Korištenje agilnih metoda vođenja projekata u razvoju IT usluga

Maran, Mia

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Mia Maran

KORIŠTENJE AGILNIH METODA VOĐENJA PROJEKATA U RAZVOJU IT USLUGA

ZAVRŠNI RAD

Varaždin, 2022.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Mia Maran

JMBAG: 0016142651

Studij: Poslovni sustavi

KORIŠTENJE AGILNIH METODA VOĐENJA PROJEKATA U RAZVOJU IT USLUGA

ZAVRŠNI RAD

Mentor/Mentorica:

Doc. dr. sc. Katarina Pažur Aničić

Varaždin, rujan 2022.

Sadržaj

1. Uvod

Sa sigurnošću možemo reći kako je industrija informacijske tehnologije (u nastavku teksta IT) jedna od najbrže rastućih industrija. Ponekad je takav nagli i brz razvoj teško pratiti pa razne IT kompanije nastoje pronaći alate, tehnike i pristupe koji će im u tome pomoći. Jedan od takvih pristupa su i agilne metode vođenja projekata koje, kako i samo ime kaže, pružaju timovima agilnost u projektima i procesu razvoja IT usluga. Agilnost im osigurava da se pravovremeno i na pravi način prilagode dinamici i zahtjevima svakodnevno promjenjivog tržišta.

Cilj ovog rada je uvidjeti kako su to agilne metode, a posebice Scrum okvir, olakšali i unaprijedili upravljanje projektima razvoja IT usluga.

U teorijskom dijelu ovog rada bit će prikazan razvoj agilnih metoda iz vrlo popularnih tradicionalnih metoda, biti će obrađene posebnosti koje sa sobom nose agilne metode te će detaljno biti proučeni koncepti vezani za jednu od najpoznatijih i najčešće korištenih agilnih metoda – Scrum okvir.

Nakon detaljne obrade Scrum okvira u teorijskom dijelu, u praktičnom dijelu će kroz studiju slučaja biti prikazano koliko se teorija Scrum okvira uklapa u stvarne događaje i svakodnevno poslovanje jedne hrvatske IT kompanije koja ju koristi. Cilj istraživanja je uvidjeti kako se i u kojoj mjeri teorija Scrum okvira zaista primjenjuje u praksi te koliko su Scrum, ali i ostale agilne metode praktične za korištenje u vođenju projekata izrade softverskih rješenja.

Osim studije slučaja, u praktičnom dijelu, također je ukratko prikazano kako pripremiti jedan IT projekt koristeći Scrum okvir te poznati Jira softver.

Na kraju ovog rada zaokružit ćemo priču zaključkom o tome kako se teorija Scrum okvira i agilnih metodologija prenosi u praksu, a pogotovo u slučaju kada se radi o agenciji koja ne proizvodi vlastiti proizvod već proizvod za klijente.

2. Projektni menadžment

Kako bismo uopće mogli pričati o projektnom menadžmentu, važno je na početku objasniti što je to projekt te koji su njegovi osnovni koncepti. Prema Project Management Body of Knowledge (u daljnjem tekstu PMBOK) i kao što prenosi Vodič kroz znanje o upravljanju projektima, projekt je „privremeni pothvat kojim se stvara jedinstven proizvod, usluga ili rezultat.“ (PMI, 2011., str. 5). U današnjim organizacijama projekti se definiraju kao složeni poslovni poduhvati koji su usmjereni konačnim ciljevima (Andrejić, Đorović, Pamučar, 2011., str 142).

Jedno od glavnih obilježja kod pokretanja projekata je stvaranje takoreći prividne organizacije, unutar već postojeće organizacije. Prividnu organizaciju čini grupa stručnjaka koji zajedničkim naporima rješavaju određeni problem te nastoje postići postavljeni cilj projekta. Projektni pristup koristi se u raznim djelatnostima, u svim vrstama poduzeća te za rješavanje raznih problema i poslovnih izazova. Glavne karakteristike projekta su da projekti imaju ograničeno trajanje i ograničene resurse (pogotovo novčane i ljudske) te da su vođeni unaprijed postavljenim ciljevima i planom aktivnosti kako postići taj cilj (Salameh, 2014., str. 53).

Kako bi se osigurao kontinuirani napredak projekta te u konačnici postizanje zadanih projektnih ciljeva, u drugoj polovici 20. stoljeća razvila se disciplina projektnog menadžmenta. Počeci projektnog menadžmenta vežu se uz projekt Manhattan kojim je američka vojska razvijala atomsku bombu tokom Drugog svjetskog rata (Majstorović, Majstorović, 2019., str. 4).

Kako definira Kerzner, a u svom radu prenose Andrejić, Đorović i Pamučar (2011., str.144.) projektni menadžment je planiranje, organiziranje, usmjeravanje i kontrola resursa poduzeća u svrhu postizanja specifičnih ciljeva definiranih za pojedini projekt. Osim toga prema udruzi voditelja projekata (Project Management Institute u daljnjem tekstu PMI), suština upravljanja projektima jest primjena znanja, vještina, alata i tehnika na projektne aktivnosti kako bi se zadovoljila očekivanja i potrebe zainteresiranih strana (Salameh, 2014., str. 53).

U sklopu projektnog menadžmenta postoje određena područja znanja upravljanja projektima, a ona se koriste kako bi se opisali dijelovi projekta kojima je potrebno upravljati. Za uspješno upravljanje projektom zaduženi su projektni menadžeri koji moraju ova područja imati pod stalnim nadzorom, a za to koriste mnoge alate i resurse koji su im na raspolaganju. PMBOK opisuje 10 područja znanja upravljanja projektima, a to su (Fertalj, Car, Nižetić Kosović, 2016., str. 5):

1. Upravljanje integracijom (engl. Integration Management) – uključuje različite procese i metodologije upravljanja projektima u svrhu bolje produktivnosti i sinkronizacije rada unutar timova
2. Upravljanje djelokrugom (engl. Scope Management) – identifikacija projektnih zadataka i ciljeva kroz procese prikupljanja zahtjeva članova, stvaranje strukturne raščlambe posla (engl. Work breakdown structure (WBS)) te praćenje i upravljanje promjenama
3. Upravljanje rasporedom (engl. Schedule Management) – dio je faze planiranja projekta, a cilj je postaviti realistični vremenski okvir za postizanje ciljeva pojedinog projekta
4. Upravljanje troškovima (engl. Cost Management) - planiranje i kontrola troškova vezanih uz projekt, a podrazumijeva analizu, izvješćivanje, predviđanje i praćenje troškova za vrijeme trajanja projekta
5. Upravljanje kvalitetom (engl. Quailty Management) – nadzor nad svim aktivnostima vezanim za stvaranje rezultata projekta kako bi se osigurala najviša moguća kvaliteta
6. Upravljanje resursima (engl. Resource Management) – cilj je pravilnom preraspodjelom izvući najviše moguće iz ljudskih, novčanih i materijalnih resursa potrebnih za izvođenje projekta
7. Upravljanje komunikacijom (engl. Comunication Management) – stvaranje kanala i poticanje učestale razmjene pravovremenih, istinitih i razumljivih informacija
8. Upravljanje rizicima (engl. Risk Management) – cilj je identificirati, evaluirati i spriječiti ili ublažiti negativne, ali i pozitivne rizike na projektu
9. Upravljanje nabavom (engl. Procurement Management) – izgradnja i održavanje odnosa s vanjskim resursima potrebnim za ostvarenje ciljeva projekta
10. Upravljanje sudionicima (engl. Stakeholder Management) – identificiranje dionika projekta i njihovih očekivanja te razvoj strategije za upravljanje dionicima

2.1. Životni ciklus projekta

Možemo reći da sve na ovome svijetu ima svoj životni ciklus – od ljudi do proizvoda – pa tako i projekti. Životni ciklus projekta sastoji se od 5 faza: inicijacija, planiranje, provedba, kontrola i praćenje te zatvaranje projekta.

Prva faza jest inicijacija projekta. Najvažniji cilj ove faze je dokazati da projekt ima vrijednost te da je izvediv. Za to je zadužen voditelj projekta koji raspisuje projektnu dokumentaciju, pri čemu je najvažnija povelja projekta. U povelji, voditelj projekta opisuje

poslovnu viziju i misiju, ciljeve i koristi projekta, popis potrebnih dionika, opseg projekta, rizike i proračun projekta te time nastoji dokazati da je projekt izvediv, isplativ te da ima svrhu (Brown, 2021.). Osim izrade dokumentacije u ovoj se fazi određuje veličina projektnog tima, utvrđuje obujam projekta i njegove faze te okvirno planiraju resursi (kao što su npr. računala, uredski prostor, komunikacijska oprema i slično) potrebni za uspješnu provedbu projekta (Rožman,2021., str. 12).

Nakon što voditelj projekta „dobije zeleno svijetlo“ za provedbu projekta prelazi se u sljedeću fazu životnog ciklusa, a to je planiranje. Cilj je izraditi plan projekta koji definira kako će se upravljati svakim područjem projekta (raspored, troškovi, opseg, rizici, resursi, dionici) (Quiambao, 2022.). Možemo reći kako je jedan od najvažnijih rezultata ove faze dobar, detaljan i opsežan plan koji će voditi njegove dionike kroz posljednje 2 faze životnog ciklusa projekta.

Nakon pripreme kvalitetnog plana moguć je prelazak u fazu provedbe projekta. U ovoj fazi rješavaju se zadaci navedeni u projektnom planu u svrhu postizanja projektnih ciljeva. Paralelno s ovom fazom izvršava se i faza kontrole i praćenja projekta. Zadatak projektnog menadžera u ovoj fazi je da nadgleda sve projektne aktivnosti kako bi uvidio ide li sve kako je planirano. Ako to nije slučaj, projektni menadžer mora reagirati što brže kako bi otklonio nastale poteškoće tj. odstupanja od plana. Kako bi mu u tome pomogli, ostali članovi projektnog tima moraju dobro upravljati zadacima te izvještavati projektnog menadžera o statusu zadataka kako bi se osiguralo da se projekt izvršava unutar unaprijed određenog vremenskog okvira (Rožman, 2021., str. 16). Osim vremenskog rasporeda potrebno je nadzirati i ostala područja projekta kao što su troškovi, mogući rizici i dostupnost resursa.

Posljednja faza jest faza zatvaranja projekta. U ovoj se fazi zainteresiranim stranama prezentiraju konačni rezultati projekta. Ako su zainteresirane strane zadovoljne s ishodima, resursi se raspuštaju te se potpisuje dokumentacija (Rožman, 2021., str. 16).

2.2. Upravljanje projektima u razvoju IT usluga

Razvojem tehnologije, radi povećanja efikasnosti pojavila se potreba za razvojem IT usluga.

Prema Gartner Glossary-u (bez dat.) „IT usluga odnosi se na primjenu poslovne i tehničke ekspertize kako bi se organizacijama omogućilo stvaranje, upravljanje i optimizacija ili pristup informacijama i poslovnim procesima“.

