces prilagodljiv spremembam [63]. Zagotavlja nadzorne mehanizme za načrtovanje izdaje produkta in upravljanje s spremenljivkami tekom procesa razvoja. To organizacijam omogoča, da lahko kadarkoli spremenijo projekt in rezultate, s čimer zagotovijo objavo najprimernejše rešitve. Zahteve so lahko zelo dobro prioritizirane. Negativna plat Scrum metodologije je pomanjkanje dokumentacije [46]. Poleg tega je potrebno dobro ekipno sodelovanje in predanost članov,



Slika 2.3: Najbolj uporabljene agilne metodologije [39].

sicer obstaja precejšnje tveganje, da bo projekt neuspešen.

2.1.4 Kanban

Koncept Kanbana je kot del proizvodnega sistema Toyota (angl. Toyota production system) razvil Taiichi Ohno za uporabo v avtomobilski industriji [50]. Metodologijo je razvil zaradi težkih ekonomskih pogojev, v katerih so se v podjetju znašli po drugi svetovni vojni.

Kanban se je prvič začel uporabljati kot metodologija pri razvoju programske opreme leta 2004, ko jo je David J. Anderson vpeljal v majhno ekipo pri podjetju Microsoft [4].

Kanban je japonski izraz, kjer "kan" pomeni vizualni in "ban" kartica [20]. Uporaba Kanbana pomeni razdeljevanje dela v kose, pisanje njihovih opisov na kartice, ki se jih prilepi na Kanban tablo, tako da delovni proces vidi celotna ekipa. Pomembno je, da so omejitve obsega dela izrecno predstavljene na tabli. Kanban tabla zagotavlja transparenten prikaz razvojnega procesa, ker prikazuje dodeljevanje dela razvijalcem, sporoča prioritete in poudarja ozka grla. Eden ključnih ciljev Lean-Kanban pristopa je zmanjševanje obsega dela. Proces je optimiziran in čas dokončanja projekta se bolj sklada

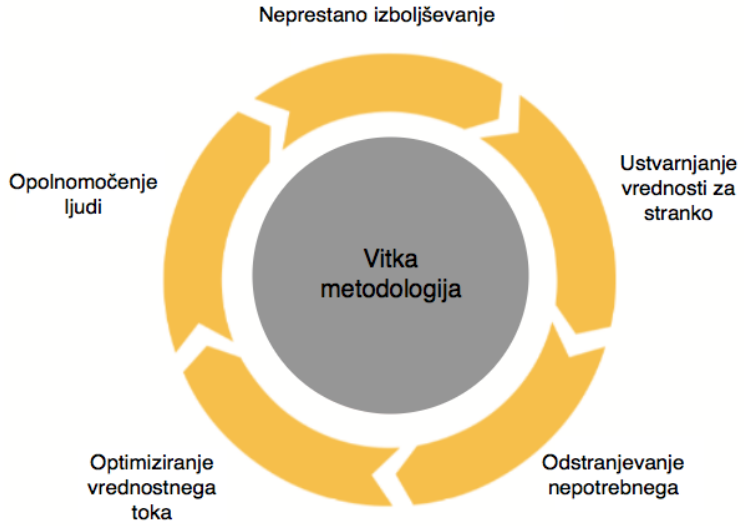
s predpisanim rokom. Metodologija omogoča boljše razumevanje celotnega razvojnega procesa [66]. Poleg tega se izboljša kvaliteta razvite programske opreme in zadovoljstvo strank [3]. Kanban ni metodologija, ki bi jo lahko samostojno uporabljali, poleg Kanban tehnik je potrebno vpeljati tudi druge prakse [37]. Izziv pri vpeljevanju Kanban metodologije je težavnost vpeljevanja sprememb v kulturi in filozofiji organizacije. Poleg tega so potrebna specifična znanja in izobraževanja zaposlenih. Glavni razlog za vpeljavo Kanbana so preprostost, osredotočenost na tok dela in neobveznost iteracij [2].

2.1.5 Vitek razvoj (angl. Lean)

Osnovni koncepti vitkega razvoja (angl. Lean) izvirajo iz leta 1913, ko sta Taylor in Ford postavila proizvodnjo v Highland Parku, kjer so z vpeljavo določenih tehnik in orodij uspeli hitrost in kvaliteto avtomobilske proizvodnje močno izboljšati [18]. Vendar pa so imele njihove metode tudi pomanjkljivosti. Kasneje so Japonci izboljšano verzijo metodologije vpeljali v podjetju Toyota za proizvodno linijo njihove avtomobilske industrije. Vitek razvoj je bil prvič predstavljen v Krafcikovi študiji leta 1988 [42]. Metodologija se je šele v tem stoletju začela uporabljati tudi za razvoj programske opreme. Prvo študijo o vitkem načinu razvoja programske opreme sta objavila Mary in Tom Poppendieck leta 2003 [55]. Metodologija ima tri ključne elemente: vitki koncepti, vitka načela in vitke prakse [74]. Vitki koncepti so, da vrednost določa stranka in da je razumevanje vrednosti, ki jo zaznava stranka, bistvenega pomena. Tok vrednosti določa vsak korak procesa in kategorizira korake glede na njihovo dodano vrednost. Pomembno je, da je razvojni proces ves čas v teku. Nič se ne razvije, dokler ni potrebno. Značilno je stremljenje k popolnosti v procesu z neprestanim prepoznavanjem in odstranjevanjem nepotrebnega. Vodilno načelo vitke metodologije je odstranjevanje stvari, ki so za stanko nepotrebne iz procesa razvoja [55].

Na področju razvoja programske opreme so nepotrebne stvari lahko:

- dodatne funkcionalnosti,



Slika 2.4: Cikel vitkega razvoja [29].

- čakanje,
- menjava nalog,
- dodatni procesi,
- delno opravljeno delo,
- napake,
- neuporabljena kreativnost zaposlenih.

Primeri praks vitkega razvoja so analiza vrednosti toka procesa (angl. Value stream mapping) z vidika uporabnika procesa in kupca in analiza izvora problema z orodjem petih "zakaj?" [65]. Cikel vitkega razvoja je predstavljen na Sliki 2.4 .

Uspešna implementacija vitke filozofije in načel je dolg in drag proces, kjer je potrebno vpeljati tako koncepte kot tudi tehnike [19]. Eden glavnih razlogov za neuspešno vpeljavo je pomanjkanje zaupanja, predanosti in sprejemanja sprememb pri zaposlenih. Težave, do katerih lahko pride pri vitki

metodologiji, so potencialno nezadovoljstvo strank, draga implementacija, stroški produktivnosti in nesprejetje metodologije s strani zaposlenih. Po drugi strani pa so se koncepti, prakse, načela in miselnost vitkosti izkazali za odlično metodologijo izboljšanja produktivnosti in kvalitete izdelkov [32].

2.1.6 Ekstremno programiranje

Metodologijo ekstremnega programiranja (angl. extreme programming) je razvil Beck in jo opisal v knjigi, objavljeni leta 2000 [11]. Ekstremno programiranje oz. XP pospeši hitrost razvoja in ekipi omogoča fleksibilnost pri spremembah zahtev [45]. Gre za metodologijo, ki je najbolj primerna za manjše ekipe s petimi do petnajstimi razvojniki. Osredotoča se na najboljše razvojne prakse in je sestavljena iz dvanajstih praks [10]:

1. **Načrtovalske igre:** Stranke določijo obseg in čas objav produkta na podlagi ocen razvojne ekipe. Programerji implementirajo zgolj funkcionalnosti, ki jih zahtevajo zgodbe določene iteracije.

2. **Majhne izdaje:** Sistem gre v produkcijo nekaj mesecev pred rešitvijo celotnega problema. Nove objave so narejene zelo pogosto, vse od dnevnih do mesečnih objav.

3. **Metafore:** Ideja je definirana z metaforo ali nizom metafor, ki so razumljive tako strankam kot tudi razvojnikom.

4. **Enostaven načrt:** Ves čas se poganjajo testi, koda vsebuje čim manj razredov in metod ter se ne podvaja.

5. **Testiranje:** Razvojniki napišejo teste preden začnejo z razvojem, testi so zbrani in morajo vsi pravilno delovati. Stranke napišejo funkcionalne teste za zgodbe v iteraciji. Tudi ti testi morajo delovati, vendar je včasih potrebna poslovna odločitev, ki primerja stroške odpravljanja napake in stroške zamude.

6. **Refaktoriranje:** Zasnova sistema se razvija s preoblikovanjem obstoječe strukture, ki obdrži vse teste v teku.

7. **Programiranje v paru:** Celotno razvojno kodo napišeta dve osebi z enim monitorjem, tipkovnico in miško.

8. **Kolektivne lastnine:** Vsak razvojnik lahko kadarkoli izboljša kodo kjerkoli v sistemu, če vidi priložnost za to.

9. **Stalne integracije:** Nova koda je vključena v sedanji sistem po manj kot nekaj urah. Pri integraciji je sistem na novo zgrajen in vsi testi morajo biti opravljeni, drugače so vse nove spremembe zavržene.

10. **40-urni tedni:** Nihče ne more delati dva zaporedna tedna nadur. Če se prepogosto uporablja nadure, je to znak globljih težav, ki jih je potrebno obravnavati.

11. **Prisotnost strank:** Stranka je celoten delovni čas zraven ekipe.

12. **Pravila:** Člani ekipe, ki uporabljajo XP metodologijo, se strinjajo, da se bodo držali pravil. Pravila je možno kadarkoli spremeniti, če se ekipa strinja s posledicami sprememb.

Osnovna struktura oziroma razvojni cikel metodologije omogoča prilagajanje tudi majhnim ekipam [58]. Izjema je, če je v ekipi samo en programer, kjer je potrebno nekatere prakse, kot je programiranje v paru, izpustiti. XP je dobra izbira za projekte, kjer pogosto pride do sprememb in kjer so zahteve uporabnikov slabo definirane [40]. Slabost metodologije je, da testiranje in programiranje izvaja ista oseba, kar predstavlja tveganje, da bodo nekatere napake ostale neodkrite, saj na produkt gleda iz iste perspektive.

2.1.7 Crystal

Družino metod Crystal je zasnoval Alistair Cockburn [22] v 90. letih prejšnjega stoletja. Metode je razvil na podlagi večletnega opazovanja razvojnih ekip.

Ugotovil je, da ekipe ne uporabljajo formalnih metod in kljub temu delajo uspešne projekte. Dokumentiral je način njihovega dela in iz tega razvil Crystal metode. Osnovane so na naslednjih načelih [33]:

- Vsak projekt potrebuje nekoliko drugačen nabor pravil, dogovorov ali drugačno metodologijo.
- Na delovanje projekta vplivajo težave ljudi, ki ga razvijajo. Če se te težave razrešijo, se posamezniki izboljšajo in tudi timsko delo postane bolj učinkovito.
- Boljša komunikacija in pogoste objave izboljšajo kvaliteto produkta.

Metodologije so namenjene različnim velikostim delovnih ekip in kritičnosti projekta [12]:

- Clear: metoda namenjena ekipam do osem članov,
- Yellow: za ekipe z deset do dvajset člani,
- Orange: za ekipe z dvajset do petdeset člani,
- Red: za petdeset do sto-članske ekipe,
- Metode poimenovane po temnejših barvah: namenjene ekipam z več kot sto člani.

Najbolj agilna metoda, to je Crystal Clear, se osredotoča na komunikacijo v majhnih ekipah, ki razvijajo programsko opremo. Na drugi strani pa je Crystal Violet namenjena življenjsko pomembnim projektom večjih ekip.

Razvoj po metodologiji Crystal ima sedem značilnosti:

- pogosta dostava,
- konstantne povratne informacije,
- neprestana komunikacija,

- osebna varnost,
- osredotočenost,
- enostaven dostop do uporabnikov,
- avtomatizirano testiranje in pogoste postavitve.

Metode Crystal omogočajo souporabo poljubnih razvojnih praks ter orodij, združimo jih lahko na primer z XP ali Scrum praksami [1]. Odvisno od kritičnosti in velikosti projekta Crystal pri nekaterih metodah zapoveduje določene prakse, vendar ne podaja navodil za njihovo izvajanje.

2.1.8 Feature-driven development (FDD)

Metodologijo "Feature-driven development" oziroma FDD sta Jeff De Luca in Peter Code razvila leta 1997-98 za projekt razvoja programske opreme v singapurski banki, na katerem je 15 mesecev delalo 50 ljudi [1]. FDD je metodologija iterativnega in inkrementalnega razvoja programske opreme, ki združuje modelski in agilni razvoj s poudarkom na začetnem objektnem modelu, delitvi dela na funkcije (angl. feature) in iterativnem oblikovanju za vsako funkcijo [53]. Metodologija naj bi bila primerna za razvoj kritičnih sistemov. FDD predstavlja iterativni način razvoja, ki vključuje prakse, ki naj bi bile uspešne v industriji [1]. FDD proces je edinstven za vsak primer posebej. V celotnem procesu se poudarja vidik kvalitete in vključuje pogoste in oprijemljive objave, hkrati pa se natančno spremlja napredek projekta. Posamezna iteracija je sestavljena iz dveh faz: oblikovanja in razvoja [72]. Na začetku se pridobi seznam zahtev strank, ki se jih nato razdeli na posamezne funkcionalnosti. Nato se definira postopek iterativnega in inkrementalnega procesa tako, da postopoma razvijajo funkcionalnost in rezultate sproti dostavljajo stranki. Po vsaki iteraciji in glede na povratne informacije stranke preglejujejo seznam funkcionalnosti. FDD predlaga tedenski 30-minutni sestanek, kjer obravnavajo status vseh funkcionalnosti [52]. Vodi se zapisnik sestanka, s katerim se spremlja napredek zahtev. Pomemben del

metodologije FDD je pregled kode [1]. Po vsaki iteraciji se koda deli s celotno ekipo, sledi sestanek, kjer se pogovori o težavah oziroma nepravilnostih, ki jih je ekipa našla v kodi. S pomočjo te prakse se celotna ekipa seznani z delom ostalih članov, ekipa se bolj drži kodirnih standardov, in najde se napake, ki jih modularno testiranje (angl. unit testing) ne odkrije. Pri metodologiji FDD ima vodja projekta končno besedo pri obsegu projekta, urniku in članih kolektiva [1]. Metodologija je primerna za ekipe, kjer imajo člani različen nivo izkušenj. Zaradi hierarhije v razvojni ekipi, ki omogoča manjše ekipe za posamezno iteracijo v okviru velike projektne skupine, je dobra za obširne projekte. Metodologija je bolj primerna za projekte, kjer se zahteve uporabnikov ne spreminjajo veliko.