Primarna svrha IT usluga je pomoć tvrtkama da posluju efikasnije i efektivnije, a neke od najpoznatijih IT usluga su (Indeed, 2021.):

- Usluge računalstva u oblaku (engl. Cloud services)
- Softver kao usluga (engl. Software as a service (SaSS))
- Razvoj softvera (engl. Software development as a Service (SDaaS))
- Zaštita mreže (engl. Network Security)
- Rješavanje problema i tehnička podrška (engl. Troubleshooting and technical support)
- Instalacija i održavanje hardware-a (engl. Hardware installations and maintenance)

S obzirom da su glavna tema ovog rada agilne metode vođenja projekta koje su u praksi većinski korištene za upravljanje projektima razvoja softvera u ovom će se radu pod pojmom IT usluge smatrati usluge razvoja softvera.

U ovom kontekstu softverom odnosno softverskim proizvodom smatramo: „Svaki proizvod, uslugu ili dodatak koji se oslanja na softver kao temeljnu komponentu svoje vrijednosne ponude. Glavni dio te vrijednosne ponude je da je softverski proizvod (ili bi trebao biti) dizajniran za rješavanje stvarnog problema za stvarne ljude ili poduzeća.“ (3 Pillar Global, 2021.).

Kod razvoja bilo koje IT usluge, u ovom slučaju kod izrade IT proizvoda - softvera, treba uzeti u obzir tri cilja: pružiti pravi proizvod, dobro upravljati projektom razvoja te napraviti proizvod na pravilan način (Čubranić, Kaluža, Novak. 2013., str. 242). Pravi proizvod jest upravo onaj kojeg je klijent odnosno naručitelj naručio i koji očekuje da će dobiti. Taj proizvod mora zadovoljiti potrebe klijenata i rješavati problem zbog kojeg su ga zatražili. Da bi se projekt razvoja uspješno završio potrebno je njime dobro upravljati, a to znači da je potrebno usvojiti dovoljno procesa i praksi tako da se kvalitetno organizira rad svih uključenih u projekt. Posljednje i možemo reći najvažnije, jest izrada proizvoda na pravi način. Kroz cijeli projekt potrebno je provoditi konstantne provjere usklađenosti svrhe i dizajna proizvoda u odnosu na zahtjeve klijenta.

Kako bi se zadovoljili navedeni ciljevi, organizacije koje se bave razvojem i pružanjem IT usluga teže pronaći metode projektnog menadžmenta kojima mogu osigurati čestu komunikaciju s klijentima i njihovo uključenje u proces razvoja, pojednostavljene poslovne procese i što bržu isporuku u svrhu smanjenja rizika za organizaciju.

Kako navodi Wysocki, a prenosi Radoš (2021.,str. 17): „Upravljanje projektima razvoja softvera disciplina je ocjenjivanja karakteristika softvera, odabir najboljeg životnog ciklusa za razvoj softvera, a zatim odabir odgovarajućeg pristupa upravljanju projektom kako bi se osiguralo

zadovoljenje potreba kupaca za isporuku poslovne vrijednosti, što je moguće učinkovitije.“ U ovoj definiciji vidimo zadovoljenje sva tri prethodno navedena cilja za razvoj kvalitetnog softvera.

Kada govorimo o razvoju softverskih proizvoda možemo primijetiti da između tradicionalnih projekata kao što su građevinarski i slični projekti i projekata razvoja softverskih proizvoda postoji nekoliko razlika zbog kojih takvi projekti zahtijevaju posebne vrste upravljanja projektom (Radoš, 2021., str. 17).

Osnovna razlika je u specifičnostima softverskih proizvoda. Svaki softverski proizvod ima svoju djelotvornost. Pod tim pojmom podrazumijeva se postizanje zadovoljavanja potreba korisnika tokom korištenja softverskog proizvoda. Isto tako softverski proizvodi se vremenom i uporabom ne troše tako lako poput drugih „klasičnih“ proizvoda, ali ih je potrebno dugoročno održavati i konstantno poboljšavati. Isto tako u obzir treba uzeti da su u tvrtkama koje proizvode softver najvažniji resursi ljudi i njihova znanja i vještine, dok su kod drugih proizvoda to većinom novac i materijalni resursi kao proizvodni pogoni i slično (Bevanda, Sinković, 2007., str. 3).

Upravo te specifičnosti zahtijevaju primjenu posebnih oblika upravljanja projektima kako bi se omogućilo efikasno rješavanje i izbjegavanje prepreka koje one uzrokuju.

Radoš (2021., str. 17) također navodi kako je jedna od glavnih razlika između tradicionalnih projekata i projekata razvoja softvera u tome što softver ima jedinstveni razvojni ciklus koji zahtjeva više iteracija testiranja te konstantne povratne informacije od korisnika, što nije moguće izvesti u tradicionalnom životnom ciklusu projekta**.**

Osnovne faze razvojnog ciklusa softvera su („What is the Software Development Life Cycle (SDLC)“, 2013.):

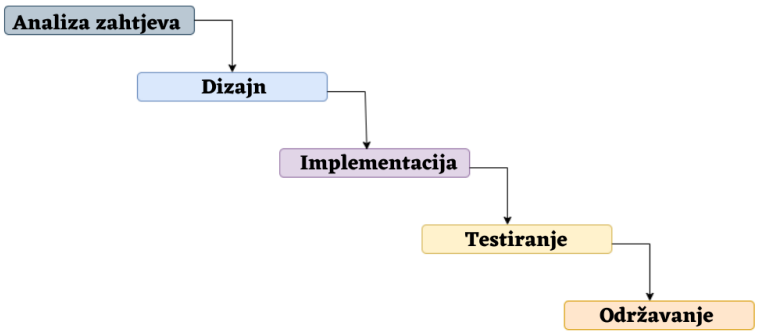
1. Prikupljanje i analiza zahtjeva – faza u kojoj se detaljno razrađuju i dokumentiraju zahtjevi
2. Dizajn – detaljno se raspisuju tehnički zahtjevi (potrebne baze podataka, sigurnosni procesi, zahtjevi za hardver i softver). Osim toga, u ovoj se fazi provode analize rizika i pišu funkcionalne i nefunkcionalne specifikacije
3. Implementacija – u ovoj fazi započinje se s pisanjem koda te se provode jedinični testovi
4. Testiranje – nakon što se softver migrira u testno okruženje provode se razni testovi kao što su integracijski testovi te testovi prihvaćanja korisnika

5. Uvođenje i održavanje – obuka korisnika o softveru te održavanje softvera nakon njegove primjene u stvarnim sustavima.

2.3. Metode upravljanja projektima razvoja softvera

Postoji nekoliko metoda koje se mogu koristiti u razvojnom ciklusu softvera. U ovom će poglavlju biti ukratko objašnjena najčešće korištena tradicionalna metoda - vodopadna, dok će sljedeće poglavlje biti posvećeno agilnim metodologijama.

Za softvere u kojima su svrha, ciljevi i koraci kako ih postići unaprijed detaljno i u potpunosti definirani idealan je vodopadni model razvoja softvera. To je tradicionalni linearni oblik modela koji se temelji na sekvencijalnom procesu što znači da sljedeća faza razvojnog ciklusa softvera može započeti tek nakon što je prethodna završila.



Slika 1: Vodopadni model (izrada autora prema Marsden, Minder, Parker, 2008.)

Svaka se od faza prikazanih na slici 1 izvodi samo jednom te se tek nastankom formalnog dokumenta i po samom završetku faze može preći u sljedeću fazu (Palmquist, Lapham, Gracia-Miller, Chick, 2013., str. 5). Ako uzmemo primjer sa slike 1, razvoj bi išao tako da bi početna faza bila faza analize zahtjeva. Izlaz iz tog procesa bio bi dokument koji sadržava popis zahtjeva.

Nakon što je dokument službeno potvrđen može se preći u fazu dizajna i tako sve do kraja modela.

Osim isključivo sekvencijalnog toka ovaj model obilježava i mala uključenost klijenta i potreba za opsežnom dokumentacijom u svakom koraku (Palmquist, Lapham, Gracia-Miller, Chick, 2013., str 5.). Mala uključenost klijenta odnosi se na činjenicu da je klijent najčešće uključen samo u fazi postavljanja zahtjeva te fazama ispitivanja i uvođenja gotovog proizvoda, dok u fazama dizajna i implementacije nema nikakve uloge.

Prednosti ovog modela su to što je lako razumljiv, jasno definira očekivane rezultate i ključne točke u projektu, naglašava važnost analize prije dizajna i dizajna prije implementacije, nema financijskih iznenađenja i testiranje je olakšano.

Isto tako postoje i određeni nedostatci, a to su činjenica da ovaj model često zahtjeva duže vrijeme za ostvarenje, klijent ne vidi proizvod do samog kraja projekta te da ovaj model jednostavno nije dizajniran za lako i jeftino prihvaćanje promjena nastalih u sredini toka odnosno nije fleksibilan (Kientiz, 2017.). Na primjer, ako bi došlo do promjena u toku faze implementacije projekt bi morao biti vraćen na početak odnosno u fazu analize zahtjeva. Isto tako, s obzirom da se testiranje provodi tek nakon potpunog dovršetka implementacije, otkrivene greške je teže i skuplje ispraviti.

Kroz godine, navedeni nedostatci vodopadnog modela nastojali su se smanjiti ili u potpunosti otkloniti razvojem raznih drugih „tradicionalnih“ modela. Konačno 2001. razvijene su agilne metode koje u potpunosti rješavaju navedene nedostatke. Agilne metode svojim su potpuno novim pristupom razvojnom procesu uvele svježinu na tržište te olakšale poslovanje milijunima tvrtki u posljednjih 20 godina. Više o agilnim metodama, njihovom nastanku i razvoju slijedi u nastavku.

3. Agilne metode upravljanja IT projektima

Sama riječ „agilno“ objašnjava glavnu bit agilnih metoda, a to je fleksibilnosti i lako prilagođavanje ubrzanom tržištu.

Prema Agile Alliance-u: „Agilnost je sposobnost stvaranja i reagiranja na promjene. To je način suočavanja s neizvjesnim i turbulentnim okruženjem i konačnog uspjeha u njemu.“ (Agile Alliance, bez dat.).

U kontekstu razvoja softvera agilne metode su „iterativni i inkrementalni (evolucijski) pristup razvoju softvera koji se izvodi na visoko suradnički način od strane autonomnih timova unutar učinkovitog upravljačkog okvira uz ceremoniju „taman dovoljno“ koja proizvodi visokokvalitetan softver na isplativ i pravodoban način koji zadovoljava promjenjive potrebe svojih dionika.“ (Lapham, Williams, Hammons, Burton, Schenker, 2010., str 5).

Ako ujedinimo ove dvije definicije vidljivo je kako su glavni ciljevi agilnosti i agilnih metoda prilagodba promjenama, visoka suradnja i postojanje okvira unutar kojeg posluju autonomni timovi sa svrhom proizvodnje softverskih proizvoda najveće kvalitete čija je osnovna svrha zadovoljavanje potreba i rješavanje problema krajnjih korisnika.