2.2 Organizacijske karakteristike

Izzivi, s katerimi se raziskovalci soočajo pri izboljševanju uspešnosti razvoja programske opreme, so predvsem organizacijski [59]. V vsaki organizaciji je mogoče identificirati organizacijske karakteristike, ki so pri njih vzpostavljene. Za ugotavljanje stanja organizacijskih karakteristik v posamezni organizaciji moramo definirati ključne organizacijske karakteristike. Leta 1995 so Silver, Markus in Beath postavili model [67], kjer so definirali ključne dimenzije organizacijskih karakteristik. V magistrskem delu smo se osredotočili na dimenzije, ki so v literaturi najbolj uveljavljene, in sicer na strategijo podjetja, njegovo strukturo in organizacijsko kulturo. Pri definiranju posameznih karakteristik smo se oprli na model, ki ga je Hovelja [35] postavil v svojem doktorskem delu. Model je prvi poskusil vključiti vse obstoječe organizacijske karakteristike, za katere je v literaturi dokazano, da pomembno vplivajo na uspešnost uporabe informacijskih tehnologij v podjetjih.

2.2.1 Strategije

Prva organizacijska dimenzija je dimenzija strategij. V literaturi lahko zasledimo štiri glavne strateške karakteristike, ki vplivajo na uspešnost uporabe

informacijskih tehnologij, ki sta jih identificirala Bakos in Treacy v študiji leta 1986 [7]. Te karakteristike se osredotočajo na stopnjo oblikovanja strategij. Poleg teh pa smo v model vključili tudi karakteristiko strateškega načrtovanja, ki bolj celostno zajame vse stopnje strategij.

1. **Strateško načrtovanje:** Karakteristika strateškega načrtovanja predstavlja analizo okolja (tekmecev, kupcev, ...), oblikovanje strategije (pomen stroškov, povečevanja raznolikosti proizvodov in storitev, ...), uresničevanje strategije (odgovorne osebe, časovni roki, sankcije, . . .) ter oceno in nadzor uresničenega. Strategija načrtovanja pomeni, da so cilji podjetja jasni, realno dosegljivi, časovno definirani, pokrivajo ključna področja in so vezani na nagrajevanje [27].

2. **Stroškovna strategija:** Karakteristika stroškovne strategije predstavlja izboljšanje izvedbene učinkovitosti in funkcijske uspešnosti. Strategija opredeljuje cilje podjetja povezane s stroški razvoja (plače, oprema, licence, strežniki, ...).

3. **Strategija diferenciacije:** Karakteristika strategije diferenciacije opredeljuje kakovost razvoja. Strategija predstavlja izboljšanje kakovosti poslovnih procesov, proizvodov in/ali storitev s pomočjo inovacij.

4. **Strategija obrambe tržnega položaja:** Karakteristika strategije obrambe tržnega položaja pomeni izboljšanje pogajalskega položaja v razmerjih do kupcev in dobaviteljev ter oblikovanje vstopnih ovir.

5. **Strategija sodelovanja:** Karakteristika strategije sodelovanja oziroma povezovanja predstavlja izboljšanje sinergij s poslovnimi partnerji. Strategija pomeni sodelovanje s poslovnimi partnerji in strankami [7].

2.2.2 Strukture

Druga dimenzija organizacijskih karakteristik je dimenzija struktur. Ob pregledu literature [14, 30, 48, 9, 41] smo identificirali osem ključnih strukturnih

karakteristik, ki vplivajo na uspešnost uporabe IT.

1. **Specializacija:** Karakteristika specializacije predstavlja raznolikost delovnih nalog za katere je zaposleni odgovoren. Manjši razpon različnih delovnih nalog pomeni večji nivo specializacije.

2. **Centralizacija:** Karakteristika centralizacije oziroma decentralizacije predstavlja stopnjo svobode odločanja posameznega zaposlenega pri opravljanju delovnih nalog. V večji meri kot vodstvo podjetja odločanje pri opravljanju delovnih nalog prepušča izvajalcem teh nalog, manj je podjetje centralizirano.

3. **Hierarhija:** Karakteristika hierarhije pomeni obseg kontrole nad zaposlenimi, ki kaže potrebo po večjem številu ravni v hierarhiji, torej navpično diferenciacijo. Večji obseg kontrole pomeni večjo hierarhičnost.

4. **Formalizacija:** Karakteristika formalizacije pomeni predpisanost hitrosti in metod oziroma postopkov dela. Večja določenost pravil, ki se jih mora posameznik držati na delovnem mestu, pomeni višjo stopnjo formalizacije.

5. **Struktura opravljanja delovnih nalog:** Karakteristika strukture opravljanja delovnih nalog se nanaša na način opravljanja delovnih nalog, in sicer ali se jih opravlja individualno ali ekipno.

6. **Profesionalizacija:** Karakteristika profesionalizacije oziroma strokovnosti pomeni povprečno raven izobrazbe in obsega strokovnih znanj zaposlenih. Profesionalizacija se odraža s kadrovskimi standardi podjetja kot na primer potrebna znanja, spretnosti, ujemanja kandidata s kulturo podjetja.

7. **Sistem nagrajevanja:** Karakteristika sistema nagrajevanja se nanaša na nagrajevanje zaposlenih (denarne nagrade, napredovanje), ki je ve-

zano na uspešno delo, hitrost osvajanja novih znanj, vodenje ali način komuniciranja zaposlenih.

8. **Struktura povezav podjetja z okoljem:** Karakteristika strukture povezav podjetja z okoljem predstavlja kakovost in obseg povezav med podjetjem in njegovim okoljem. Struktura povezav pomeni dostopnost potrebnih informacije o novih tehnologijah in dobrih praksah. Povezovanje lahko pomeni sodelovanje v strokovnih, panožnih in tehnoloških združenjih, dostop do strokovnih revij, seminarjev in kongresov.

2.2.3 Kultura

Zadnja organizacijska dimenzija je kultura organizacije. V literaturi lahko zasledimo veliko študij, ki pravijo, da je organizacijska kultura bistvenega pomena za uspešnost organizacije. Pomemben izziv vodstva podjetja je določiti, kakšne karakteristike kulture so najbolj učinkovite za njihovo organizacijo in ali obstaja potreba po spremembi organizacijske kulture [6].

Delobbe, Haccoun in Vandenberghe so leta 2002 objavili raziskavo, v kateri so analizirali vprašalnike o organizacijski kulturi in z empirično študijo identificirali pet ključnih karakteristik organizacijske kulture [24].

1. **Podpora:** Karakteristika podpore združuje priznavanje za delo, stabilnost in načrtovanost delovnega okolja. Vprašalniki, ki preverjajo to karakteristiko, vsebujejo trditev kot na primer »Nadrejeni poskušajo rešiti probleme zaposlenih.«.

2. **Solidarnost:** Karakteristika solidarnosti združuje pripadnost in vključenost v organizacijo kot tudi sodelovanje in solidarnost sodelavcev. Vprašalniki, ki preverjajo to karakteristiko, vsebujejo trditev kot na primer »Zaposleni po navadi radi delajo drug z drugim.«.

3. **Inovativnost:** Karakteristika inovativnosti združuje uspešnost, produktivnost in inovativnost, odprtost do sprejemanja novosti ter pripravljenost tveganja. Vprašalniki, ki preverjajo to karakteristiko, vsebu-

jejo trditev kot na primer »V tem podjetju se neprestano trudimo, da bi razvili nove poslovne učinke.«.

4. **Kontrola:** Karakteristika kontrole pove, kako pravila, formalni postopki in hierarhična moč vplivajo na delovanje v združbi. Združuje priznavanje hierarhije, spoštovanje, oblikovanje jasnih pravil in formalizacijo dela. Vprašalniki, ki preverjajo to karakteristiko, vsebujejo trditev kot na primer »Kar moram početi je natančno določeno s formalnimi postopki.«.

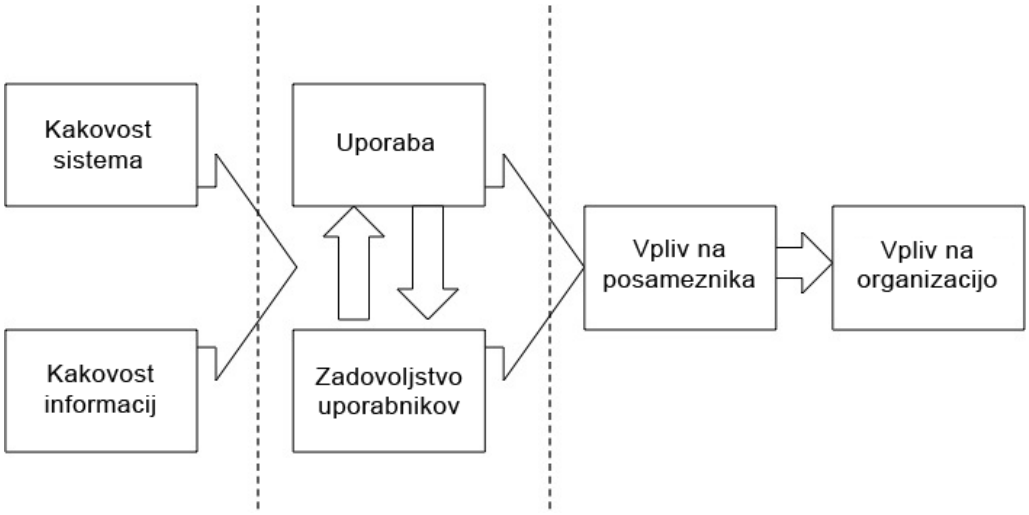
5. **Neprestano učenje:** Karakteristika neprestanega učenja združuje usposabljanja in pristojnost za delo. Vprašalniki, ki preverjajo to karakteristiko, vsebujejo trditev kot na primer »Stalno izboljšujem svojo usposobljenost z branjem strokovnih revij/člankov s svojega področja.«.

2.3 Merjenje uspešnosti

Da bomo lahko postavili model za ocenjevanje uspešnosti razvoja programske opreme, je potrebno pregledati najbolj uveljavljene modele merjenja uspešnosti. V literaturi lahko zasledimo najbolj pogosto uporabljen model merjenja uspešnosti informacijskih sistemov, ki sta ga postavila DeLone in McLean leta 1992 [25]. Deset let po objavi pa sta model še dopolnila in objavila izboljšano verzijo [26]. Dodatne kriterije ocenjevanja uspešnosti projektov razvoja programske opreme zasledimo tudi v Atkinsonovem modelu [5].

2.3.1 Model merjenja uspešnosti DeLone in McLean

Model [25] zajema šest dimenzij uspešnosti, in sicer kakovost sistema, kakovost informacij, uporabnost, zadovoljstvo uporabnikov, vpliv na posameznika in vpliv na organizacijo. Dimenzije so predstavljene na Sliki 2.5. Gre



Slika 2.5: DeLonov in McLeanov model merjenja uspešnosti iz leta 1992 [25].

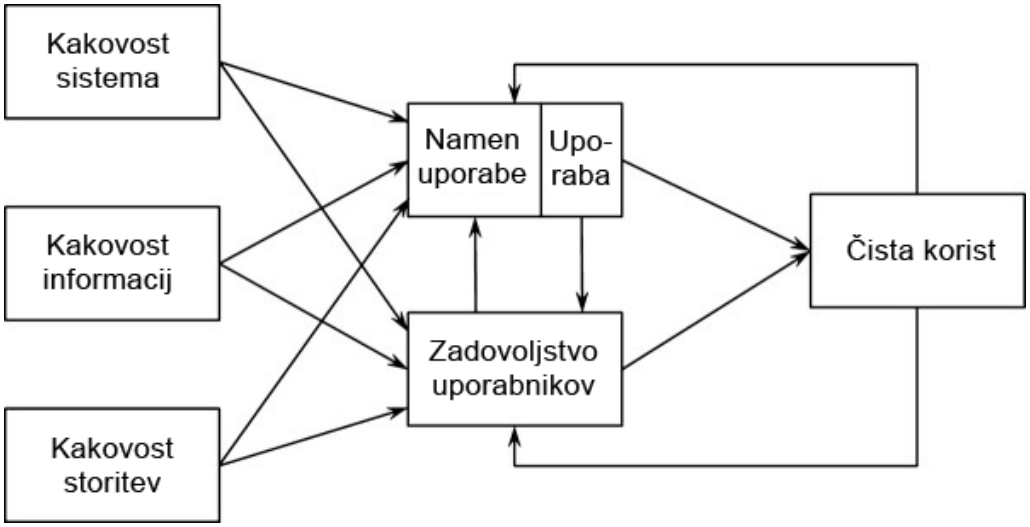
za enega najbolj razširjenih in uporabljenih modelov merjenja uspešnosti na področju informacijskih sistemov.

Leta 2003 sta DeLone in McLean [26] objavila prenovljeno verzijo modela, ki je model še izboljšala in izpopolnila. Nova verzija modela definira naslednje dimenzije uspešnosti:

1. **Kakovost sistema:** Sistem ima želene značilnosti, kot na primer enostavnost uporabe, fleksibilnost, zanesljivost, enostavno učenje, intuitivnost, prilagodljivost in odzivnost.

2. **Kakovost informacij:** Sistem ima želene značilnosti sistemskih izhodov (poročila o upravljanju in spletne strani) kot na primer ustreznost, razumljivost, točnost, jedrnatost, popolnost, pravočasnost in uporabnost.

3. **Kakovost storitev:** Dimenzija predstavlja kakovost podpore, ki jo uporabniki prejemajo od IS oddelka, oziroma drugega osebja v IT podpori. Na primer: odzivnost, natančnost, zanesljivost, tehnična usposobljenost in empatija.



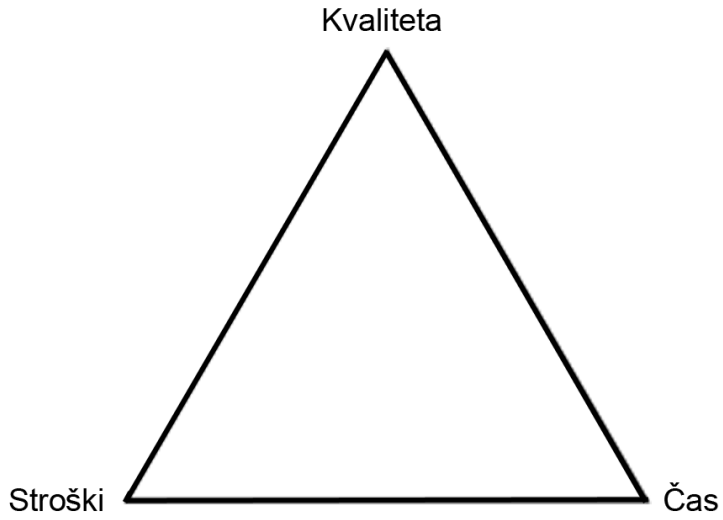
Slika 2.6: Prenovljen DeLonov in McLeanov model merjenja uspešnosti [26].

4. **Namen uporabe oziroma uporaba:** Dimenzija predstavlja stopnjo oziroma način, na katerega zaposleni in kupci izkoriščajo zmogljivosti informacijskega sistema. Na primer: obseg uporabe, pogostost uporabe, vrsta uporabe, ustreznost uporabe, obseg in namen uporabe.

5. **Zadovoljstvo uporabnikov:** Dimenzija predstavlja raven zadovoljstva uporabnikov, ki se jo da meriti preko podpore uporabnikov, spletnih obrazcev in poročil.

6. **Čista korist** Dimenzija predstavlja obseg koristi za posameznike, skupine, organizacije, industrije in narode. Na primer za izboljšanje odločanja, večjo produktivnost, večjo prodajo, zmanjšanje stroškov, izboljšanih dobičkov, tržno učinkovitost, blaginjo potrošnikov, ustvarjanje novih delovnih mest in gospodarski razvoj [54].

Dimenzije so med seboj povezane kot prikazuje Slika 2.6.



Slika 2.7: Železni trikotnik [5].

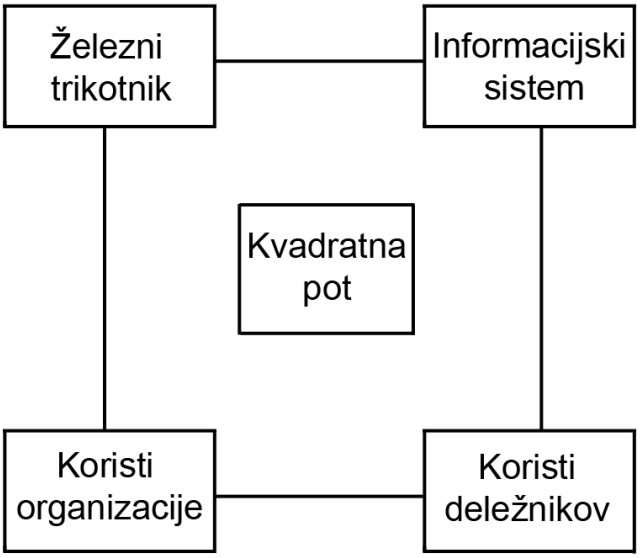
2.3.2 Atkinsonov model merjenja uspešnosti

Najbolj pogosti kriteriji za ocenjevanje uspešnosti projekta so bili dolgo časa kvaliteta, čas in stroški oziroma tako imenovan železni trikotnik [5] (angl. The Iron Triangle), ki je predstavljen na Sliki 2.7.

V študiji [5], objavljeni leta 1999, je Roger Atkinson izpostavil pomanjkljivosti železnega trikotnika in predlagal dodatne kriterije merjenja uspešnosti, ki bistveno izboljšajo merjenje uspešnosti. Razvil je tako imenovan model kvadratne poti (angl. The Square Route), ki je sestavljen iz štirih dimenzij uspeha, predstavljenih na Sliki 2.8.

1. **Železni trikotnik:** Kriterij, ki je sestavljen iz kvalitete, časa in stroškov. Gre za v splošnem najbolj uporabljene kriterije za ocenjevanje uspešnosti.

2. **Koristi organizacije:** Kriterij predstavlja tako direktne kot posredne



Slika 2.8: Kvadratna pot [5].

koristi, ki jih ima organizacija od projekta, ki ga ocenjujemo. Uspešno izvedeni IT/IS projekti vodijo do direktnih ali posrednih finančnih koristi za organizacijo [23].

3. **Koristi deležnikov:** Kriterij, ki predstavlja koristi projekta za uporabnike in ostale deležnike. Vodja projekta lahko projekt oceni kot uspešen, ker so bile vse zahteve izpolnjene, medtem ko uporabnike in drugi deležnike bolj zanima učinkovitost in uporabniška izkušnja.

4. **Informacijski sistem:** Kriterij predstavlja povezavo med tehnologijo in uspehom projekta. Nivo tehničnega znanja in integracije tehnologije v proces ima velik vpliv na uspešnost projekta [49].

Poglavje 3

Razvoj modela

Na podlagi pregleda literature, kjer smo pregledali najbolj uveljavljene metodologije razvoja programske opreme, definirali organizacijske karakteristike in preučili najbolj razširjene modele za ocenjevanje uspešnosti, smo postavili celovit model ocenjevanja uspešnosti razvoja programske opreme, ki upošteva tudi organizacijske karakteristike organizacije. Model je osnovan na pristopu, ki sta ga leta 2012 razvila Vavpotič in Hovelja [34], ter pristopu, ki pri ocenjevanju vključuje tudi sociološke vidike in sta ga leta 2009 objavila Vavpotič in Bajec [73].

3.1 Faze razvoja programske opreme

Razviti model proces razvoja programske opreme se deli na razvojne faze, ki so povzete po metodologiji RUP (angl. Rational Unified Process). Model opredeljuje pet ključnih tehničnih razvojnih faz, pri čemer smo se omejili na sledeče [71]:

1. **Zajem zahtev** Faza je sestavljena iz aktivnosti, ki privedejo do zahtev, ki jih je potrebno uresničiti. Te aktivnosti so na primer pogovor s strankami, prototipiranje, testiranje in dokumentacija zahtevanih funkcionalnosti.

2. **Načrtovanje arhitekture** Na podlagi zajetih zahtev se v fazi načrtovanja arhitekture izdela načrt, kako jih v fazi implementacije realizirati.

3. **Implementacija** Faza sledi izdelanemu načrtu in se drži zajetih zahtev.

4. **Testiranje** Faza testiranja preverja pravilno implementacijo vseh zahtev in identificira morebitne napake implementirane programske opreme.

5. **Postavitev (angl. deployment)** Zaključna faza procesa razvoja programske opreme predstavlja objavo produkta in izročitev končnim uporabnikom.

3.2 Ocenjevanje uspešnosti

Prvi korak modela ocenjevanja uspešnosti razvoja programske opreme z upoštevanjem vpliva organizacijskih karakteristik predstavlja ocenjevanje uspešnosti posameznih razvojnih faz trenutnega procesa razvoja programske opreme. Za ocenjevanje uspešnost posameznih faz model uporablja splošno najbolj uporabljene kriterije za ocenjevanje uspešnosti. To so kvaliteta, čas in stroški trenutnega načina dela. Trojico kriterijev poznamo tudi pod imenom železni trikotnik [5]. Model predvideva, da ta del ocenjevanja izvedemo z vodstvom podjetja s pomočjo trditev, ki so predstavljene v Tabeli 3.1.

Za celostno ocenjevanje model predvidi ocenjevanje tudi s strani zaposlenih v posameznih razvojnih fazah. Način dela v njihovi fazi ocenijo po zelo uveljavljenem kriteriju zadovoljstva uporabnikov, ki ga v svojem modelu uporabljata DeLone in McLean [26]. Kriterij zadovoljstva uporabnikov ocenjujemo s pomočjo trditve predstavljene v Tabeli 3.2.

Za vrednotenje trditev model pri vseh vprašalnikih uporablja sedemstopenjsko Likertovo lestvico [16], predstavljeno v Tabeli 3.3, ki meri strinjanje anketiranca s posamezno trditvijo. Lestvica se stopnjuje od števila ena, ki predstavlja najnižjo stopnjo strinjanja, do števila sedem, ki predstavlja najvišjo stopnjo strinjanja.

Tabela 3.1: Trditve za ocenjevanja načina dela s strani vodstva podjetja.

Kriterij ocenjevanja	Trditev
kvaliteta	Način dela pri <faza razvoja>močno poveča kakovost produkta/ov.
stroški	Način dela pri <faza razvoja>močno zniža stroške.
čas	Način dela pri <faza razvoja>močno poveča hitrost razvoja.

Tabela 3.2: Trditev za ocenjevanja načina dela s strani zaposlenih.

Kriterij ocenjevanja	Trditev
zadovoljstvo	Z načinom dela pri <fazi razvoja>sem zelo zadovoljen.

Tabela 3.3: Sedemstopenjska Likertova lestvica.

1	Sploh se ne strinjam
2	Se ne strinjam
3	Delno se ne strinjam
4	Nevtralno
5	Delno se strinjam
6	Se strinjam
7	Popolnoma se strinjam

3.3 Organizacijske karakteristike

Posebnost modela je upoštevanje organizacijskih karakteristik podjetja. Pri pregledu literature smo zasledili pomembnost organizacijskih karakteristik za izboljševanje uspešnosti razvoja programske opreme. Pri postavljanju modela smo se osredotočili na najbolj uveljavljene dimenzije organizacijskih karakteristik, ki pomembno vplivajo na uspešnost uporabe informacijskih tehnologij v podjetjih. Te dimenzije so sicer kultura, strukture in strategije organizacije. Vsaka izmed dimenzij zajema vrsto karakteristik, ki so naštete v nadaljevanju.

3.3.1 Kultura

Prva dimenzija organizacijskih karakteristik našega modela je kultura organizacije. Gre za zelo pomembno dimenzijo, ki veliko razkrije o delovanju organizacije. V modelu smo uporabili pet ključnih karakteristik kulturne dimenzije, ki so jih Delobbe, Haccoun in Vandenberghe identificirali kot tiste, ki pomembno vplivajo na uspešnost uporabe IT v podjetju [24]. V raziskavi leta 2002 so analizirali vprašalnike o organizacijski kulturi in z empirično študijo identificirali podporo zaposlenih, solidarnost, inovativnost, kontrolo in neprestano učenje kot ključne karakteristike organizacijske kulture.

Stanje karakteristik preverjamo z vprašalnikom, namenjenem vodstvu podjetja in vodjem ekip, ki je predstavljen v Tabeli 3.4.

Tabela 3.4: Trditve kulturne dimenzije, namenjene vodstvu podjetja.

Organizacijska karakteristika	Trditev
Podpora zaposlenih	Nadrejeni poskušamo rešiti probleme podrejenih, izkazujejo priznavanje in podporo delu podrejenih, skrbijo za stabilnost in načrtovanost delovnega okolja.
Solidarnost	Zaposleni radi delamo drug z drugim, si pomagamo med seboj in smo solidarni.
Inovativnost	V podjetju se neprestano trudimo, da bi razvili nove poslovne učinke.
Kontrola	Vse, kar v podjetju počnemo, je natančno določeno s formalnimi postopki in pravili.
Neprestano učenje	Zaposleni stalno izboljšujemo svojo strokovnost z branjem strokovnih revij, obiskovanjem tečajev in kongresov z ustreznih strokovnih področij.

Poleg trenutnega stanja želimo z modelom preveriti tudi vpliv karakteristik kulture organizacije na delo posameznikov v vsaki razvojni fazi. Za ta namen uporabljamo vprašalnik, namenjen zaposlenim, ki je predstavljen v Tabeli 3.5.

Tabela 3.5: Trditve kulturne dimenzije, namenjene zaposlenim.

Organizacijska karakteristika	Trditev
Podpora zaposlenih	Podpora, ki jo prejmem od nadrejenih zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Solidarnost	Solidarnost in pomoč, ki mi jo sodelavci nudijo zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Inovativnost	Inovativnost pri razvoju novih poslovnih učinkov zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Kontrola	Količina formalnih postopkov in pravil zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Neprestano učenje	Mera stalnega izboljševanja strokovnosti z branjem strokovnih revij, obiskovanjem tečajev in kongresov zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.

3.3.2 Strukture

Druga dimenzija organizacijskih karakteristik modela je dimenzija struktur organizacije. Na podlagi pregleda literature smo identificirali osem strukturnih karakteristik, ki pomembno vplivajo na uspešnost uporabe IT. Te karakteristike so specializacija, struktura opravljanja delovnih nalog, formalizacija, hierarhija, centralizacija in profesionalizacija. Za pregled trenutnega stanja smo pripravili vprašalnik namenjen vodstvu podjetja in vodjem ekip, ki je predstavljen v Tabeli 3.6.

Tabela 3.6: Trditve strukturne dimenzije, namenjene vodstvu podjetja.

Organizacijska karakteristika	Trditev
Specializacija	Povprečni zaposleni v podjetju opravlja veliko različnih delovnih opravkov.
Struktura opravljanja delovnih nalog	Večino delovnih nalog v podjetju opravljamo v ekipah in skupinah in ne individualno.
Formalizacija	Hitrosti in metode dela so v našem podjetju natančno predpisane.
Hierarhija	Stopnja nadzora nad posameznikom je v podjetju majhna.
Centralizacija	Vodstvo podjetja odločanje pri opravljanju delovnih nalog v čim večji meri prepušča izvajalcem teh nalog in jih ne sprejema samo.
Profesionalizacija	Podjetje ima visoke kadrovske standarde, tako da lahko zaposleni s svojim obsegom strokovnih znanj vrhunsko opravljajo delovne naloge v podjetju.
Sistem nagrajevanja	Zaposleni so nagrajeni (denarne nagrade, napredovanje) za svoje uspešno delo v skupinah ter za hitrost pri osvajanju/učenju in uporabljanju novih tehnologij in tehnik organiziranja (vodenje, komuniciranje...).
Struktura povezav podjetja z okoljem	Podjetje ima široko mrežo povezav z okoljem, ki omogoča dostop do več virov informacij (panožna, tehnološka združenja, strokovne revije in seminarji ter informacije pridobljene od poslovnih partnerjev).

Poleg informacij o trenutnem stanju model predpostavlja tudi pridobiva-

nje informacij o vplivu strukturnih karakteristik na delo zaposlenih v vsaki razvojni fazi. Te informacije pridobimo z vprašalnikom, namenjenim zaposlenim, ki je sestavljen iz trditev v Tabeli 3.7.

Tabela 3.7: Trditve strukturne dimenzije, namenjene zaposlenim.

Organizacijska karakteristika	Trditev
Specializacija	Raznolikost mojih delovnih opravkov zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Struktura opravljanja delovnih nalog	Način opravljanja delovnih nalog (individualno, v skupini, ekipi) zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Formalizacija	Predpisanost metod in hitrosti dela zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Hierarhija	Stopnja nadzora zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Centralizacija	Svoboda odločanja pri opravljanju delovnih nalog zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Profesionalizacija	Obseg strokovnih znanj v naši razvojni skupini zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Sistem nagrajevanja	Mera nagrajevanja za uspešno delo, hitro učenje, vodenje,... zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.
Struktura povezav podjetja z okoljem	Dostopnost informacij (panožna, tehnološka združenja, strokovne revije in seminarji ter informacije pridobljene od poslovnih partnerjev) zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.