3.1. Povijest agilnih metoda

Povijest agilnih metoda seže u veljaču 2001. godine kada se 17 mladih razvojnih inženjera okupilo na skijanju u planinama Wasatch u Utahu. Oni su razgovorom utvrdili kako imaju jedan zajednički problem, a to je da su tvrtke u kojima su bili zaposleni bile previše usmjerene na pretjerano planiranje i dokumentiranje ciklusa razvoja softvera što je posljedično izazvalo da se izgubi iz vida ono što je zaista cilj svakog razvojnog ciklusa – udovoljiti klijentima. (Drumond, bez dat.)

Nazvavši se "Agilni savez" (engl. Agile Alliance), ova skupina mislilaca o razvoju softvera daljnjom su razradom navedenog problema došli do zaključka da im je potrebna reforma poslovnih procesa. Glavne ideje o tome što i kako treba promijeniti zapisali su, potpisali i tako je nastao poznati Agilni manifest za razvoj softvera (engl. Agile Manifesto).

Agilni manifest za razvoj softvera sastoji se od četiri vrijednosti i dvanaest načela. Manifest započinje rečenicom: „Tražimo bolje načine razvoja softvera razvijajući softver i pomažući drugima pri njegovom razvoju.“ („Manifesto for Agile Software Development“, 2001.)

Četiri vrijednosti razvoja agilnog softvera su („Manifesto for Agile Software Development“, 2001.):

„Više cijenimo:“

1. „Ljude i njihove međusobne odnose nego procese i alate“
2. „Upotrebljiv softver nego iscrpnu dokumentaciju“
3. „Suradnju s naručiteljem nego pregovaranje oko ugovora“
4. „Reagiranje na promjenu nego ustrajanje na planu.“

Dvanaest načela koja proizlaze ih navedenih temeljnih vrijednosti su sljedeće („Manifesto for Agile Software Development“, 2001.):

1. „Najvažnije nam je zadovoljstvo naručitelja koje postižemo ranom i neprekinutom isporukom softvera koji nosi vrijednost.“
2. „Spremno prihvaćamo promjene zahtjeva, čak i u kasnoj fazi razvoja. Agilni procesi koriste promjene kako bi naručitelju stvorili kompetitivnu prednost.“
3. „Često isporučujemo upotrebljiv softver, u razmacima od nekoliko tjedana do nekoliko mjeseci, nastojeći da razmak bude čim kraći.“
4. „Poslovni ljudi i razvojni inženjeri moraju svakodnevno zajedno raditi, tijekom cjelokupnog trajanja projekta.“
5. „Projekte ostvarujemo oslanjajući se na motivirane pojedince. Pružamo im okruženje i podršku koja im je potrebna, i prepuštamo im posao s povjerenjem.“
6. „Razgovor uživo je najučinkovitiji način prijenosa informacija razvojnom timu i unutar tima.“
7. „Upotrebljiv softver je osnovno mjerilo napretka.“
8. „Agilni procesi potiču i podržavaju održivi razvoj. Pokrovitelji, razvojni inženjeri i korisnici trebali bi moći neograničeno dugo zadržati jednak tempo rada.“
9. „Neprekinuti naglasak na tehničkoj izvrsnosti i dobar dizajn pospješuju agilnost.“
10. „Jednostavnost – vještina povećanja količine posla kojeg ne treba raditi – je od suštinske važnosti."
11. „Najbolje arhitekture, projektne zahtjeve i dizajn, stvaraju samo–organizirajući timovi.“

12. „Tim u redovitim razmacima razmatra načine da postane učinkovitiji, zatim usklađuje i prilagođava svoje ponašanje.“

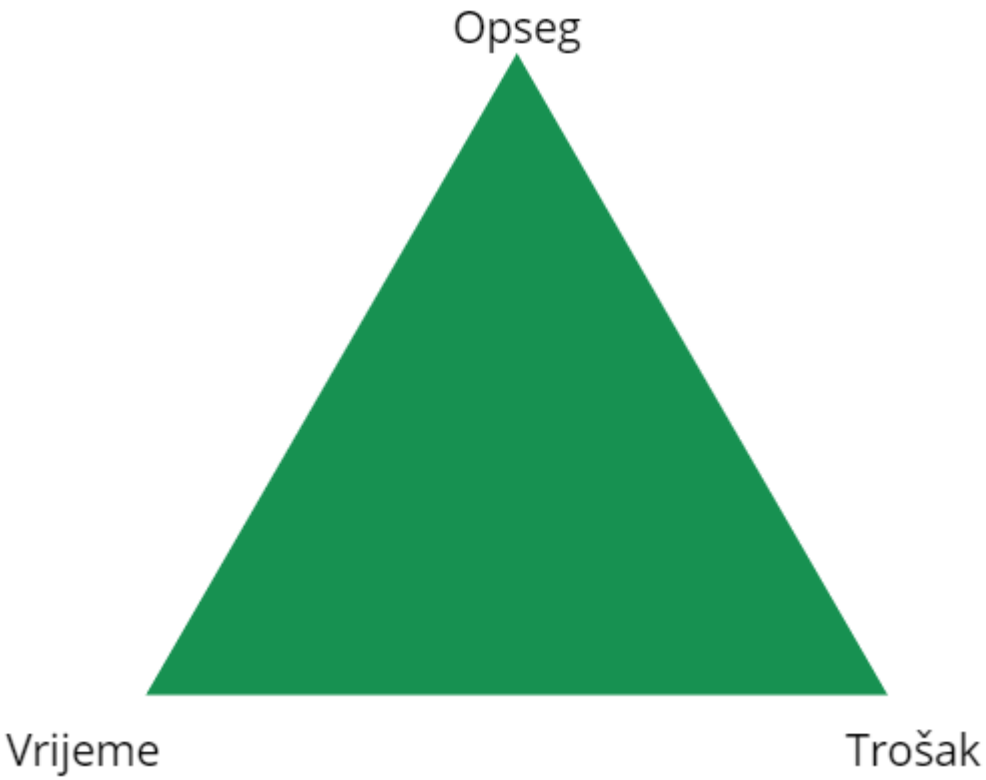
Na temelju ovih dvanaest načela i četiri vrijednosti kroz godine su se razvili koncepti koji danas obilježavaju agilne metode. Prema Agile Alliance-u (bez dat.) neki od najvažnijih agilnih koncepata su:

- Korisničke priče – razdvajanje ukupnog posla projekta na manje cjeline opisane iz korisničkog pogleda. Svaka korisnička priča nosi vrijednost koja doprinosi kvaliteti cjelokupnog projekta
- Dnevni sastanci – kako bi se osigurala koordiniranost tima te u svrhu lakšeg rješavanja prepreka i problema nastalih u timu, svaki dan se održavaju kratki „sastanci s nogu“
- Tim – grupa stručnjaka koja se obično sastoji od 5-8 članova koji kombinacijom vlastitih znanja i vještina rješavaju složene zadatke s ciljem zadovoljenja korisničkih zahtjeva
- Inkrementalni razvoj – svaki inkrement čini upotrebljivi dio softvera koji povećava vrijednost krajnjeg proizvoda
- Iterativni razvoj - omogućava timu da se vraća u prethodne faze projekta u slučaju pojave novih zahtjeva ili promjena
- Retrospektiva - omogućava članovima tima konstantno unapređenje procesa, odnosa unutar tima, ali i njihovih vlastitih znanja i kompetencija

Osim navedenih, jedan od najvažnijih elemenata agilnih metoda je agilni način razmišljanja. S obzirom da prava definicija agilnog razmišljanja ne postoji, provedena su različita istraživanja kojim se nastojalo utvrditi glavne karakteristike agilnog razmišljanja. Tako je u istraživanju kojeg su 2020. proveli Mordi i Schoop izdvojeno preko 20 karakteristika agilnog razmišljanja, a one najčešće spomenute su povjerenje, odgovornost, vlasništvo, kontinuirano poboljšanje i spremnost na učenje (Mordi, Schoop, 2020., str. 8).

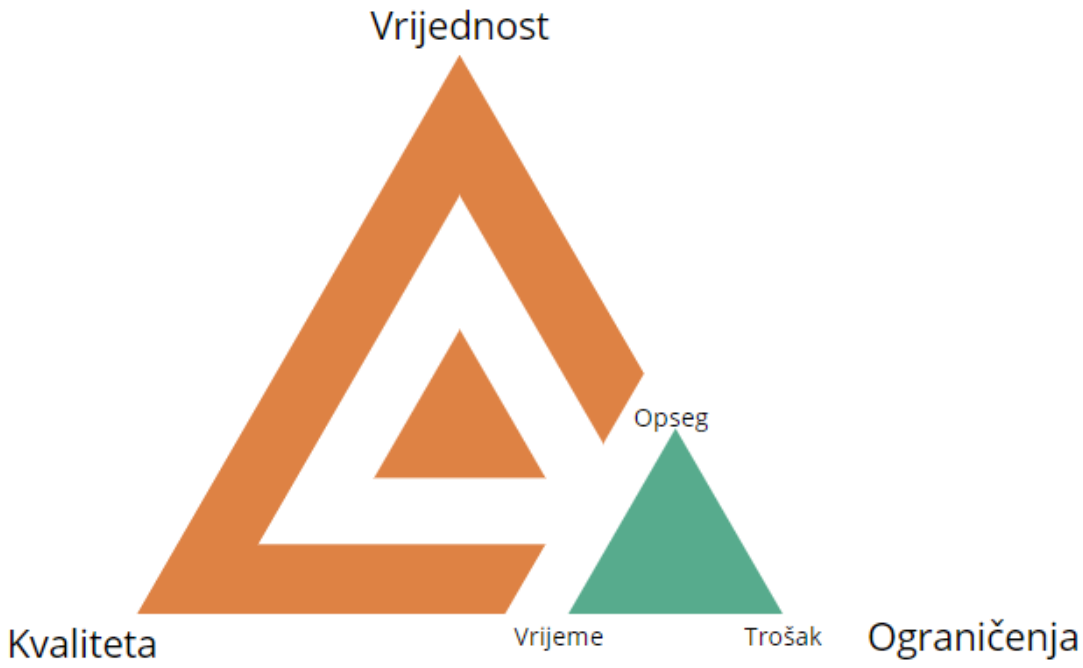
3.2. Agilno naspram tradicionalno

Jedan od glavnih razloga nastanka agilnih metoda bili su razni nedostatci do tada korištenih tradicionalnih metoda. Agilne su metode, na neki način, nadogradnja na tradicionalne metode, a to se najbolje vidi na usporedbi „željeznog“ (engl. Iron Triangle) i agilnog trokuta.



Slika 2: Željezni trokut (izrada autora prema Salazar, Caraballo, 2018.)

„Željezni“ ili projektni trokut (engl. Project Management Triangle), korišten u tradicionalnim metodama, je prikaz osnovnih kriterija pomoću kojih mjerimo uspjeh projekta, a to su vrijeme, trošak i opseg (Adler, Helm, Pollack, 2018.). Trokut se temelji na logici da se niti jedan od navedenih kriterija ne može promijeniti bez promjene preostalih kriterija. Na primjer, ako se tokom razvoja promijeni opseg projekta automatski će biti potrebno produljiti ili skratiti trajanje projekta i/ili povećati ili smanjiti troškove.