3.3.3 Strategije

Zadnja organizacijska dimenzija je dimenzija strategij organizacije. Na podlagi pregleda literature smo v modelu uporabili pet ključnih karakteristik strategij organizacije, ki najbolj vplivajo na uspešnost uporabe informacijskih tehnologij. Iz študije, ki sta jo leta 1986 opravila Bakos in Treacy [7], smo povzeli štiri strateške karakteristike, ki se osredotočajo na stopnjo oblikovanja strategij, in sicer stroškovno strategijo, strategijo diferenciacije, strategijo obrambe tržnega položaja in strategijo sodelovanja. Poleg omenjenih štirih karakteristik smo v model vključili tudi karakteristiko strateškega načrtovanja, ki bolj celostno zajame vse stopnje strategij. Za identificiranje trenutnega stanja strategij organizacij smo pripravili vprašalnik, namenjen vodstvu podjetja in vodjem ekip, ki je predstavljen v Tabeli 3.8.

Tabela 3.8: Trditve strateške dimenzije, namenjene vodstvu podjetja.

Organizacijska karakteristika	Trditev
Strateško načrtovanje	Analiza okolja (tekmecev, kupcev, itd.), oblikovanje strategije (pomen stroškov, povečevanja raznolikosti proizvodov in storitev, itd.), uresničevanje strategije (odgovorne osebe, časovni roki, sankcije, itd.) ter ocena in nadzor uresničenega je oziroma bo proces, ki pomembno prispeva k rasti dodane vrednosti podjetja.
Stroškovna strategija	Izboljšanje operativne učinkovitosti in funkcijske uspešnosti je oziroma bo za rast dodane vrednosti v podjetju ključnega pomena.
Strategija diferenciacije	Izboljšanje kakovosti poslovnih procesov, proizvodov in/ali storitev je oziroma bo za rast dodane vrednosti v podjetju ključnega pomena.
Strategija obrambe tržnega položaja	Izboljšanje pogajalskih položajev in oblikovanje vstopnih ovir je oziroma bo za rast dodane vrednosti v podjetju ključnega pomena.
Strategija sodelovanja	Izboljšanje skupnega delovanja s poslovnimi partnerji je oziroma bo za rast dodane vrednosti v podjetju ključnega pomena.

Za celosten vpogled model predpostavlja oceno vpliva strateških karakteristik na delo zaposlenih v vsaki posamezni razvojni fazi. Oceno postavljamo s pomočjo vprašalnika, namenjenega zaposlenim, ki je predstavljen v Tabeli 3.9.

Tabela 3.9: Trditve strateške dimenzije, namenjene zaposlenim.

Organizacijska karakteristika	Trditev
Strateško načrtovanje	Cilji podjetja (jasnost, realna dosegljivost, časovna definiranost, pokritost ključnih področij in vezanost ciljev na nagrajevanje) zelo pozitivno vplivajo na učinkovitost mojega dela pri <faza razvoja>.
Stroškovna strategija	Cilji podjetja, ki opredeljujejo stroške razvoja (plače, oprema, licence, strežniki,...) zelo pozitivno vplivajo na učinkovitost mojega dela pri <faza razvoja>.
Strategija diferenciacije	Cilji podjetja, ki opredeljujejo kakovost razvoja zelo pozitivno vplivajo na učinkovitost mojega dela pri <faza razvoja>.
Strategija obrambe tržnega položaja	Cilji podjetja, ki opredeljujejo želje strank pri razvoju zelo pozitivno vplivajo na učinkovitost mojega dela pri <faza razvoja>.
Strategija sodelovanja	Sodelovanje s poslovnimi partnerji in strankami zelo pozitivno vpliva na učinkovitost mojega dela pri <faza razvoja>.

3.4 Model ocenjevanja uspešnosti

Model ocenjevanja uspešnosti razvoja programske opreme z upoštevanjem vpliva organizacijskih karakteristik, ki smo ga razvili, je sestavljen iz petih faz.

1. **Popis procesa dela pri razvojnih fazah ter identifikacija organizacijskih karakteristik** V začetni fazi s tehničnimi vodjami popišemo trenutni proces dela pri fazah zajema zahtev, načrtovanja arhitekture,

implementacije, testiranja in postavitve. Poleg tega uskladimo uporabljene organizacijske karakteristike z vodstvom podjetja.

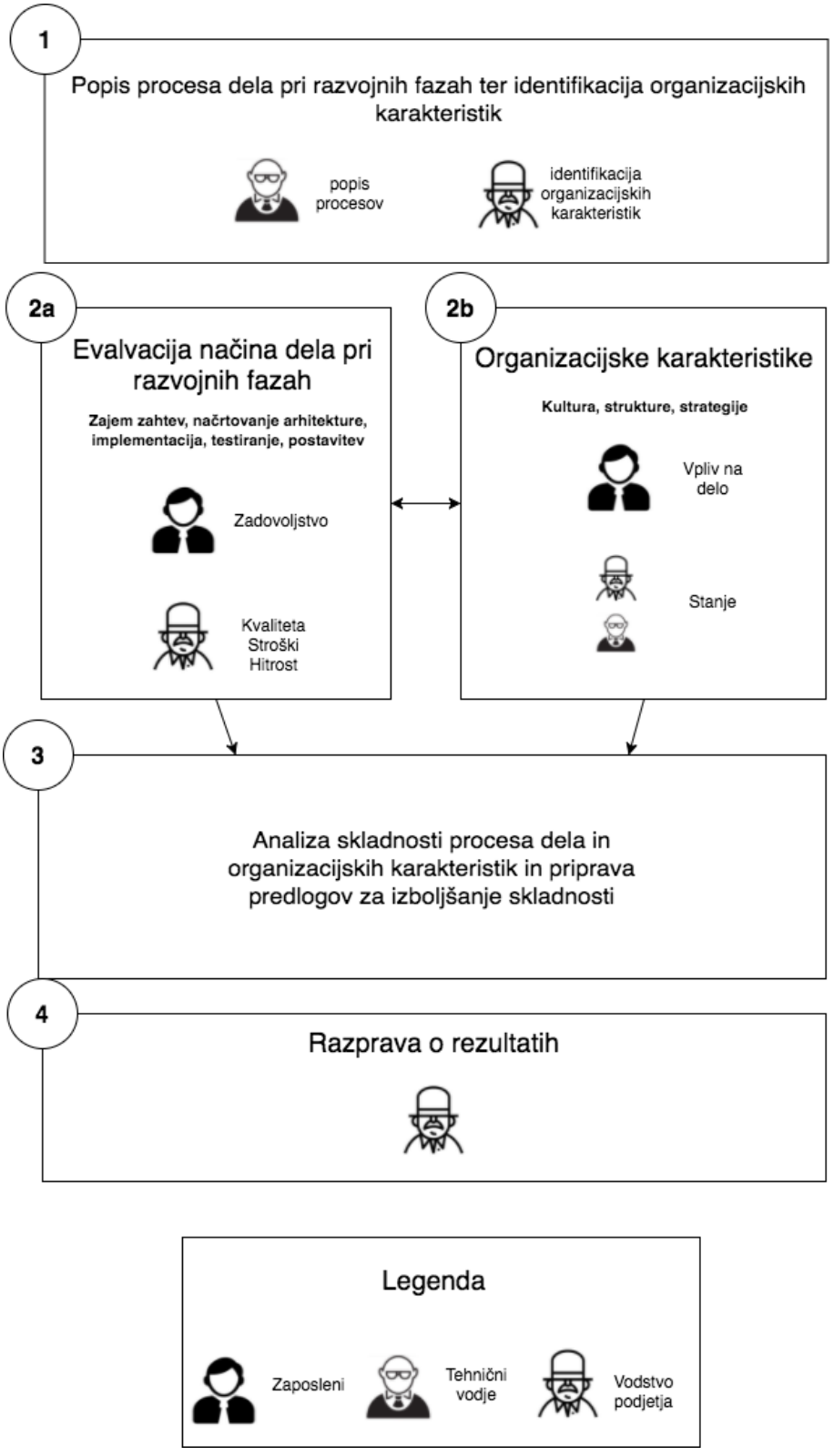
2. **Evalvacija načina dela za vsako razvojno fazo** Pri tej fazi ocenjujemo zadovoljstvo zaposlenih z načinom dela pri fazah razvoja. Poleg tega pri tej fazi s kriteriji kvalitete, stroškov in hitrosti z vodstvom podjetja ocenimo način dela pri vsaki razvojni fazi.

3. **Organizacijske karakteristike** Pri fazi z vodstvom podjetja in tehničnimi vodji ocenimo stanje organizacijskih karakteristik podjetja. Ocenimo posamezne karakteristike dimenzij kulture, struktur in strategij podjetja. Obenem naredimo evalvacijo vpliva organizacijskih karakteristik na delo zaposlenih v vsaki posamezni razvojni fazi.

4. **Analiza skladnosti procesa dela in organizacijskih karakteristik in priprava predlogov za izboljšanje skladnosti** Pri tej fazi analiziramo rezultate predhodnih faz in na podlagi le-teh pripravimo predloge za izboljšanje skladnosti.

5. **Razprava o rezultatih** Pri tej fazi rezultate predstavimo vodstvu podjetja in prediskutiramo potencialne izboljšave stanja.

Model je predstavljen na Sliki 3.1.



Slika 3.1: Model ocenjevanja uspešnosti razvoja programske opreme z upoštevanjem vpliva organizacijskih karakteristik.

Poglavje 4

Študija primera

Postavljen model je bil preizkušen v okviru pluralne študije [75] izvedene v dveh podjetjih z različnimi organizacijskimi karakteristikami, ki je vključevala razvijalce, tehnične vodje in upravi podjetij. S študijo, v kateri je skupno sodelovalo 36 udeležencev, smo preverili delovanje, uporabnost in koristnost razvitega modela za potrebe izboljšanja procesa razvoja programske opreme. Na podlagi tega smo lahko ocenili, kje prihaja do odstopanj, kako jih lahko odpravimo, ter kako ta odstopanja vplivajo na uspešnost razvoja programske opreme. Rezultate smo predstavili vodstvu obeh podjetij in od njih pridobili povratno informacijo na podlagi katere smo lahko sklepali o koristnosti modela.

4.1 Opis podjetij

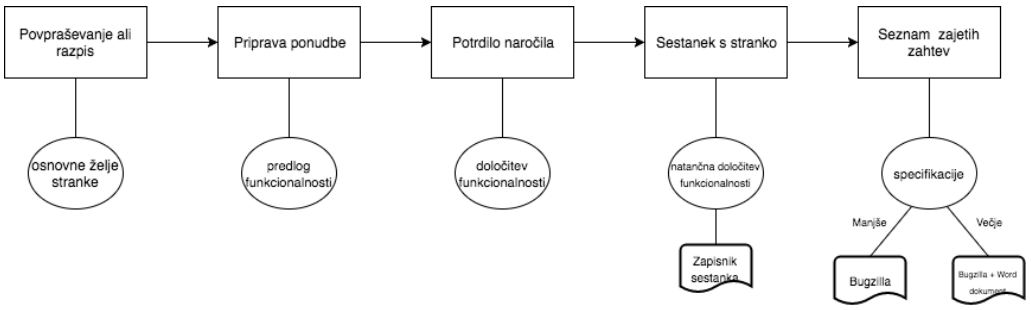
Študija primera je bila opravljena v dveh slovenskih podjetjih, ki se ukvarjata z razvojem programske opreme.

4.1.1 Podjetje A

Eno izmed v študijo vključenih podjetij je tipičen predstavnik uveljavljenega podjetja, ki je bilo ustanovljeno pred štirinajstimi leti. Ukvarja se z razvojem rešitev za elektronsko upravljanje z dokumentnim gradivom, avto-

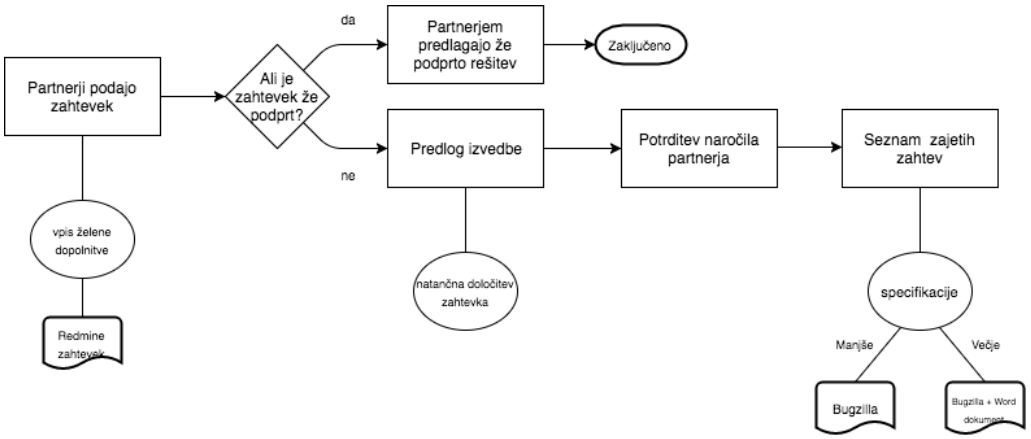
matizacijo izvajanja poslovnih procesov in elektronsko izmenjavo dokumentov. Rešitve podpirajo različne tipe dokumentov in postopkov dela (vhodna/izhodna pošta, pogodbe, likvidacija računov, nabava, dopusti, potni nalogi, reklamacije, vodenje sej in sestankov ter drugo). Podjetje ima šestnajst zaposlenih, od tega osem razvijalcev programske opreme. V podjetju smo prepoznali tri ključne deležnike: vodstvo podjetja, vodji ekip in zaposlene, ki so udeleženi v procesu razvoja programske opreme. V študiji primera je sodelovalo petnajst zaposlenih. Zaposleni so sodelovali pri delih študije, ki se nanaša na faze razvoja v katere so vključeni. Tako so pri fazi zajema zahtev sodelovali trije, pri načrtovanju arhitekture eden, pri implementaciji šest, pri testiranju dva in pri fazi postavitve trije.

Za začetek smo dobro spoznali proces dela pri posamezni fazi. Podjetje sodeluje s partnerji, ki jim pomagajo pri prodaji in sodelovanju s strankami. Pri zajemu zahtev imajo dva različna postopka, enega za interni zajem zahtev in drugega za zajem zahtev partnerskih podjetij. Postopek internega zajema zahtev je predstavljen na Sliki 4.1. Proces se prične z objavljenim razpisom ali poslanim povpraševanjem, kjer zajemalci zahtev dobijo osnovne smernice oziroma želje stranke. Temu sledi priprava ponudbe, kjer zajemalci zahtev na podlagi osnovnih smernic pripravijo predlog funkcionalnosti. Nato stranka potrdi naročilo, kjer bolje določijo funkcionalnosti, ki bi jih želeli imeti. Naročilu sledi sestanek s stranko, kjer se jim še natančneje predstavijo možne rešitve in skupaj določijo funkcionalnosti. Sestanek je dokumentiran z zapisnikom sestanka. Na njegovi podlagi pripravijo seznam zajetih zahtev. Če so zahteve manjše, jih vpišejo v spletno orodje Bugzilla, ki je namenjeno objavljanju in spremljanju stanja hroščev (angl. bug). Večje zahteve pa bolj podrobno predstavijo v Word dokumentu in prav tako vpišejo v orodje Bugzilla.



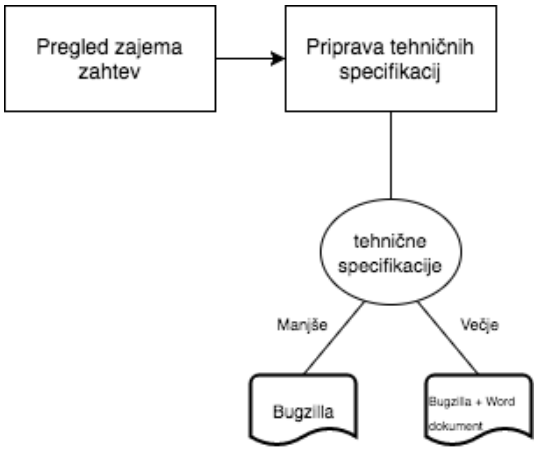
Slika 4.1: Zajem zahtev podjetja A pri internem postopku.

Kadar zahteve zajamejo zunanji partnerji, se uporablja postopek opisan na Sliki 4.2. Partnerji podjetju A podajo zajete zahteve v obliki zahtevka v spletnem orodju Redmine, ki se uporablja za vodenje projektov in spremljanje stanja zahtevkov. Pogosto se zgodi, da je zahtevek že podprt z obstoječo rešitvijo, vendar pa partnerji produkta ne poznajo dovolj dobro. V tem primeru jim je potrebno rešitev predstaviti in proces se tu zaključi. Kadar gre za nov zahtevek, zajemalci zahtev podjetja A pripravijo predlog izvedbe, kjer natančno določijo vsebino zahtevka. Partnersko podjetje nato v navezi s stranko potrdi naročilo. Zajemalci zahtev podjetja A nato pripravijo seznam zajetih zahtev; če so te manjše, jih vpišejo v Bugzillo, večje pa bolj podrobno predstavijo v Word dokumentu in prav tako vpišejo v Bugzillo.



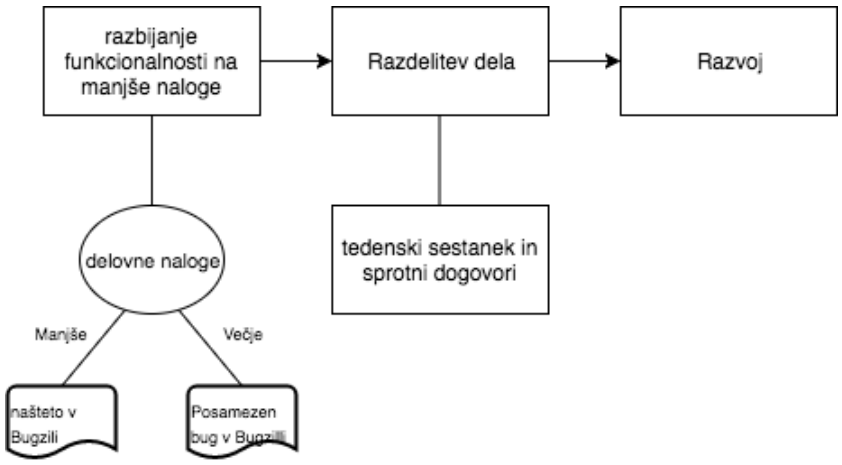
Slika 4.2: Zajem zahtev podjetja A pri postopku z zunanjimi partnerji.

Zajemu zahtev sledi faza načrtovanja arhitekture, ki je opisana na Sliki 4.3. Načrtovalci arhitekture pregledajo zajete zahteve. Na podlagi tega določijo, kako se bodo zahteve implementirale. Kompleksnejše probleme podrobno razdelajo in pripravijo tehnične specifikacije. Manjše zahteve zgolj vpišejo v Bugzillo, večjim pa v Bugzilli dodajo še Word dokument s podrobnejšimi opisi.



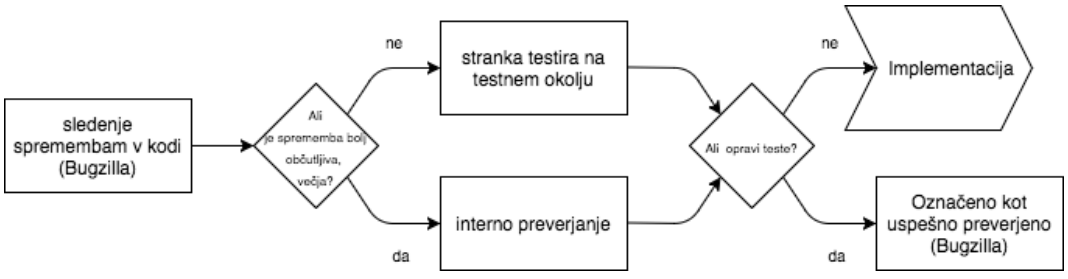
Slika 4.3: Načrtovanje arhitekture podjetja A.

Fazi načrtovanja arhitekture sledi faza implementacije, ki je predstavljena na Sliki 4.4. Implementacija se prične z razbijanjem funkcionalnosti na manjše delovne naloge. Manjše naloge se našteje v določenem zahtevku v Bugzilli, večje pa se v Bugzilli vpiše kot nov zahtevek. Sledi razdelitev dela. V grobem se delo razdeli na tedenskih sestankih, bolj podrobno pa se sproti ustno dogovori. Sledi razvoj.



Slika 4.4: Implementacija podjetja A.

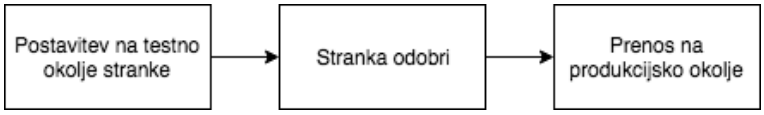
Naslednja je faza testiranja, ki je opisana na Sliki 4.5. Testerji z orodjem Bugzilla sledijo spremembam v kodi. Za vsako spremembo se odločijo ali je večja in bolj občutljiva. V približno polovici primerov se odločijo, da gre za večjo spremembo in jo interno testirajo, drugače pa jo stranka testira na testnem okolju. Stranke imajo ponavadi svoj IT oddelek ali IT osebo, ki je zadolžena za testiranje. Če sprememba ne opravi testiranja, se vrne v fazo implementacije. Ko pa je testiranje uspešno, se spremembo označi kot uspešno preverjeno v orodju Bugzilla.



Slika 4.5: Testiranje podjetja A.

Zadnja faza v procesu razvoja je postavitev, ki je predstavljena na Sliki 4.6. Začne se s postavitvijo na testno okolje stranke. Nato potrebujejo odobritev

s strani stranke, ki ji sledi prenos na produkcijsko okolje. S tem se proces razvoja zaključi.



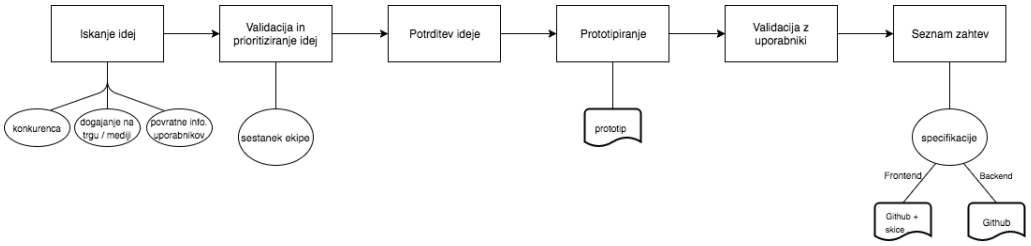
Slika 4.6: Postavitev podjetja A.

4.1.2 Podjetje B

Drugo podjetje je predstavnik mladega, zagonskega podjetja. Podjetje razvija aplikacijo, ki elektronski pošti dodaja klepet in druge izboljšave. Podjetje ima trideset zaposlenih in tri ekipe, ki se ukvarjajo z razvojem programske opreme. Ena izmed ekip je zadolžena za zaledni sistem, druga za razvoj mobilne aplikacije za iOS operacijski sistem in tretja za razvoj namizne aplikacije. Tudi v tem podjetju smo prepoznali tri ključne deležnike, vodstvo podjetja, vodje ekip in zaposlene, ki so udeleženi v procesu razvoja programske opreme. V študiji je sodelovalo enaindvajset zaposlenih, in sicer dva pri merjenju faze zajema zahtev, trije pri načrtovanju arhitekture, devet pri implementaciji, trije pri testiranju in dva pri postavitvi. Poleg tega sta sodelovala tudi dva predstavnika vodstva podjetja in tri vodje posameznih ekip.

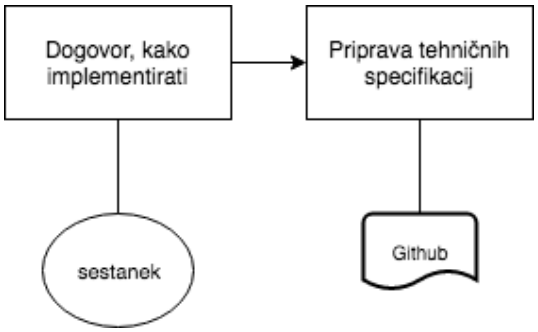
Podjetje B je zagonsko podjetje, tako da postopki dela niso natančno predpisani in so prilagodljivi situaciji. Za potrebe študije smo opisali tipičen proces dela v podjetju. Proces razvoja se začne s fazo zajema zahtev, ki je prikazana na Sliki 4.7. Prvi korak je iskanje idej, ki jih lahko dobijo na podlagi spremljanja konkurence in dogajanja na trgu z branjem ustreznih medijev. Prav tako ideje dobijo iz povratnih informacij obstoječih uporabnikov aplikacije. Nato sledi validacija idej in prioritiziranje, kar poteka na sestanku ekipe. Idejo nato potrdijo in izdelajo prototip z orodji kot na primer proto.io. Kasneje naredijo validacijo z uporabniki. Zadnji korak pa je priprava seznama zahtev, kjer specifikacijo vpišejo v orodje Github. Zah-

teve, namenjene frontend ekipam, imajo priložene tudi skice, ki jih ponavadi naložijo v programu Zeplin.



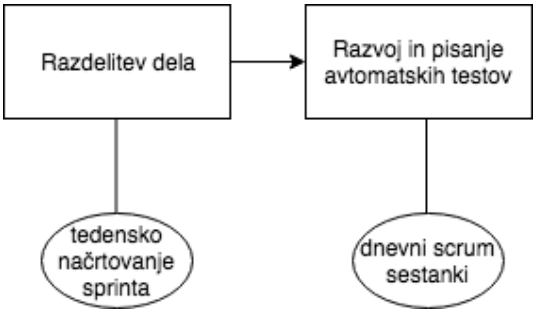
Slika 4.7: Zajem zahtev podjetja B.

Zajemu zahtev sledi faza načrtovanja arhitekture, ki je prikazana na Sliki 4.8. Faza se prične s sestankom, kjer se dogovorijo, kako implementirati določene zahteve. Na podlagi tega dogovora nato pripravijo tehnične specifikacije, ki jih vnesejo v orodje GitHub.



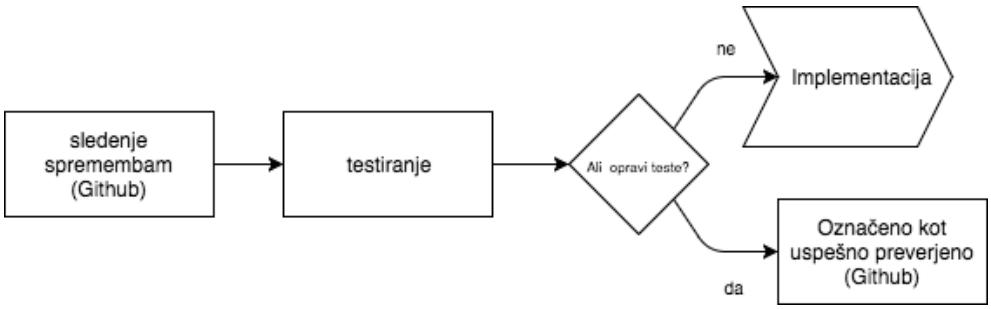
Slika 4.8: Načrtovanje arhitekture podjetja B.

Sledi faza implementacije, ki je opisana na Sliki 4.9. Implementacija se začne z razdelitvijo dela, ki jo naredijo na tedenskih načrtovanjih sprinta. Temu sledi razvoj in pisanje avtomatskih testov. Ekipe imajo dnevne scrum sestanke.



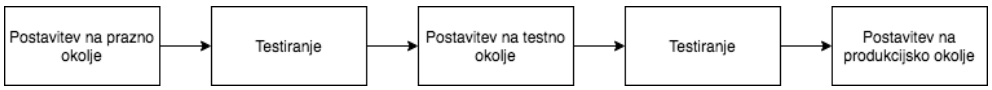
Slika 4.9: Implementacija podjetja B.

Faza testiranja je opisana na Sliki 4.10. Prične se s sledenjem sprememb na orodju GitHub. Spremembe nato testirajo. Če spremembe ne opravijo testov, se vrnejo v fazo implementacije, drugače pa se označijo kot uspešno preverjene v orodju GitHub.



Slika 4.10: Testiranje podjetja B.

Zadnja faza v procesu razvoja je faza postavitve, ki je opisana na Sliki 4.11. Začne se s postavitvijo na prazno okolje. Sledi testiranje. Uspešnemu testiranju sledi postavitev na testno okolje s testnimi uporabniškimi računi. Ponovno sledi testiranje. Zadnji korak, ki sledi uspešnemu testiranju, je postavitev na produkcijsko okolje.