Slika 3: Agilni trokut (izrada autora prema Salazar, Caraballo, 2018.)

S druge strane, kriteriji agilnog trokuta su: vrijednost, kvaliteta i ograničenja koja čine vrijeme, trošak i opseg (Highsmith, 2009.). Vrijednost se odnosi na vrijednost koja se pruža korisnicima i glavni cilj vezan za taj kriterij je proizvesti proizvod koji se može koristiti. Kvaliteta se odnosi na kvalitetu samog softvera, a cilj je izraditi povjerljiv i lako prilagodiv softver. Ograničenja se odnose na već spomenuta ograničenja koja čine dio željeznog trokuta te je u tom pogledu cilj postići prethodno navedene kriterije (kvalitetu i vrijednost) unutar zadanih ograničenja (vremena, troška i opsega) (Highsmith, 2009.).

Iako veoma ovisne jedna o drugoj, agilne i tradicionalne metode imaju i određene razlike prikazane u tablici 1.

Tablica 1: Razlike agilnih i tradicionalnih metoda

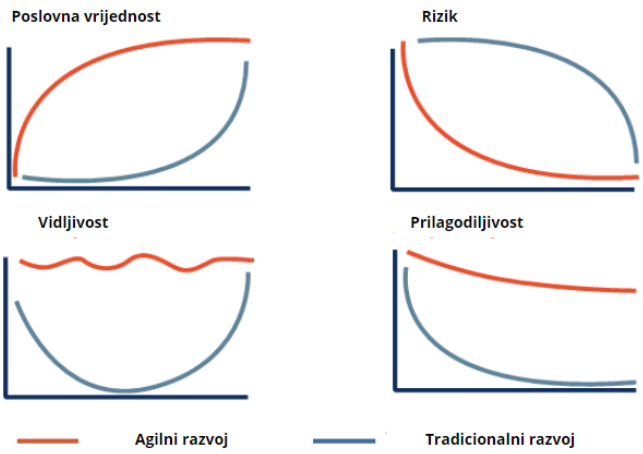
ZNAČAJKA	TRADICIONALNE METODE	AGILNE METODE
Zahtjevi korisnika	Jasno definirani na početku projekta	Nisu jasno definirani, nema detaljne vizije
Veličina tima	Veći timovi	Manji timovi
Uključenost naručitelja	Mala (na početku i kod isporuke)	Velika (tijekom cijelog projekta)
Trajanje projekta	Dugo	Razno
Upravljanje	Komande i kontrola	Vodstvo i suradnja
Komunikacija	Formalna	Neformalna
Razvojni model	Model životnog ciklusa	Evolucijski, iterativni
Količina rizika i promjena	Minimalna	Minimalan rizik, velika mogućnost promjena
Veličina projekta	Veliki projekti	Mali i srednji projekti
Troškovi ponovnog započinjanja	Visoki	Niski
Testiranje	Po završetku kompletnog koda	U svakoj iteraciji

(izrada autora prema Vresk, Pihir i Tomičić Furjan, 2020; Carr, 2022)

Usporedbu agilnog i tradicionalng razvoja možemo vidjeti i na slici 4.

Poslovna vrijednost kod agilnih metoda raste tokom razvoja zahvaljujući redovitim isporukama dijelova krajnjeg proizvoda. S druge strane, kod tradicionalnog razvoja poslovna se vrijednost očituje tek na kraju, kod isporuke gotovog proizvoda. Rizik je kod tradicionalnih metoda puno veći nego kod agilnih metoda. Razlog tome su mogućnosti promjene zahtjeva, pojave grešaka, nesporazum s naručiteljima i slično. Navedeni rizici, zahvaljujući razvoju i isporuci manjih dijelova softvera, puno su manji kod agilnih metoda. Isto tako česta isporuka kod agilnih metoda omogućuje naručitelju konstantnu vidljivost proizvoda. Svakom novom isporukom naručitelj može vidjeti kako dotad izrađeni proizvod radi i imati viziju kako bi to moglo izgledati na kraju. Kod tradicionalnih metoda to nije slučaj. Naručitelj gotov softver vidi tek pri isporuci. Što se tiče prilagodivljivosti, može se zaključiti kako se agilni projekti puno lakše prilagođavaju velikim

promjenama dok se kod tradicionalnih svi važni elementi isplaniraju na početku te se taj plan provodi do kraja.



Slika 4: Agilni i tradicionalni razvoj (izrada autora prema Stephan, 2015.)

Na temelju navednog možemo utvrditit kako su agilne i tradicionalne metode veoma različite. Svaka od metoda ima svoje prednosti i mane, a izbor one koja im najbolje odgovara je u rukama tvrtke koja ih planira koristiti.

3.3. Prednosti i nedostatci agilnih metoda

Postoje mnoge prednosti agilnih metoda, a neke od glavnih su: osiguranje zadovoljstva korisnika, manji rizik razvoja, fleksibilnost, bolja komunikacija, brža isporuka gotovog proizvoda, lako prihvaćanje promjena, mogućnost kontinuiranog poboljšanja te činjenica da cilj ne mora uvijek biti jasno definiran (Milošević, 2018; Hales, 2021).

Česta i neprekidna isporuka dijelova gotovog proizvoda osigurava zadovoljstvo korisnika, omogućava bržu isporuku gotovog proizvoda i smanjuje rizik razvoja. Suradnja i česti sastanci članova tima osiguravaju bolju komunikaciju te daju prostora za konstantno poboljšanje članova tima. Naručitelj ne mora od samog početka projekta imati jasnu viziju kako će krajnji softver izgledati jer su promjene prihvatljive.

Agilne metode imaju i poneki nedostatak. Kao najčešće nedostatke Milošević (2018.) navodi da planiranje može biti manje konkretno, tim mora biti upućen, dokumentacija se može

zanemariti, programeri moraju biti u potpunosti posvećeni projektu te finalni proizvod može biti veoma različit od zamišljenog. Osim toga Vresk, Pihir i Tomičić Furjan (2020.) kao nedostatke dodaju i činjenice da postoji nedostatak specifikacije i dokumentacije te da troškovi mogu potencijalno biti veći.

Iako se smanjenjem dokumentacije nastojalo timu „olakšati posao“, kod manjka dokumentacije i specifikacije vrlo lako može doći do toga da se tim u potpunosti izgubi u zahtjevima. Valja također napomenuti, kako ideja stvaranja manje dokumentacije nije osmišljena s namjerom da se dokumetacija u potpunosti zanemari, a to se pri korištenju agilnih metoda nekada vrlo lako može dogoditi. Također, tim mora biti upućen i veoma posvećen projektu na kojem radi što zna biti iscrpljujuće. Zbog stalnih promjena, lako je moguće da softver na kraju uopće ne izgleda kako je na početku projekta zamišljen te da se prekorače planirani vremenski i novčani okviri.

3.4. Vrste agilnih metoda

Metoda sama po sebi označava plan ili niz koraka koji olakšavaju postizanje određenih ciljeva. Tako su agilne metode niz pravila kojim se timovima olakšava put do postizanja agilnog okruženja kako bi olakšali svoj posao te proizveli proizvod visoke kvalitete.

„Agilna metoda je iterativna i inkrementalna taktika za razvoj softvera koja koristi stalno planiranje, razumijevanje, nadogradnju, timsko partnerstvo, razvoj i isporuku. Agilni proces je fragmentiran u zasebne modele na kojima timovi rade, čime se potiče fleksibilnost za promjene.“ (Sarangam, 2022.)

Vrlo je važno napomenuti kako pojam *agile* ne označava metodu. *Agile* je skup vrijednosti i principa izraženih u Agilnom manifestu na temelju kojeg su se kasnije razvijale složenije agilne metode odnosno okviri. Neke od najpoznatijih agilnih metoda su: Scrum, Ekstremno programiranje (engl. Extreme Programming), Kanban, Metoda razvoja dinamičkih sustava (engl. Dynamic Systems Development Method), Razvoj vođen funkcionalnostima (engl. Feature-Driven Development), Lean razvoj softvera (engl. Lean Software Development), Adaptivni razvoj softvera (engl. Adaptive Software Development) i Kristal (engl. Crystal) (Altvater,2017.).

U nastavku rada ukratko ću objasniti neke od često korištenih metoda: Ekstremno programiranje, Kanban i Lean. Posebno poglavlje posvetit ću Scrum okviru, s obzirom na to da je

on prema popularnom „State of Agile Reportu“ tvrtke Digital.ai u posljednje tri godine najkorišteniji okvir za upravljanje agilnim projektima (Digital.ai, bez dat.)

3.4.1. Ekstremno programiranje

Ekstremno programiranje (u daljnjem tekstu XP) jedna je od prvih agilnih metoda. Razvio ju je Kent Beck u devedesetim godinama prošlog stoljeća te je u tom razdoblju bila najčešće korištena metoda (Fowler, 2013.).

Ova metoda smatra se jednom od najuspješnijih metoda za agilni razvoj softvera jer stavlja velik fokus na interakciju s korisnikom, a u konačnici i na njegovo zadovoljstvo (Sharma, Sarkar, Gupta, 2012., str. 894).

XP poboljšava projekt razvoja softvera fokusom na 5 važnih vrijednosti i principa (Wells, 2013.):

- Komunikacija – odnosi se na komunikaciju unutar tima i komunikaciju s naručiteljem
- Jednostavnost – odnosi se na jednostavnost koda i dizajna
- Povratna informacija – odnosi se na povratnu informaciju nakon testiranja softvera
- Poštovanje – odnosi se na poštovanje prema svakom članu tima
- Hrabrost – odnosi se na hrabri pristup programera svakodnevnim promjenama zahtjeva i tehnologije

Proces razvoja softvera unutar okvira ekstremnog programiranja kreće s prikupljanjem korisničkih zahtjeva. Ovisno o zahtjevima cjelina se dijeli na manje dijelove te službeno započinje faza planiranja. Tada se pišu korisničke priče, planiraju datumi izdanja, projekt se dijeli na iteracije te se pokreće planiranje prve iteracije. Slijedi faza razvoja. Tokom te faze moguće je pristizanje novih korisničkih zahtjeva koji se moraju uvažiti. Nakon razvoja slijedi testiranje najnovije verzije softvera. Preferirani način testiranja je testiranje jedinica (engl. Unit Testing) te je vrlo važno da sve jedinice prođu test prije izdavanja nove verzije. Ako se tokom faze testiranja pronađu greške u kodu one će biti ispravljene u sljedećoj iteraciji. Po završetku navedenih faza potrebno je napraviti provjeru projekta kojom se nastoji utvrditi koliko je posla do sada napravljeno i koliko je posla preostalo(Wells 1999; Sharma, Sarkar, Gupta, 2012., str. 894).

Svaka se iteracija razvija pomoću programiranja u paru odnosno procesa u kojem dva programera programiraju zajedno na jednom računalu. Jedan programer ima ulogu „suvozača“ i

provjerava kod drugog kolege. Nakon određenog vremena dolazi do zamjene mjesta te prijašnji „suvozač“ sada postaje onaj koji piše kod.