Slika 4.11: Postavitev podjetja B.

4.2 Izvedba študije

Študija primera je razdeljena na 6 faz, in sicer:

1. **Načrtovanje študije** Pri začetni fazi smo dobro spoznali obe podjetji in za vsako definirali ključne vloge. Člane podjetij smo razdelili v 4 primarne vloge:

   - vodstvo podjetja,
   - vodje ekip,
   - zaposlene, ki opravljajo delo povezano z razvojem programske opreme,
   - ostale zaposlene.

   Za vsakega zaposlenega, ki opravlja delo povezano z razvojem programske opreme, smo določili v katerih razvojnih fazah je udeležen.

2. **Popis procesa dela pri posamezni razvojni fazi** V drugi fazi smo se pogovarjali z vodji ekip, zato da smo dobro spoznali proces dela in uporabo metodologij za razvoj programske opreme v obeh podjetjih. Rezultate te faze smo podrobno opisali v prejšnjem poglavju.

3. **Evalvacija stanja po posameznih razvojnih fazah** V nadaljevanju smo želeli ugotoviti katere razvojne faze so najbolj problematične, kar smo izvedeli s pomočjo vprašalnikov o uspešnosti posameznih razvojnih faz, ki so jih izpolnili vodje podjetij. Uspešnost so ocenjevali po kriterijih železnega trikotnika. Za bolj natančno razumevanje problema smo vprašalnike podkrepili še z dodatnimi pogovori o problematičnih

fazah, kjer so nam vodje podjetji bolj podrobno razložili težave posameznih faz. Obenem pa smo zaposlene povprašali o zadovoljstvu z načinom dela v posameznih fazah. Zanimalo nas je tudi, ali se ocenjevanje uspešnosti posameznih faz s strani vodstva ujema z zadovoljstvom zaposlenih z načinom dela v določeni fazi.

4. **Evalvacija stanja organizacijskih karakteristik** V tej fazi smo ocenili kakšne so organizacijske karakteristike obeh podjetji. To smo ocenili na podlagi vprašalnika, ki so ga izpolnili vodje ekip in vodstvo podjetja. Kasneje smo preverili še vpliv organizacijskih karakteristik na delo zaposlenih v vsaki posamezni razvojni fazi. To smo preverili s pomočjo petih različnih vprašalnikov, ki so jih izpolnili zaposleni vključeni v posamezno razvojno fazo. Nato smo ugotovitve preverili s pomočjo intervjujev z zaposlenimi, ki so nam dali še boljši vpogled.

5. **Analiza** Na podlagi študije smo nato izpostavili karakteristike, ki najbolj pozitivno in najbolj negativno vplivajo na zaposlene. Še posebej smo bili pozorni na karakteristike, ki različno vplivajo na zaposlene v fazah, ki jih je vodstvo ocenilo kot najbolj uspešne, in tistimi, ki so jih ocenili kot najmanj uspešne.

6. **Interpretacija rezultatov** Nato smo na podlagi tega pripravili predloge, ki bi izboljšali skladnost procesa razvoja programske opreme z organizacijskimi karakteristikami podjetja. Rezultate študije in predloge izboljšav smo na koncu predstavili vodstvu podjetij.

4.3 Analiza rezultatov

4.3.1 Podjetje A

Rezultati raziskave ocenjevanja uspešnosti in zadovoljstva v podjetju A so prikazani v Tabeli 4.1, kjer vidimo povprečne ocene uspešnosti za posamezne razvojne faze, ki so jih podali v vodstvu podjetja, in zadovoljstvo zaposlenih

z načinom dela v posamezni razvojni fazi. Strinjanje s posamezno trditvijo so ocenjevali po sedemstopenjski lestvici. Vodstvo podjetja je najnižje ocenilo faze zajema zahtev, testiranje in postavitev. Fazi implementacije in načrtovanja arhitekture je ocenilo nadpovprečno. Prav tako lahko vidimo, da so zaposleni najmanj zadovoljni z načinom dela pri zajemu zahtev, testiranju in postavitvi. Zaposleni so najbolj zadovoljni z načinom dela pri načrtovanju arhitekture in implementaciji.

Tabela 4.1: Ocene uspešnosti in zadovoljstva po posameznih razvojnih fazah.

	Vodstvo podjetja	Zaposleni
Zajem zahtev	3.83	4.67
Načrtovanje arhitekture	5.58	6.00
Implementacija	5.67	6.33
Testiranje	4.83	5.00
Postavitev	4.00	5.33

Iz tega lahko ugotovimo, da vodstvo podjetja vidi enake probleme kot jih vidijo zaposleni. To pomeni, da ima podjetje zelo dobro osnovo za izboljšanje procesa razvoja, saj različni deležniki vidijo težave na enakih mestih. Timski duh je močno prisoten v tem podjetju.

Vprašalniki o organizacijskih karakteristikah so pokazali, da na splošno na delo zaposlenih najbolj pozitivno vplivajo karakteristike predstavljene v Tabeli 4.2. Prva dva stolpca prikazujeta stopnjo prisotnosti posamezne karakteristike po oceni vodstva podjetja in vodij ekip. Nadaljnji stolpci pa prikazujejo vpliv karakteristik na delo zaposlenih v posamezni razvojni fazi. Na prvem mestu karakteristik, ki najbolj pozitivno vplivajo na delo zaposlenih v vseh razvojni fazah, je **solidarnost**, za katero so tudi vodstvo in vodje ekip mnenja, da je na zelo visokem nivoju. Intervjuji z zaposlenimi so pokazali, da v vseh razvojnih fazah karakteristika solidarnosti zelo pozitivno

vpliva na njihovo delo. Povedali so, da je njihov produkt zelo obsežen in da nihče ne ve vsega, tako da je res pomembno, da so sodelavci pripravljeni pomagati in pojasniti stvari, če je to potrebno.

Poleg tega so vprašalniki pokazali tudi pozitiven vpliv karakteristik **profesionalizacije, podpore zaposlenih, strukture opravljanja delovnih nalog in centralizacije** na delo zaposlenih v vseh razvojnih fazah. Vprašalniki o stanju organizacijskih karakteristik so pokazali, da tudi vodstvo podjetja in vodje ekip menijo, da sta karakteristiki profesionalizacije in podpore zaposlenih na visokem nivoju, da naloge opravljajo večinoma bolj skupinsko in da je podjetje bolj decentralizirano. Zaposleni so v intervjujih povedali, da je za njihovo delo zelo pozitivno to, da je najbolj pomembna kakovost opravljenega dela, pri načinu opravljanja nalog pa je zaželena svoboda. Zaposleni, ki imajo predhodne izkušnje v drugih organizacijah, so povedali, da je v tem podjetju podpora nadrejenih na veliko višjem nivoju kot drugje, kar zelo pozitivno vpliva na njihovo delo. Zaposleni so tudi pohvalili raven profesionalizacije, ki se jim zdi pri razvoju programske opreme zelo pomembna za dobro opravljanje dela. Izpostavili so tudi, da je struktura opravljanja delovnih nalog primerna za dobro opravljanje dela.

Tabela 4.2: Karakteristike, ki najbolj pozitivno vplivajo na delo zaposlenih podjetja A.

	vodstvo podjetja	vodje ekip	zajem zahtev	načrtovanje arhitekture	implementacija	testiranje	postavitev
KULTURA							
Podpora zaposlenih	5.5	6	5	7	6.17	5.5	5.67
Solidarnost	6	7	5	7	6.83	6.5	6.67
STRUKTURE							
Decentralizacija	5.5	6	6	7	5.5	7	6.67
Struktura opravljanja delovnih nalog	6	5.5	5	7	6	6	5.67
Profesionalizacija	5.5	6	5.33	6	6.67	6	5.33

V Tabeli 4.3 so prikazane karakteristike, pri katerih je opaziti največjo razliko pri vplivu na delo zaposlenih v fazah, ki jih je vodstvo podjetja ocenilo kot uspešne in ostalimi fazami.

Kontrola - Pri karakteristiki kontrole lahko opazimo veliko razliko med zaposlenimi, ki delajo v fazah, pri katerih je z načinom dela zadovoljno tako vodstvo podjetja kot zaposleni, in zaposlenimi v fazah, pri katerih niti vodstvo niti zaposleni niso zadovoljni z načinom dela. Vodstvo podjetja in vodje ekip so ocenili, da je stopnja kontrole nizka, s čimer so se v pogovorih strinjali tudi zaposleni, ki delajo v fazah zajema zahtev, testiranja in postavitve. Zaposleni so mnenja, da bi v teh fazah potrebovali malo več nadzora, saj bi s tem dosegli večji pregled nad delom. Pogosto ne vedo, kaj njihovi sodelavci delajo. Pogosto samo vodja ve, kaj se dogaja in ostali nimajo informacij. Zaposleni v fazi zajema zahtev so povedali, da bi bilo zelo dobro vedeti s kakšnimi težavami se soočajo sodelavci, ker se včasih zgodi, da rešujejo podoben problem, kot ga je nekdo že imel. Ker ne vedo, kaj delajo sodelavci, ponovno rešujejo problem in nimajo informacij, kako je bila težava rešena. Izpostavili so tudi, da se morajo včasih pogovarjati s strankami, za katere je zadolžen kateri od sodelavcev in takrat bi jim informacije o tem, kaj delajo sodelavci, prišle zelo prav.

Predlog 1: Tedenski sestanki zaposlenih v fazah zajema zahtev in testiranja

Predlog je bil zaposlenim v fazah zajema zahtev in testiranja zelo všeč. Povedali so, da so sestanke imeli, vendar so z njimi prenehali pred nekaj leti zaradi zdravstvenih težav vodje ekipe. Na sestankih bi bilo potrebno uskladiti prioritete zahtev strank. Zaposleni v fazi implementacije včasih dobijo več podobnih zahtev hkrati in v teh primerih bi bilo potrebno, da se zajemalci zahtev med seboj uskladijo glede prioritet. To bi lahko naredili na tedenskih sestankih. Zaposleni v fazah implementacije in načrtovanja arhitekture že imajo redne tedenske sestanke in so zadovoljni s stopnjo nadzora pri njihovem delu.

Sistem nagrajevanja - Tudi pri karakteristiki sistema nagrajevanja opa-

zimo razliko med odgovori zaposlenih, ki delajo v fazah, kjer imajo dober način dela, in odgovori zaposlenih v ostalih fazah. Prvi so mnenja, da trenutni sistem nagrajevanja dobro vpliva na njihovo delo, drugi pa pravijo, da trenutni sistem slabo vpliva na njihovo delo. Zaposleni so povedali, da so lansko leto uvedli nagrajevanje v obliki dodatka pri decembrski plači, vendar kriteriji nagrajevanja niso bili jasno določeni. Nekateri se tudi počutijo neopažene, zdi se jim, da vodstvo ne ve kaj počnejo.

Predlog 2: Letni pogovori vodstva z zaposlenimi

Vodstvo bi lahko opravilo letne pogovore z vsemi zaposlenimi. Na pogovoru bi zaposleni na kratko predstavili, kaj so naredili v preteklem letu in kaj so njihovi cilji za prihodnje leto. Pogovorili bi se lahko tudi o težavah in kariernih ciljih. Pogovori bi vodstvu dali informacije o delu zaposlenih na podlagi katerih bi lažje nagrajevali. Zaposlenim bi pogovor omogočil načrtovanje kariere ter jim preprečil občutek, da so obstali. Hkrati bi bili zaposleni s pogovorom opaženi, prejeli pa bi lahko tudi povratno informacijo o svojem delu.

Tabela 4.3: Karakteristike, s katerimi so zaposleni v slabo ocenjenih fazah najmanj zadovoljni.

	vodstvo podjetja	vodje ekip	zajem zahtev	načrtovanje arhitekture	implementacija	testiranje	postavitev
KULTURA							
Kontrola	3	2	2.33	7	5.67	4	4.67
STRUKTURE							
Sistem nagrajevanja	6	5	2.67	6	5.67	4.5	4.5

Karakteristike, ki najslabše vplivajo na zaposlene v vseh fazah, so prikazane v Tabeli 4.4.

Formalizacija - Zaposleni v vseh fazah so mnenja, da stopnja formalizacije slabo vpliva na njihovo delo. Kot je razvidno iz odgovorov vodij podjetja in ekip, je stopnja formalizacije relativno nizka. Zaposleni so v intervjujih povedali, da bi bilo za njihovo delo bolje, da bi imeli več predpisanih protokolov. To se predvsem kaže pri fazi testiranja in postavitve. Pri testiranju imajo večkrat dilemo, v katerih primerih se spremembe v kodi testirajo interno in v katerih jih testira stranka. Pri postavitvi pa zaposleni pravijo, da občasno ne vedo, kdaj je primerno dati nekaj na produkcijsko okolje in kdaj potrebujejo dovoljenje strank.

Predlog 3: Sestanek vodje razvoja, zaposlenih v fazah postavitve, testiranja in vseh, ki imajo stik s strankami.

Na skupnem sestanku bi lahko obravnavali omenjena vprašanja in postavili smernice. Lahko bi naredili bolj natančna navodila za testiranje in postavitev.

Predlog 4: Zaposlitev novega testerja

Zelo koristno bi bilo, da bi zaposlili vsaj še eno osebo za testiranje, saj se trenutno s testiranjem ena oseba ukvarja polovico delovnega časa, druga četrtino in tretja četrtino.

Struktura povezav podjetja z okoljem - Vprašalniki so pokazali, da zaposleni v vseh fazah razvoja menijo, da struktura povezav podjetja z okoljem slabo vpliva na njihovo delo. Ob pogovoru z zaposlenimi smo ugotovili, da karakteristika strukture povezav podjetja z okoljem ni prioritetna, vendar bi bilo dobrodošlo, da bi imeli dostop do več virov informacij.

Predlog 5: Izobraževanja in obisk konferenc

Razmislili bi lahko o dodatnih izobraževanjih oziroma tečajih ali pa obiskih strokovnih konferenc oziroma seminarjev. S tem bi zaposleni pridobili boljši pregled nad novimi tehnologijami in dogajanjem na področju njihove industrije.