Osim programiranja u paru, ono što je XP uveo kao novosti u svijet razvoja softvera su i zajedničko vlasništvo nad kodom, učestalo integriranje, razvoj vođen testiranjem (engl. Test Driven Development) opsežni pregledi koda (engl. Code review) te refaktoriranje koda (Livermore, 2007., str. 82) .

3.4.2. Kanban

Iako je začet 1950-ih u Japanskoj automobilskoj industriji, Kanban je danas jedna od najčešće korištenih agilnih metoda u IT industriji. Riječ kanban se s japanskog ugrubo može prevesti kao znak ili kartica. To se odnosi na kartice koje se pomiču po poznatoj Kanban ploči. (Arbulu, Ballard, Harper, 2003., str. 3)

Glavna ideja Kanbana je vizualizacija posla, limitiranje posla u tijeku (engl. Work In Progress, u daljnjem tekstu WIP) te mjerenje vremenskog ciklusa (Ahmad, Markkula, Ovio, 2013., str. 10). Osnovni cilj Kanbana je postići isporuku na vrijeme (engl. Just in time).

Ako se pravilno koristi, Kanban ploča omogućava smanjenje posla u tijeku (WIP), lakše uočavanje uskih grla i ograničenja te koordinaciju posla unutar tima. (Corona, Pani, 2013., str. 2) Ploča je najčešće podijeljena na tri stupca, a to su (Parsons, Thorn, Inkila, MacCallum, 2018., str. 720):

- Nezavršeni rad ili za napraviti (engl. To do)
- U tijeku
- Gotovo

Po potrebi, navedeni stupci mogu se prilagođavati (dodati novi stupci, preimenovati gore navedeni stupci, maknuti nepotrebni stupci i slično). Najčešća praksa jest da se stupac u tijeku rastavi na više manjih stupaca.

Ploča funkcionira na način da se na nju, za početak, sve priče postavljaju u prvi stupac (u gornjem slučaju u stupac nezavršeni rad). Kada član tima započne s radom kartica korisničke priče na kojoj član započinje raditi seli se u sljedeći stupac (u gornjem slučaju stupac u tijeku). Kada član tima u potpunosti završi s implementacijom korisničke priče pomiče karticu u sljedeći, u našem slučaju i posljednji stupac (gotovo).

Ključna stvar koja razlikuje Kanban ploču od tipičnih agilnih ploča s korisničkim pričama jest da Kanban ploča ima ograničenje posla u tijeku odnosno postoji točno određen broj kartica koje istovremeno mogu biti u stupcu „U tijeku“. Ako dođe do situacije da je dostignut maksimalni dozvoljeni broj kartica u tom stupcu potrebno je korisničke priče s tih kartica što prije izvršiti i „pogurati ih“ u stupac „Gotovo“ radije nego ih vraćati u stupac „Nezavršeni posao“. (Parsons et al., 2018., str. 721)

3.4.3. Lean razvoj softvera

Abrahamsson, Ebert i Oza (2012.) definiraju Lean kao „proizvodnu paradigmu s krajnjim fokusom na stvaranje vrijednosti za kupca, eliminiranje otpada, optimizaciju tokova vrijednosti, osnaživanje ljudi i kontinuirano poboljšanje“.

Lean se, baš kao i Kanban, u početku koristio u proizvodnim pogonima, a glavni ciljevi bili su ojačati timove, smanjiti otpad, povećati fokus na kupce i težiti ka optimiziranju cijelog proizvodnog ciklusa (Abrahamsson, Ebert, Oza, 2012. str. 22).

Brz razvoj tehnologije i povećanje potražnje na tržištu povećali su i potrebu za boljom organizacijom procesa proizvodnje softvera te smanjenjem troškova procesa. Upravo to je ono što je potaknulo tvrtke u industriji razvoja softvera da razmisle o Lean razvojnom procesu kao potencijalnom rješenju tadašnjih problema.

Neke od glavnih prednosti Lean razvoja su brži razvoj softvera s manje troškova i napora, razvojni ciklus je kraći, gotov proizvod se nastoji kontinuirano poboljšavati, a tim je neovisan i odgovoran (Đumić, 2021., str. 2).

Lean razvoj softvera temelji se na sedam principa, a to su (Lutkevich, 2021.):

1. Eliminirati otpad – cilj je eliminirati nepotreban kod i funkcionalnosti
2. Izraditi kvalitetan proizvod – koristeći se raznim taktikama, kao što su programiranje u paru i razvoj vođen testovima, timovi koji koriste Lean razvoj softvera nastoje osigurati najveću kvalitetu proizvoda
3. Pojačati i proširiti učenje – znanje članova tima mora se međusobno dijeliti kroz preglede koda i na sastancima
4. Odgoditi obveze što je duže moguće – cilj je implementirati funkcije što kasnije tako da se izbjegne potreba za popravljanjem završenog posla

5. Dostavljati brzo – softver se izdaje često i brzo te se dobiva povratna informacija od korisnika na temelju koje se poboljšava daljnji proces razvoja
6. Poštovati ljude – poštivati članove tima i korisnike, dobro upravljati konfliktima, proaktivno komunicirati i dobivati čestu povratnu informaciju
7. Optimizirati cjelinu – tim ispituje proces od početka do kraja i nastoji ga poboljšati kako bi omogućio efikasan tok projekta.

4. Scrum

Scrum je najpopularniji okvir za vođenje agilnih projekata. Svojom jednostavnošću i slobodom koju pruža vrlo je čest odabir timova koji kreću na svoj put ka agilnosti.

Prema Schwaber u Sutherlandu, „Scrum je jednostavan okvir koji pomaže ljudima, timovima i organizacijama da stvore vrijednost kroz prilagodljiva rješenja za kompleksne probleme.“ (Schwaber, Sutherland, 2020. str. 3)

Scrum nije tehnika, metoda niti „kuharica“ koju je nužno pratiti. Scrum je okvir unutar kojeg možemo primijeniti različite metode, procese i tehnike. Kako navode Schwaber i Sutherland u svom Vodiču kroz Scrum (2020., str. 3) „Umjesto da ljudima daju detaljne upute, pravila Scruma usmjeravaju njihove odnose i interakcije“.

Scrum je vođen iterativnim i inkrementalnim pristupom, a za cilj ima optimizaciju procesa kontrolom i predviđanjem rizika te angažiranjem skupine ljudi čijim spojem vještina i stručnog znanja posao može biti uspješno završen (Schwaber, Sutherland, 2020. str. 3).

Scrum je jedna od agilnih metoda koja je najviše usmjerena na krajnje korisnike. Upravo je korisnik taj koji definira funkcionalnosti koje očekuje u krajnjem proizvodu (softveru), a kasnije tim te zahtjeve prioritizira i dijeli na faze koje se zatim jedna po jedna ostvaruju (Wysocki,2006., str. 209).

Scrum se temelji na sustavu pet vrijednosti, a to su: predanost, usredotočenost, otvorenost, poštovanje i hrabrost (Schwaber, Sutherland, 2020. str. 3). Ove vrijednosti Scrum tim uči kroz Scrum uloge, artefakte i događaje te rješavajući kompleksne probleme s kojima se svakodnevno susreću. Sve odluke koje donosi Scrum tim trebale bi se temeljiti na poštivanju ovih vrijednosti.

Unutar Scrum okvira propisani su preporučeni koncepti kao što su Scrum tim, Scrum događaji te Scrum artefakti kojima se timovi nastoje voditi i unutar kojih nastoje djelovati kako bi si olakšali svakodnevni rad na projektima razvoja softverskih proizvoda. Navedeni će koncepti biti detaljno objašnjeni u nastavku ovog poglavlja.

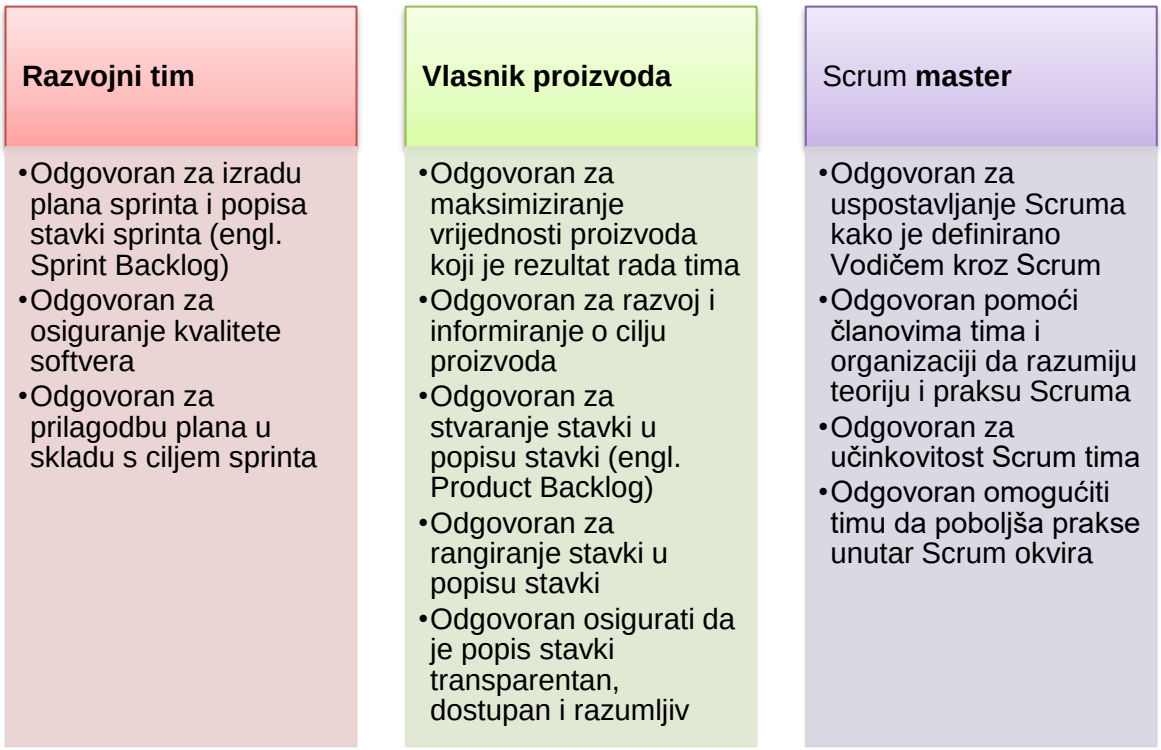
4.1. Scrum tim

Prema Schwaberovom i Sutherlandovom Vodiču kroz Scrum (2020., str. 5.) Scrum tim je „kohezivna grupa profesionalaca koja je u svakom trenutku usredotočena na jedan cilj, a to je cilj proizvoda (engl. Product Goal).“

Scrum tim sastoji se od maksimalno 10 članova. U taj tim ubraja se jedan Scrum master, jedan vlasnik proizvoda (engl. Product Owner) te razvojni tim (engl. Developers) (Schwaber, Sutherland, 2020. str. 5). Iako naziv razvojni tim predlaže da se radi isključivo o članovima koji pišu programski kod, pod pojmom razvojni tim zapravo se podrazumijevaju svi članovi tima koji su dio stvaranja bilo kojeg dijela upotrebljivog inkrementa unutar sprinta. Pa tako razvojni tim mogu činiti analitičari, dizajneri, testeri, itd. (Alblas, 2019.).