Tabela 4.4: Karakteristike, ki najslabše vplivajo na zaposlene v vseh fazah.

	vodstvo podjetja	vodje ekip	zajem zahtev	načrtovanje arhitekture	implementacija	testiranje	postavitev
KULTURA							
Neprestano učenje	4	3.5	4	5	5.25	3	3.5
STRUKTURE							
Formalizacija	3	3	3.33	3	4.17	4	5.33
Struktura povezav podjetja z okoljem	5.5	4.5	4	4	4.83	3.5	4.33
STRATEGIJE							
Strategija obrambe tržnega položaja	4	4.5	3.67	3	4.83	5.5	4.33
Strategija sodelovanja	5	7	3.67	2	4	6	5.33

4.3.2 Podjetje B

Raziskava uspešnosti in zadovoljstva v podjetju B je pokazala, da vodstvo podjetja najslabše ocenjuje uspešnost faz implementacije in testiranja, kar lahko vidimo pri Tabeli 4.5. V pogovorih z vodstvom podjetja smo izvedeli, da vidijo največjo možnost izboljšave v fazi implementacije, in sicer konkretno pri kvaliteti implementacije ene izmed treh delovnih ekip. Vodstvo podjetja je nadpovprečno zadovoljno s fazami načrtovanja arhitekture, postavitve in zajema zahtev. Prav tako lahko vidimo, da so zaposleni najmanj zadovoljni z načinom dela pri implementaciji in testiranju. Zaposleni so najbolj zadovoljni z načinom dela pri načrtovanju arhitekture, postavitvi in zajemu zahtev.

Tabela 4.5: Ocene uspešnosti in zadovoljstva po posameznih razvojnih fazah.

	Vodstvo podjetja	Zaposleni
Zajem zahtev	4.66	6.00
Načrtovanje arhitekture	5.33	6.00
Implementacija	3.66	4.00
Testiranje	4.00	5.00
Postavitev	4.66	6.00

Prav tako kot pri podjetju A lahko tudi tu opazimo, da vodstvo podjetja vidi enake probleme, kot jih vidijo zaposleni. Kar pomeni, da tudi pri podjetju B različne skupine vidijo težave na enakih mestih.

V Tabeli 4.6 so predstavljene karakteristike, ki najbolj pozitivno vplivajo na delo zaposlenih v podjetju B. Na prvem mestu sta karakteristiki **solidarnosti in strukture opravljanja delovnih nalog**. Zaposleni so povedali, da se jim zdi na delovnem mestu ključnega pomena solidarnost, ki je močno prisotna v tem podjetju. Povedali so tudi, da jim močno ustreza struktura opravljanja delovnih nalog, ki jih v veliki meri opravljajo individualno.

Poleg tega rezultati kažejo, da na delo zaposlenih zelo pozitivno vpliva tudi **podpora zaposlenih in profesionalizacija**. Zaposleni so povedali, da imajo dobro podporo s strani vodstva podjetja, kar zelo dobro vpliva na njihovo delo. Prav tako so povedali, da na njihovo delo ključno vpliva profesionalizacija, saj je delo zahtevno. Pri večini faz so zaposleni pohvalili raven profesionalizacije in povedali da jim le-ta omogoča bolj kvalitetno delo. V fazi postavitve so rezultati pokazali težavo pri profesionalizaciji. Pri pogovoru z zaposlenimi smo izvedeli, da nivo profesionalizacije enega člana ekipa ni dovolj visok in imajo zaradi tega obilico težav.

Predlog 1: menjava zaposlenega

Na delo v fazi postavitve bo pozitivno vplivala menjava zaposlenega, ki ne dosega ravni profesionalizacije preostalih članov ekipe.

Tabela 4.6: Karakteristike, ki najbolj pozitivno vplivajo na delo zaposlenih podjetja B.

	vodstvo podjetja	vodje ekip	zajem zahtev	načrtovanje arhitekture	implementacija	testiranje	postavitev
KULTURA							
Podpora zaposlenih	6	6	6.50	4.33	6	6.67	6.5
Solidarnost	6	6.33	6.5	6	6.56	6.67	7
STRUKTURE							
Struktura opravljanja delovnih nalog	6	6.33	6.5	4.33	6.11	7	7
Profesionalizacija	6	5.67	6.5	5.33	5.89	6.33	4

V Tabeli 4.7 so prikazane karakteristike, ki najslabše vplivajo na zaposlene podjetja B.

Specializacija - Zaposleni v fazi testiranja so izpostavili, da so njihove delovne naloge ozko specializirane na določeno področje. V podjetju so tri razvojne ekipe - ekipa, ki se ukvarja z zalednim delom, ekipa, ki se ukvarja z namizno aplikacijo in ekipa za iOS aplikacijo. Vsaka ekipa ima svojega testerja. Izvedeli smo, da se včasih zgodi, da se na neki platformi nekaj implementira, kar vpliva tudi na druge, vendar tega ne izvedo. Najbolj problematično je to pri testiranju zalednega dela aplikacije, ki se lahko testira le preko ene izmed platform. Večkrat se zgodi, da tester zaledne ekipe ne izve za spremembe na posamezni platformi, kar mu oteži delo.

Predlog 2: Sestanki zaposlenih v fazi testiranja

Dobro bi bilo, da bi tudi testerji med seboj delovali kot ekipa. Tedenski sestanki bi izboljšali komunikacijo in omogočili izmenjavo dobrih praks med testerji.

Kontrola - Zaposleni so mnenja, da stopnja kontrole slabo vpliva na njihovo delo. Vodstvo podjetja in vodje posameznih ekip ocenjujejo, da je stopnja kontrole nizka. Problematika je najbolj očitna pri fazi implementacije, saj pogosto prihaja do velikega števila napak.

Predlog 3: Tedensko predstavljanje novosti

Celotno podjetje ima enkrat tedensko sestanke, kjer ponavadi marketinški oddelek in vodstvo podjetja predstavita trenutne izzive. Na tem sestanku bi bilo dobro dodati še predstavitve ekip mobilne in namizne aplikacije. Predstavili bi lahko funkcionalnosti, ki so jih implementirali v preteklem tednu. S tem bi bili vsi dobro obveščeni o novostih, hkrati pa bi imeli tudi razvojniki priložnost pokazati svoje delo in se pohvaliti. S tem bi tudi izboljšali kvaliteto razvoja, saj bi se razvojniki počutili bolj odgovorno, če bi morali javno predstavljati svoje delo sodelavcem.

Neprestano učenje - Zaposleni v vseh fazah so mnenja, da stopnja neprestanega učenja slabo vpliva na njihovo delo. Kot je razvidno iz odgovorov vodij podjetja in ekip, je stopnja neprestanega učenja relativno nizka.

Zaposleni so v intervjujih izpostavili, da imajo občutek, da držanje predpisanih rokov na njihovo delo nima vpliva. V podjetju imajo elemente scrum metodologije, vendar se jih ne držijo popolnoma.

Predlog 4: Sestanki za izboljšanje

Po vsakem roku bi bilo dobro imeti retrospektivne sestanke kot jih predlaga metodologija scrum. Na sestankih za izboljšanje bi pohvalili zaposlene, če so se držali roka, ali pa se pogovorili o razlogih za zamudo, če bi rok prekoračili. Tako bi lahko izboljšali ocene pri postavljanju rokov in dali zaposlenim motivacijo, da se rokov držijo.

Sistem nagrajevanja - Zaposleni v vseh fazah z izjemo faze načrtovanja arhitekture so ocenili, da sistem nagrajevanja slabo vpliva na njihovo delo. V pogovorih z njimi smo izvedeli, da jim karakteristika sistema nagrajevanja ni prioritetna, vendar se jim zdi ključno, da bi bili nagrajeni za lojalnost podjetju.

Predlog 5: Nagrada za uspešno priporočilo novega zaposlenega

Zaposlene bi iniciativa spodbujala k priporočanju novih zaposlenih in jih v primeru enoletnega uspešnega sodelovanja s priporočenim kandidatom nagradila. To bi obenem pomagalo tudi pri uspešnem zaposlovanju novih kadrov.

Tabela 4.7: Karakteristike, ki najbolj negativno vplivajo na delo zaposlenih podjetja B.

	vodstvo podjetja	vodje ekip	zajem zahtev	načrtovanje arhitekture	implementacija	testiranje	postavitev
KULTURA							
Kontrola	3	3.33	5.5	3.33	4.67	5.67	6
Neprestano učenje	5	3	5.5	5	4.67	5	4.5
STRUKTURE							
Specializacija	4	5.33	6	5	5.33	4	6
Sistem nagrajevanja	6	6	4.5	5.33	4.56	5	5

4.4 Odziv vodstva

4.4.1 Podjetje A

Vodstvu podjetja A smo predstavili rezultate študije in jih prosili za povratne informacije. Vodstvo se je strinjalo, da so faze, ki bi se jih dalo najbolje izboljšati, faze zajema zahtev, testiranja in postavitve. Smiselne so se jim zdele tudi karakteristike, ki najbolj pozitivno vplivajo na delo zaposlenih.

Mnenje vodstva podjetja A o posameznih predlogih za povečanje skladnosti med procesom razvoja in organizacijskimi karakteristikami:

1. **Tedenski sestanki zaposlenih udeleženih v fazah zajema zahtev in testiranja** Vodstvo se strinja, da bi morali to uvesti. Med vsemi predlogi za povečanje skladnosti med procesom razvoja in organizacijskimi karakteristikami se jim zdi ta najpomembnejši. Predlog želijo uvesti že z naslednjim tednom.

2. **Letni pogovori vodstva z zaposlenimi** Predlog se jim zdi zelo smiseln. Ob predlogu so se spomnili na zaposlenega, ki jih je pred kratkim zapustil in se zdaj ponovno vrnil. So mnenja, da je podjetje zapustil, ker se v podjetju niso dovolj dobro zavedali, da bi se zaposleni želel usmeriti v določeno področje, kar so mu zdaj, po odpovedi in ponovni priključitvi podjetju, omogočili. Zdi se jim, da bi se tovrstne težave lahko odpravile, če bi vpeljali predlagane pogovore z zaposlenimi. Prepoznali so, da bi z uvedbo pridobili tudi nov način prepoznave dodane vrednosti posameznega zaposlenega in morebitnih težav s katerimi se pri delu sooča. Pogovore želijo vpeljati kot stalno prakso, ki bi se odvijala vsaj dvakrat letno, oziroma, če bi se izkazalo koristno, na vsake tri mesece. Z iniciativo bi lahko začeli v decembru, da s tem zaključijo letošnje leto in postavijo cilje za prihodnje.

3. **Določitev smernic za testiranje in postavitev** Predlog se zdi vodstvu podjetja A smiseln, vendar se jim ne zdi prioriteten. Ker predlog

ne zahteva veliko truda in časa, ga nameravajo izvesti v naslednjem mesecu.

4. **Zaposlitev novega testerja** Vodstvo se zaveda, da imajo premalo testerjev, vendar pravijo, da si zaradi pričakovanj lastnikov podjetja tega ne morejo privoščiti. Del testiranja imajo namen preložiti na partnerska podjetja.

5. **Izobraževanja oziroma obisk konferenc** S predlogom se vodstvo strinja. Izobraževanja in konference se jim zdijo zelo smiselne. Kot prvi korak bodo v prihodnjem mesecu vprašali zaposlene, o čem bi se želeli izobraževati, in nato organizirali nekaj primernega.

4.4.2 Podjetje B

Vodstvu podjetja B smo predstavili rezultate študije in jih prosili za povratne informacije. Vodstvo se je strinjalo, da je najbolj problematična faza implementacije in najbolj problematično področje kvalitete implementacije. Karakteristike, ki najbolj pozitivno vplivajo na delo zaposlenih, so se jim zdele smiselne.

Mnenje vodstva podjetja B o posameznih predlogih za povečanje skladnosti med procesom razvoja in organizacijskimi karakteristikami:

1. **Menjava zaposlenega** Menjava je že v teku. Trenutno potekajo razgovori s potencialnimi kandidati za prevzem delovnega mesta.

2. **Sestanki zaposlenih v fazi testiranja** Pred časom so že želeli uvesti sestanke testne skupine, vendar zaradi menjave načina dela in menjave kadra do sestankov ni prišlo. Član testne skupine je pred časom prevzel vodstveno funkcijo pri ekipi zalednega sistema, zato želijo počakati, da se v tej vlogi uveljavi in nameravajo predvidoma čez en mesec vpeljali sestanke zaposlenih v fazi testiranja.

3. **Tedensko predstavljanje novosti** S predlogom se strinjajo in si ga želijo vpeljati že s tem tednom. Vodjem ekip so naročili, da pripravijo

demonstracijo najbolj pomembnih novosti implementiranih v tekočem tednu.

4. **Sestanki za izboljšanje** Povedali so, da se pri njih stvari konstantno spreminjajo, zato so delovne naloge znane zgolj za enotedenski sprint. Najbolj pomembno se jim zdi, da so člani ekip seznanjeni s prioritizacijo delovnih nalog in da so le-te smiselno zastavljene. Predlog se jim zdi smiseln in nameravajo uvesti retrospektivne sestanke za vse ekipe ob mejniku, ki ga imajo čez deset dni.

5. **Nagrada za uspešno priporočilo novega zaposlenega** Ideja se jim zdi dobra. Uvedli bi nagrado v vrednosti 250 eur za priporočilo novega zaposlenega, ki uspešno opravi trimesečno poskusno dobo. O uvedbi nagrade bodo zaposlene obvestili na petkovem tedenskem sestanku vseh zaposlenih.

Poglavje 5

Sklepi

Glavni cilj magistrskega dela je bil razvoj modela za ocenjevanje uspešnosti razvoja programske opreme, ki pri ocenjevanju upošteva tudi skladnost z organizacijskimi karakteristikami podjetja.

Za začetek smo v Poglavju 2 naredili izbor relevantne literature s pregledom najpomembnejših metodologij razvoja programske opreme in kriterijev ocenjevanja uspešnosti, ki se uporabljajo pri razvoju programske opreme. Prav tako smo pregledali in strnili literaturo s področja organizacijskih karakteristik. Na podlagi pregleda literature smo nato v Poglavju 3 razvili celosten model ocenjevanja uspešnosti programske opreme, ki upošteva organizacijske karakteristike, ki smo jih identificirali kot ključne za uspešnost uporabe informacijskih tehnologij v podjetjih. Razviti model uporablja pet ključnih faz razvoja programske opreme, ki so opredeljene v metodologiji RUP [56], in sicer zajem zahtev, načrtovanje arhitekture, implementacijo, testiranje in postavitev. Kriteriji za ocenjevanje uspešnosti razvitega modela so najbolj pogosto uporabljeni kriteriji tako imenovanega železnega trikotnika [5] ter kriterij zadovoljstva uporabnikov, ki ga v svojem modelu definirata McLean in DeLone [26].