Glavna obilježja Scrum tima su da je multifunkcionalan (engl. cross-functional) i autonoman (engl. self - managed). Da su timovi multifunkcionalan znači da spoj znanja i vještina svih članova tima omogućava da tim u svakom sprintu stvori i isporuči novu vrijednost. Timovi su također i autonomni, a pod tim se podrazumijeva da članovi tima samostalno odlučuju tko će, kada i što raditi. (Schwaber, Sutherland, 2020. str. 5.)

Slika 5 prikazuje članove Scrum tima te njihove glavne odgovornosti.



Slika 5: Članovi Scrum tima i njihove odgovornosti (izrada autora prema Schwaber, Sutherland, 2020. str. 5 i 6)

Osim gore navedenih odgovornosti Scrum master ima i određene odgovornosti prema svakom od članova tima. Na primjer, razvojnom timu Scrum master mora pomoći da postanu bolji u autonomiji i multifunkcionalnosti, spriječiti prepreke koje mogu utjecati na napredak sprinta, pomoći im da se usredotoče na stvaranje inkrementa velike vrijednosti i osigurati da se svi Scrum događaji održe te da budu unutar određenog vremenskog okvira. Vlasniku proizvoda Scrum master pomaže u pronalaženju tehnika za definiranje cilja proizvoda, pomaže u rangiranju i upravljanju popisom stavki, olakšava suradnju dionika i pomaže u empirijskom pristupu planiranju proizvoda. Posljednje, Scrum master pomaže organizaciji tako što ima ulogu „trenera“ koji pomaže organizaciji usvojiti Scrum kroz planiranje i savjetovanje, pomaže svim dionicima shvatiti i primijeniti empirijski pristup radu te uklanja prepreke i ograničenja između dionika i timova. (Schwaber, Sutherland, 2020. str. 6. i 7.)

Česte greške koje se znaju dogoditi kod korištenja Scruma, a vezane su uz tim su da sve navedene uloge u timu ne postoje, da se uloge kombiniraju te da unutar tima nastane hijerarhija (Alblas, 2019.).

Svaka je uloga u timu vrlo važna i za dobro funkcioniranje tima svaka od njih mora postojati. Izostavljanje ili kombiniranje uloga može dovesti do usporavanja napretka tima, smanjenja fokusa te nemogućnosti isporuke upotrebljivog inkrementa. Spajanje i izbacivanje uloga najčešće se provodi jer tvrtke nastoje povećati efikasnost, no cilj Scruma nije povećati efikasnost nego efektivnost. Najčešće izbačena uloga je Scrum master, a do toga dolazi jer tvrtke, nedovoljno informirane o Scrumu, ne shvaćaju važnost te uloge u timu i organizaciji. (Alblas, 2019.)

Prema Schwaberovom i Sutherlandovom Vodiču kroz Scrum (2020., str. 5) „unutar Scrum tima ne postoje podtimovi ni hijerarhije." Mnoge tvrtke prilikom uvođenja Scruma, često nesvjesno, uvedu hijerarhiju i u Scrum tim. Najčešće je to vidljivo u odnosu vlasnika proizvoda i programera gdje je vlasnik proizvoda nadređeni (ponekad čak i voditelj tima), a programeri podređeni(Alblas, 2019.). Hijerarhija u timu nije dobra jer se ona kosi s idejom autonomnog tima odnosno idejom da su svi članovi tima jednaki te da su pojedinci u stanju u dogovoru s ostatkom tima odlučiti tko će, što, kada i kako raditi.

4.2. Scrum događaji

Sljedeći vrlo važan element Scruma su Scrum događaji koje bismo mogli definirati kao specifične vrste sastanaka kojim Scrum tim planira budući i reflektira se na prethodni rad u svrhu izrade kvalitetnijeg proizvoda.

Glavni cilj Scrum događaja je pomoći timovima održati Scrum vrijednosti. Također Schwaber i Sutherland navode da „u Scrumu događaji služe za stvaranje pravilnosti i smanjenje potrebe za sastancima koji nisu definirani Scrumom." (Schwaber, Sutherland, 2020. str. 7.) Svaki od događaja namijenjen je da se ostvari potrebna transparentnost što omogućava da su svi članovi tima u svakom trenutku „u toku" odnosno da znaju što je do sada napravljeno, što treba biti napravljeno, ima li mjesta za napredak i kako generalno napreduje projekt.

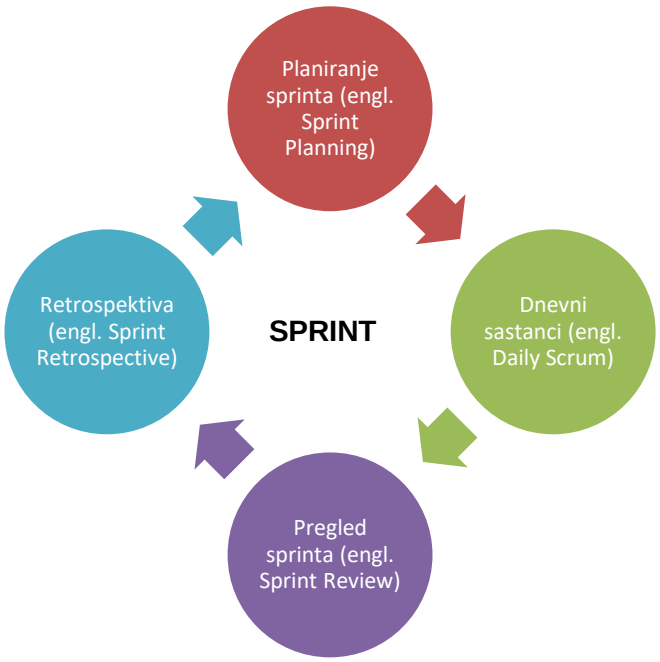
Događaj koji obilježava Scrum je sprint. Kako navode Schwaber i Sutherland (2020., str. 7) „Sprintevi su bïlo Scruma, u kojem se ideje pretvaraju u vrijednost."

Sprint je događaj fiksne duljine, najčešće u trajanju od dva do četiri tjedna unutar kojeg Scrum tim nastoji proizvesti upotrebljivi inkrement. Novi sprint započinje odmah po završetku prethodnog. (Schwaber, Sutherland, 2020. str. 7.)

Sprintevi osiguravaju da će tim ispuniti agilno načelo česte isporuke upotrebljivog softvera. Također sprintevi osiguravaju da se barem jednom mjesečno pregleda napredak projekta i prilagode radnje koje vode do ostvarenja cilja proizvoda (Rehkopf, bez dat.).

Postoji nekoliko preporuka koje vrijede tokom trajanja sprinta, a to su da se ne uvode promjene koje bi mogle ugroziti cilj sprinta, kvaliteta se ne smije smanjiti, popis stavki se može mijenjati te je moguće detaljnije razjasniti opseg posla i o njemu pregovarati s vlasnikom proizvoda. Sprint je dozvoljeno otkazati samo u slučajevima kada cilj sprinta zastari, a za to je ovlašten vlasnik proizvoda (Schwaber, Sutherland, 2020. str. 7.) .

Tijek sprinta možete vidjeti na slici 6.



Slika 6: Tijek sprinta (izrada autora prema Rehkopf, bez dat.)

Svaki sprint započinje s događajem koji se naziva planiranje sprinta (engl. Sprint Planning). To je događaj na kojem tim nastoji odgovoriti na tri osnovna pitanja (Rehkopf, bez dat. i Schwaber, Sutherland, 2020. str. 8.):

- Zašto je ovaj sprint vrijedan? – Nastoji se utvrditi na koji način proizvod može povećati svoju vrijednost. Ovo pitanje pomaže definirati cilj sprinta.
- Koji posao može biti odrađen u ovom sprintu? - Programeri u suradnji s vlasnikom proizvoda prolaze kroz popis stavki i biraju koje se stavke mogu odraditi u tom

sprintu. Koliko se stavki stavlja u sprint često ovisi o procjeni programera koja je temeljena na njihovom prijašnjem iskustvu.

- Kako će se odabrani posao odraditi? – Za svaku stavku odabranu u prethodnom koraku procjenjuje se što je potrebno odraditi kako bi stavka, a samim time i inkrement, dostigao definiciju gotovog.

Procesom koji se vodi ovim pitanjima na kraju se dolazi do popisa stavki sprinta koji čine cilj sprinta, popis odabranih stavki iz popisa stavki te plan njihove isporuke.

Nakon uspješno isplaniranog sprinta on službeno može početi. Svaki dan tokom trajanja sprinta provodi se 15 minutni dnevni sastanak (engl. Daily Scrum). Ideja ovog sastanka je provjeriti napredak u ostvarenju cilja sprinta, prilagoditi popis stavki sprinta te riješiti potencijalno nastale probleme i nedoumice koje bi mogli spriječiti uspjeh ostvarenja cilja sprinta (Rehkopf, bez dat.; Schwaber, Sutherland, 2020. str. 9.).

Za razliku od ostalih događaja na ovom događaju prisustvuje samo razvojni tim. U slučaju da vlasnik proizvoda ili Scrum master aktivno rade na stavkama popisa stavki sprinta i oni se mogu priključiti dnevnom sastanku(Schwaber, Sutherland, 2020. str. 9.). Razvojni tim ima potpunu slobodu odabrati koje tehnike i alate će koristiti tokom dnevnog sastanka sve dok je fokus sastanka na planiranju što će se raditi u sljedeća 24 sata te kako što efektivnije i efikasnije postići cilj sprinta.

Nakon što je vremenski okvir sprinta završio slijedi pregled sprinta (engl. Sprint Review). Na ovaj događaj dolazi Scrum tim i zainteresirane strane pozvane od strane vlasnika proizvoda (Gonçalves, 2018, str. 4). Svrha pregleda sprinta je provjeriti ishod sprinta i odrediti buduće planove. Tim ključnim dionicima predstavlja što je do sada napravio i raspravlja o napretku do ostvarenja cilja proizvoda (Schwaber, Sutherland, 2020. str. 9.).

Ideja ovog sastanka nije da tim dobije pohvalu ili kritiku nego da zatraže povratne informacije od zainteresiranih strana, da dogovore što će dalje i da zajedno utvrde da li je došlo do kakvih promjena i problema tokom trajanja prethodno završenog sprinta.

Posljednji, ali svakako ne najmanje važan događaj u sprintu je retrospektiva (engl. Sprint Retrospective). Svrha ovog događaja je povećati kvalitetu i učinkovitost. Tokom ovog događaja članovi tima pregledavaju kako je prošao prethodni sprint. Zajedno pokušavaju utvrditi što je dovelo do poteškoća ili prepreka, pregledavaju interakciju u timu, procese, korištene alate i definiciju gotovog. Cilj je da iz toga izvuku što je potrebno promijeniti, a što je bilo dobro i treba

se nastaviti kako bi se postigla najveća učinkovitost. (Schwaber, Sutherland, 2020. str. 9. i Gonçalves, 2018. str. 4)

4.3. Scrum artefakti

Prema Schwaberu i Sutherlandu (2020., str. 10.) „Artefakti u Scrumu predstavljaju posao ili vrijednost te su osmišljeni kako bi maksimizirali transparentnost ključnih podataka.“

Dakle, artefakti predstavljaju glavni izvor informacija kojima se Scrum tim služi tokom sprinta. Artefakti se tokom sprinta koriste za planiranje ciljeva i posla, kreiranje zadataka da bi se određeni ciljevi ispunili, organiziranje tih zadataka u sprinteve temeljem njihove važnosti i međusobne ovisnosti, ispunjenje zadataka te za pregled i analizu rezultata i planiranih ciljeva (Harris, bez dat.).

Artefakte čine popis stavki proizvoda (engl. Product Backlog), popis stavki sprinta (engl. Sprint Backlog) te inkrement (engl. Increment) (Schwaber, Sutherland, 2020. str. 10.).

Popis stavki proizvoda je, kako i samo ime kaže, popis stavki nastalih raščlanjivanjem proizvoda. U taj popis spadaju sve nove funkcionalnosti, popravci grešaka, zadaci i poslovni zahtjevi koji su potrebni za razvoj finalnog softvera (Harris, bez dat.).

Popis stavki proizvoda je „ poredan i stalno promjenjiv popis stvari potrebnih za poboljšanje proizvoda. To je jedini izvor posla koji Scrum tim preuzima“ (Schwaber, Sutherland, 2020. str. 10.). Iz ovog se popisa na početku svakog sprinta odabiru stavke koje će se izvršiti taj sprint.

Popis stavki proizvoda smatra se „živim“ artefaktom jer se po potrebi može ažurirati novim informacijama. Događaj ažuriranja popisa stavki naziva se razrada popisa stavki (engl. Product Backlog Refinement). Na događaju ažuriranja popisa stavki sastaju vlasnik proizvoda i razvojni tim te zajedno dodaju nove stavke, detalje i procjene postojećim stavkama i slažu redoslijed stavki u popisu stavki. (Harris, bez dat.; Alblas, 2018.)

Popis stavki proizvoda u sebi mora sadržavati i cilj proizvoda koji opisuje buduće stanje proizvoda koje može biti smjernica Scrum timu. Cilj proizvoda je dugoročni cilj Scrum tima. (Schwaber, Sutherland, 2020. str. 10.)

Odabirom stavki iz popisa stavki na početku svakog sprinta nastaje popis stavki sprinta. Stavke u popisu stavki sprinta prikazuju zadatke, popravke i rad koji će se odraditi u tom sprintu

te plan za isporuku inkrementa. Popis stavki sprinta je plan koji sastavlja razvojni tim kako bi u stvarnom vremenu prikazali posao koji planiraju obaviti tijekom sprinta sa svrhom postizanja cilj sprinta (Harris,, bez dat.; Schwaber, Sutherland, 2020. str. 10.).

Stavke iz popisa stavki koje se planiraju izvršiti u tom sprintu razlažu se na manje stavke (Harris, bez dat.). Na primjer, stavka „Razviti plaćanje“ u aplikaciji online prodaje može se rastaviti na stavke „Razviti plaćanje kreditnom karticom“, „Razviti plaćanje putem PayPall-a“ i „Razviti plaćanje kripto valutama“ koje će biti odrađene u prvom sprintu te će time postati dio popisa stavki sprinta. Baš kao i popis stavki ovaj se popis može ažurirati kako više informacija postaje dostupno.

Jednom postavljeni cilj sprinta ne bi se trebao mijenjati. U slučaju da se ispostavi da trenutne stavke ne doprinose ispunjenju cilja sprinta potrebno ih je, u suradnji s vlasnikom proizvoda, ažurirati (Schwaber, Sutherland, 2020. str. 10.).

Posljednji Scrum artefakt je inkrement. Prema Schwaberu i Sutherlandu (2020., str 10.) „Inkrement je konkretna stepenica prema cilju proizvoda.“ Spajanjem svih inkremenata nastalih tokom trajanja projekta nastaje finalni proizvod.

Da bi se odrađeni posao smatrao gotovim mora biti u skladu s postavljenom definicijom gotovog. To je „formalan opis stanja inkrementa u kojem on zadovoljava mjere kvalitete koje su potrebne za proizvod“ (Schwaber, Sutherland, 2020. str. 10.). Osnovna ideja definicije gotovog je da osigura da svi članovi tima imaju zajedničko razumijevanje posla koji je obavljen. Niti jedan član tima ne smije se ni u jednom trenutku pitati „Je li ovo zaista gotovo?“. Ako definicija gotovog nije dio standarda organizacije u kojoj tim posluje, tim mora sam stvoriti definiciju gotovog (Schwaber, Sutherland, 2020. str. 10. i Harris bez dat.).

5. Priprema Scrum projekta u Jiri

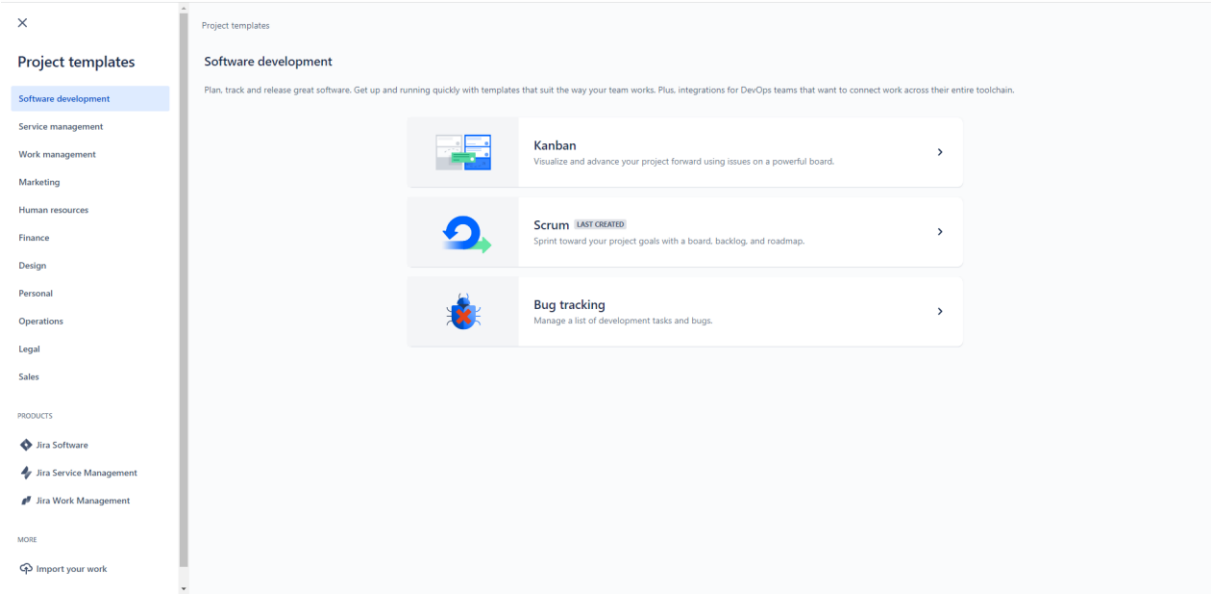
S razvojem projektnog menadžmenta na tržištu su se počeli pojavljivati razni alati koji pružaju potporu projektnim timovima u efikasnijem i efektivnijem upravljanju projektima na kojima rade. Neki od primjera takvih alata su: Trello, Asana, Hive, Wrike i mnogi drugi, no onaj koji se ističe svojom popularnošću i jednostavnošću definitivno je Jira.

2002. godine australska tvrtka Atlassian izdala je prvu verziju Jira softvera. Jira je inicijalno bila zamišljena kao alat za praćenje problema (engl. Issues) i grešaka u kodu (engl. Bugs), ali se razvojem tehnologije i tržišta razvila u snažan alat koji danas služi za efikasno upravljanje projektima (Atlassian, bez dat.).

Kako navode na svojoj web stranici, Jira nudi usluge raznim vrstama timova pa tako i agilnim timovima. U sklopu svoje agilne ponude imaju Scrum i Kanban ploče koje pružaju mogućnosti bilježenja problema, stvaranje procjena, praćenje i izradu izvještaja o napretku, slanje email notifikacija te stvaranje i korištenje popisa stavki (Atlassian, bez dat.).

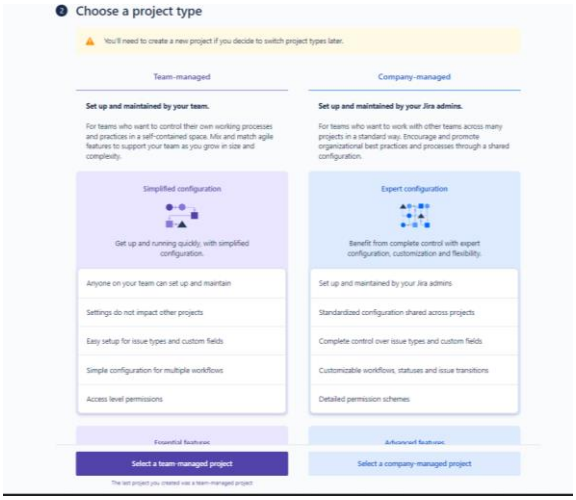
S obzirom da se ovaj rad fokusira na Scrum okvir u nastavku će biti prikazano kako uz pomoć Jire pripremiti Scrum projekt.

Za početak potrebno je stvoriti projekt. To činimo tako da u Jira softveru pritisnemo na gumb „Create project“. Pritiskom na gumb otvara se novi prozor (vidljiv na slici zaslona 1) u kojem odabiremo neki od već kreiranih predložaka projekta. S obzirom da kreiramo Scrum projekt odabrati ćemo Scrum predložak.



Slika zaslona 1: Odabir predloška projekta u Jira softweru (https://www.atlassian.com/software/jira) (izrada autora)

Sljedeća dva koraka su odabir tipa projekta te davanje imena projektu. U ovom slučaju biramo „Team managed“ tip te dajemo naziv projektu „E-glasanje“ s obzirom da se zamišljeni projekt pokreće u svrhu proizvodnje softvera koji će omogućiti online glasanje.