V nadaljevanju je bilo potrebno preveriti delovanje, uporabnost in koristnost razvitega modela za potrebe izboljšanja procesa razvoja programske opreme. Študijo smo opisali v Poglavju 4. Izvedli smo jo v dveh slovenskih

podjetjih, ki se ukvarjata z razvojem programske opreme. Z uporabo razvitega modela smo najprej dobro spoznali obe podjetji, definirali ključne vloge in popisali proces dela za vsako posamezno razvojno fazo. Nato smo z vodstvom podjetja identificirali najbolj problematične razvojne faze in obenem pridobili informacijo o zadovoljstvu zaposlenih z načinom dela v posamezni fazi. V nadaljevanju smo izvedli evalvacijo stanja organizacijskih karakteristik s pomočjo vprašalnika, ki so ga izpolnili tehnični vodje in vodstvo podjetij. Temu je sledilo preverjanje vpliva organizacijskih karakteristik na delo zaposlenih v vsaki posamezni razvojni fazi. Na podlagi pridobljenih informacij smo nato naredili analizo, kjer smo ocenili, kje prihaja do odstopanj med organizacijskimi karakteristikami in načinom razvoja programske opreme in kako te vplivajo na uspešnost razvoja. Na podlagi tega smo pripravili interpretacijo rezultatov, kjer smo pripravili predloge kako odpraviti odstopanja in uskladiti proces dela z organizacijskimi karakteristikami. Za konec smo analizo in predloge za izboljšavo predstavili vodstvu obeh podjetij in od njih pridobili povratno informacijo. Vodstvi obeh podjetij sta sprejeli večino predlogov za izboljšanje skladnosti med procesom razvoja in organizacijskimi karakteristikami njihovega podjetja. Na podlagi tega lahko sklepamo, da je razviti model koristen za usklajevanje organizacijskih karakteristik in procesa razvoja programske opreme. Tako nam je uspelo doseči vse cilje, ki smo si jih v magistrskem delu zastavili.

Model se je izkazal kot uspešen v obeh podjetjih, vključenih v študijo, s čimer lahko sklepamo o njegovi splošni koristnosti. Vendar pa bi lahko študijo razširili in uporabo razvitega modela preverili tudi v podjetjih z večjim številom zaposlenih, smiselno bi bilo vključiti tudi podjetja iz tujine, saj smo se pri študiji osredotočili zgolj na manjša do srednje velika slovenska podjetja.

Literatura

[1] P. Abrahamsson, J. Warsta, M. T. Siponen, in J. Ronkainen, "New directions on agile methods: a comparative analysis," v *Software Engineering, 2003. Proceedings. 25th International Conference on.* Ieee, 2003, str. 244–254.

[2] M. O. Ahmad, J. Markkula, in M. Oivo, "Kanban in software development: A systematic literature review," v *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on.* IEEE, 2013, str. 9–16.

[3] D. J. Anderson, "Business drivers for kanban adoption," v *Proceedings of Lean Software & Systems Conference*, 2010, str. 7–14.

[4] ——, *Kanban: successful evolutionary change for your technology business.* Blue Hole Press, 2010.

[5] R. Atkinson, "Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria," *International journal of project management*, zv. 17, št. 6, str. 337–342, 1999.

[6] K. A. Baker, "Organizational culture," *Washington, DC: US Department of Energy, Office of Science*, 2002.

[7] J. Y. Bakos in M. E. Treacy, "Information technology and corporate strategy: a research perspective," *MIS quarterly*, str. 107–119, 1986.

[8] S. Balaji in M. S. Murugaiyan, "Waterfall vs. v-model vs. agile: A comparative study on sdlc," *International Journal of Information Technology and Business Management*, zv. 2, št. 1, str. 26–30, 2012.

[9] G. Barczak, A. Griffin, in K. B. Kahn, "Perspective: Trends and drivers of success in npd practices: Results of the 2003 pdma best practices study," *Journal of Product Innovation Management*, zv. 26, št. 1, str. 3–23, 2009.

[10] K. Beck, "Embracing change with extreme programming," *Computer*, zv. 32, št. 10, str. 70–77, 1999.

[11] ——, *Extreme Programming Explained: Embrace Change*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2000.

[12] K. Beck in E. Gamma, *Extreme programming explained: embrace change*. addison-wesley professional, 2000.

[13] A. Begel in N. Nagappan, "Usage and perceptions of agile software development in an industrial context: An exploratory study," v *Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on*. IEEE, 2007, str. 255–264.

[14] T. F. Bresnahan, E. Brynjolfsson, in L. M. Hitt, "Information technology, workplace organization, and the demand for skilled labor: Firm-level evidence," *The Quarterly Journal of Economics*, zv. 117, št. 1, str. 339–376, 2002.

[15] S. Brinkkemper, "Method engineering: engineering of information systems development methods and tools," *Information and software technology*, zv. 38, št. 4, str. 275–280, 1996.

[16] J. Brooke *et al.*, "Sus-a quick and dirty usability scale," *Usability evaluation in industry*, zv. 189, št. 194, str. 4–7, 1996.

[17] T. Chow in D. B. Cao, "A survey study of critical success factors in agile software projects," *Journal of systems and software*, zv. 81, št. 6, str. 961–971, 2008.

[18] R. Čiarnienė in M. Vienažindienė, "Lean manufacturing: theory and practice," *Economics and management*, zv. 17, št. 2, str. 726–732, 2012.

[19] ——, "Lean manufacturing: theory and practice," *Economics and management*, zv. 17, št. 2, str. 726–732, 2012.

[20] L. Cocco, K. Mannaro, G. Concas, in M. Marchesi, "Simulating kanban and scrum vs. waterfall with system dynamics," v *International Conference on Agile Software Development*. Springer, 2011, str. 117–131.

[21] A. Cockburn, *Agile software development*. Addison-Wesley Boston, 2002, zv. 177.

[22] ——, *Crystal clear: a human-powered methodology for small teams*. Pearson Education, 2004.

[23] T. Cooke-Davies, "The "real" success factors on projects," *International journal of project management*, zv. 20, št. 3, str. 185–190, 2002.

[24] N. Delobbe, R. R. Haccoun, in C. Vandenberghe, "Measuring core dimensions of organizational culture: A review of research and development of a new instrument," *Unpublished manuscript, Universite catholique de Louvain, Belgium*, 2002.

[25] W. H. DeLone in E. R. McLean, "Information systems success: The quest for the dependent variable," *Information Systems Research*, zv. 3, št. 1, str. 60–95, 1992. [Online]. Available: https://doi.org/10.1287/isre.3.1.60

[26] W. H. Delone in E. R. McLean, "The delone and mclean model of information systems success: a ten-year update," *Journal of management information systems*, zv. 19, št. 4, str. 9–30, 2003.

[27] G. T. Doran, "There'sa smart way to write management's goals and objectives," *Management review*, zv. 70, št. 11, str. 35–36, 1981.

[28] T. Dybå in T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Information and software technology*, zv. 50, št. 9-10, str. 833–859, 2008.

[29] C. Ebert, P. Abrahamsson, in N. Oza, "Lean software development," *IEEE Software*, št. 5, str. 22–25, 2012.

[30] A. Groznik, A. Kovacic, B. Zoric, in D. Vicic, "E-logistics: informatization of slovenian transport logistics cluster," v *Information Technology Interfaces, 2004. 26th International Conference on.* IEEE, 2004, str. 101–106.

[31] J. C. Henderson in H. Venkatraman, "Strategic alignment: Leveraging information technology for transforming organizations," *IBM systems journal*, zv. 38, št. 2.3, str. 472–484, 1999.

[32] C. Hibbs, S. Jewett, in M. Sullivan, *The art of lean software development: a practical and incremental approach.* "O'Reilly Media, Inc.", 2009.

[33] J. Highsmith in A. Cockburn, "Agile software development: The business of innovation," *Computer*, zv. 34, št. 9, str. 120–127, 2001.

[34] T. Hovelja in D. Vavpotic, "Improving the evaluation of software development methodology adoption and its impact on enterprise performance," *Computer science and information systems*, str. 165–187, 2012.

[35] T. Hovelja, "Vpliv organizacije na ucinkovito uporabo informacijske tehnologije v podjetju," Ph.D. dissertation, Ekonomska fakulteta, Univerza v Ljubljani, 2006.

[36] Y. Hui, Y. Yan, W. Quanyu, in C. Zhiwen, "Compare essential unified process (essup) with rational unified process (rup)," v *Industrial Electro-*

nics and Applications (ICIEA), 2015 IEEE 10th Conference on. IEEE, 2015, str. 472–476.

[37] M. Ikonen in P. Abrahamsson, "Anticipating success of a business-critical software project: A comparative case study of waterfall and agile approaches," v *International Conference of Software Business.* Springer, 2010, str. 187–192.

[38] S. Jarzombek, "The 5th annual joint aerospace weapons systems support," v *Sensors, and Simulation Symposium (JAWS S3), Proceedings*, 1999.

[39] G. M. Kapitsaki in M. Christou, "Where is scrum in the current agile world?" v *2014 International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE).* IEEE, 2014, str. 1–8.

[40] S. Khramtchenko, "Comparing extreme programming and feature driven development in academic and regulated environments," *Feature Driven Development*, 2004.

[41] D.-Y. Kim, V. Kumar, in U. Kumar, "Relationship between quality management practices and innovation," *Journal of operations management*, zv. 30, št. 4, str. 295–315, 2012.

[42] J. F. Krafcik, "Triumph of the lean production system," *MIT Sloan Management Review*, zv. 30, št. 1, str. 41, 1988.

[43] P. Kruchten, *The rational unified process: an introduction.* Addison-Wesley Professional, 2004.

[44] C. Larman in V. R. Basili, "Iterative and incremental developments. a brief history," *Computer*, zv. 36, št. 6, str. 47–56, 2003.

[45] L. Layman, L. Williams, D. Damian, in H. Bures, "Essential communication practices for extreme programming in a global software deve-

lopment team," *Information and software technology*, zv. 48, št. 9, str. 781–794, 2006.

[46] M. Mahalakshmi in M. Sundararajan, "Traditional sdlc vs scrum methodology–a comparative study," *International Journal of Emerging Technology and Advanced Engineering*, zv. 3, št. 6, str. 192–196, 2013.

[47] P. McBreen, *Software craftsmanship: The new imperative*. Addison-Wesley Professional, 2002.

[48] M. Mihelcic, *Organizacija in ravnateljevanje*. Fakulteta za racunalnistvo in informatiko, Univerza v Ljubljani, 1999.

[49] H. R. Nemati in C. D. Barko, "Key factors for achieving organizational data-mining success," *Industrial Management & Data Systems*, zv. 103, št. 4, str. 282–292, 2003.

[50] T. Ohno, *Toyota production system: beyond large-scale production*. crc Press, 1988.

[51] J. A. Osorio, M. R. Chaudron, in W. Heijstek, "Moving from waterfall to iterative development: An empirical evaluation of advantages, disadvantages and risks of rup," v *Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on*. IEEE, 2011, str. 453–460.

[52] F. Paetsch, A. Eberlein, in F. Maurer, "Requirements engineering and agile software development," v *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on*. IEEE, 2003, str. 308–313.

[53] S. R. Palmer in M. Felsing, *A practical guide to feature-driven development*. Pearson Education, 2001.

[54] S. Petter, W. DeLone, in E. McLean, "Measuring information systems success: models, dimensions, measures, and interrelationships," *European journal of information systems*, zv. 17, št. 3, str. 236–263, 2008.

[55] M. Poppendieck in T. Poppendieck, *Lean Software Development: An Agile Toolkit: An Agile Toolkit*. Addison-Wesley, 2003.

[56] R. U. Process, "Rational software corporation," *Cupertino, Ca*, 1999.

[57] ——, "Best practices for software development teams," *A Rational Software Corporation White Paper. Recuperado de: https://www. ibm. com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP0 pdf*, 2001.

[58] K. N. Rao, G. K. Naidu, in P. Chakka, "A study of the agile software development methods, applicability and implications in industry," *International Journal of Software Engineering and its applications*, zv. 5, št. 2, str. 35–45, 2011.

[59] T. Ravichandran in A. Rai, "Quality management in systems development: an organizational system perspective," *MIS quarterly*, str. 381–415, 2000.

[60] K. H. Rose, "A guide to the project management body of knowledge (pmbok® guide) fifth edition," *Project management journal*, zv. 44, št. 3, 2013.

[61] W. W. Royce, "Managing the development of large software systems.[online] http://www. cs. umd. edu/class/spring2003/cmsc838p," *Process/waterfall. pdf*, 1970.

[62] K. Schwaber, "Scrum development process. oopsla'95 workshop on business object design and implementation," *Austin, USA*, 1995.

[63] ——, "Scrum development process," v *Business object design and implementation*. Springer, 1997, str. 117–134.

[64] K. Schwalbe, *Information technology project management*. Cengage Learning, 2015.

[65] O. Serrat, "The five whys technique," v *Knowledge solutions*. Springer, 2017, str. 307–310.

[66] A. Shalloway, G. Beaver, in J. R. Trott, *Lean-agile software development: achieving enterprise agility*. Pearson Education, 2009.

[67] M. S. Silver, M. L. Markus, in C. M. Beath, "The information technology interaction model: A foundation for the mba core course," *MIS Quarterly*, zv. 19, št. 3, str. 361–390, 1995. [Online]. Available: http://www.jstor.org/stable/249600

[68] I. Sommerville, "Software engineering. international computer science series," *ed: Addison Wesley*, 2004.

[69] J. Sutherland, M. Pope, in K. Rugg, "The hybrid object-relational architecture (hora): an integration of object-oriented and relational technology," v *Proceedings of the 1993 ACM/SIGAPP symposium on Applied computing: states of the art and practice*. ACM, 1993, str. 326–333.

[70] H. Takeuchi in I. Nonaka, "The new new product development game," *Harvard business review*, zv. 64, št. 1, str. 137–146, 1986.

[71] R. TP026B, "Rational unified process," *URL: https://www. ibm. com/developerworks/rational/library/content/03July/10 00/1251/1251_bestpractices_TP026B. pdf Cited October*, zv. 18, 2017.

[72] P. Trinidad, D. Benavides, A. Durán, A. Ruiz-Cortés, in M. Toro, "Automated error analysis for the agilization of feature modeling," *Journal of Systems and Software*, zv. 81, št. 6, str. 883–896, 2008.

[73] D. Vavpotic in M. Bajec, "An approach for concurrent evaluation of technical and social aspects of software development methodologies," *Information and software Technology*, zv. 51, št. 2, str. 528–545, 2009.

[74] X. Wang, K. Conboy, in O. Cawley, ""leagile" software development: An experience report analysis of the application of lean approaches in agile software development," *Journal of Systems and Software*, zv. 85, št. 6, str. 1287–1299, 2012.

[75] R. K. Yin, *Case study research: Design and methods*. Sage publications, 2013.