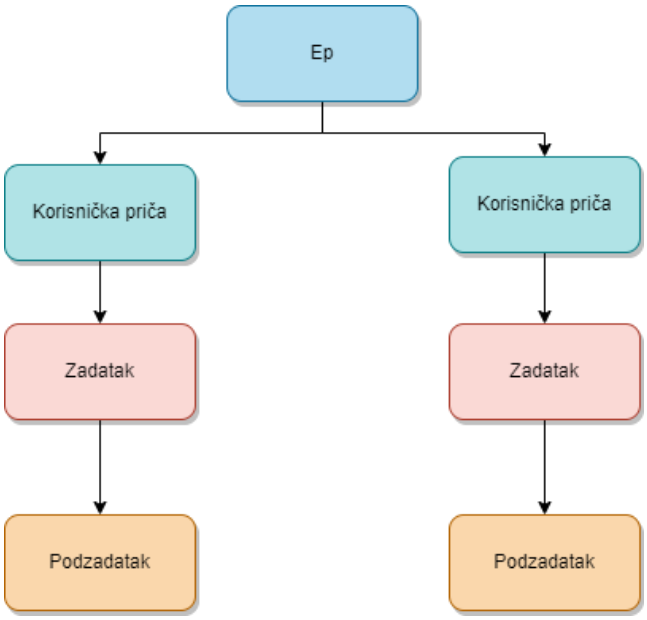
Slika zaslona 2: Odabir tipa projekta u Jira softveru (https://www.atlassian.com/software/jira) (izrada autora)

Kako bismo pripremili projekt za početak rada potrebno je prilagoditi određene dijelove predloška koje nam nudi Jira softver.

Jira nudi četiri tipova zadataka (engl. Issue types):

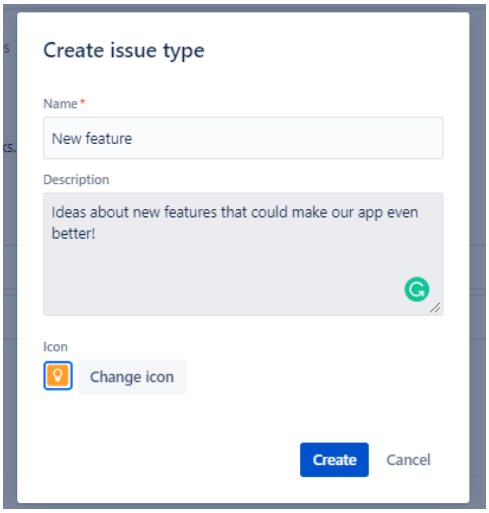
- Ep (engl. Epic)
- Korisnička priča (engl. Story)
- Zadatak (engl. Task)
- Podzadatak (engl. Subtask)

Hijerarhija je vidljiva na slici 7.



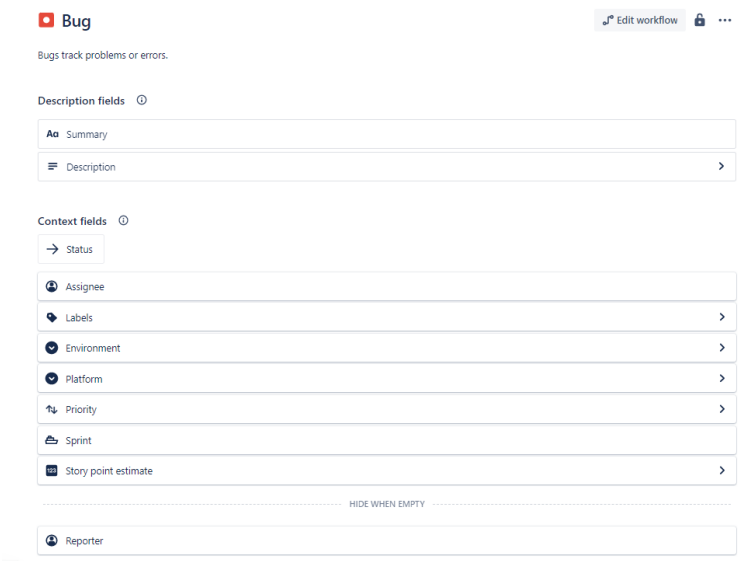
Slika 7: Hijerarhija tipova problema u Jira predlošku (izrada autora)

Osim navedenih tipova zadataka postoji još i tip „Bug“ odnosno greška te je moguće dodati i vlastite tipove. Našem ćemo projektu dodati još jedan tip zadatka pod nazivom „New feature“. Ovaj tip bit će namijenjen za bilježenje svih ideja o potencijalnim budućim funkcionalnostima.



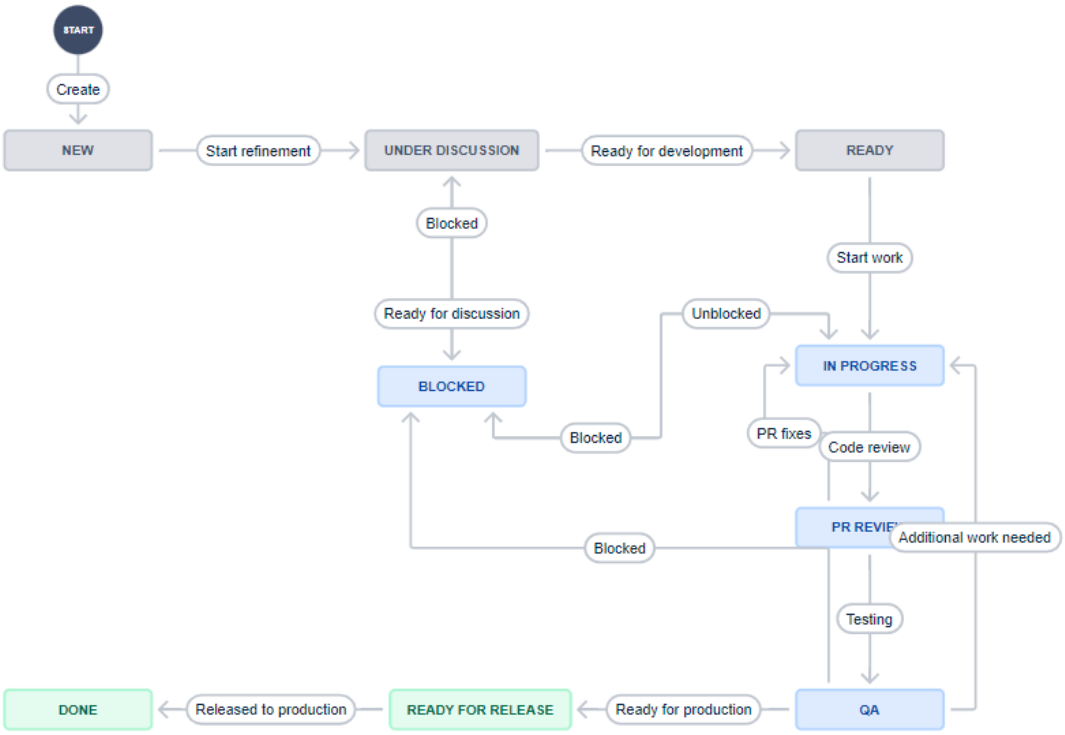
Slika zaslona 3: Dodavanje novog tipa problema u Jira softveru (https://www.atlassian.com/software/jira) (izrada autora)

Sljedeće što je potrebno napraviti kako bismo prilagodili Jira predloške našim potrebama je dodavanje potrebnih polja opisa u svaki od tipa problema. Tako smo, na primjer, u tip problema „Bug“ odnosno greška dodali dodatna polja „Enviroment“, „Platform“ i „Priority“. „Enviroment“ označava okruženje u kojem je greška pronađena, „Platform“ označava platformu na kojem je greška pronađena, a „Priority“ prioritet za rješavanje navedene greške. Navedeno je prikazano na primjeru na slici ispod.



Slika zaslona 4: Dodana polja u tip problema "Bug" u Jira softveru (https://www.atlassian.com/software/jira) (izrada autora)

Posljednje što je potrebno prilagoditi u predlošku je „Workflow“ odnosno tijek rada. Zadani workflow temelji se samo na 3 stanja, a to su „To do“, „In progress“ i „Done“. Kako bi naš projekt bio lakši za praćenje dodati ćemo još nekoliko stanja, a tijek iz jednog u drugo stanje vidljiv je na slici ispod.

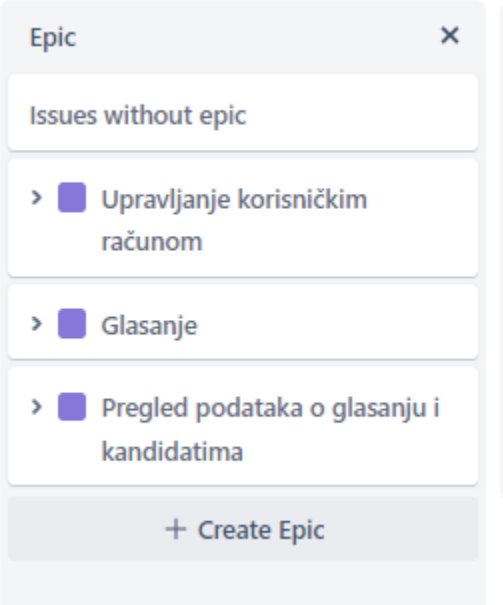


Slika zaslona 5: Tijek rada u projektu izrađenom u Jira softveru (https://www.atlassian.com/software/jira) (izrada autora)

Nakon što je određeni tip problema (npr. korisnička priča) kreiran on prvo zauzima stanje Novo (engl. New). Kada se navedena priča počinje detaljizirati ona prelazi u status Pod raspravom (engl. Under discussion). Nakon što je priča dovoljno detaljizirana prelazi u stanje Spremno (engl. Ready) te je priča spremna za početak razvoja. Kada se priča krene razvijati prelazi u stanje U toku (engl. In progress). Nakon što je određena priča razvijena potrebno je da prvo prođe Pregled koda (engl. Pull Request, na slici 5 pod nazivom PR review), a zatim i Testiranje (engl. Quality Assurance, na slici 5 pod nazivom QA). Nakon što je razvijena korisnička priča testirana prelazi u stanje Spremno za izdavanje (engl. Ready for release) te tamo ostaje dok konačno nije izdana u produkciju kada zauzima stanje Gotovo (engl. Done). Važno je napomenuti da priča iz stanja Pod raspravom, U toku i Testiranje može preći u stanje Blokirano (engl. Blocked) te se iz tog stanja također može vratiti u sva navedena stanja. Isto tako, u slučaju da se u Pregledu koda ili u Testiranju pronađu greške u kodu korisnička priča se vraća u stanje U toku.

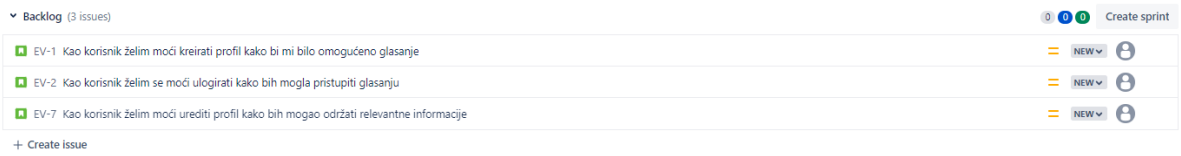
Kreiranjem tijeka rada završili smo s prilagodbom Jira predloška našim potrebama te je vrijeme za stvaranje Popisa stavki proizvoda. U njega dodajemo za početak Ep-ove koji opisuju

generalnu funkcionalnost aplikacije. U našem slučaju dodali smo tri Ep-a koja su vidljiva na slici zaslona 6.



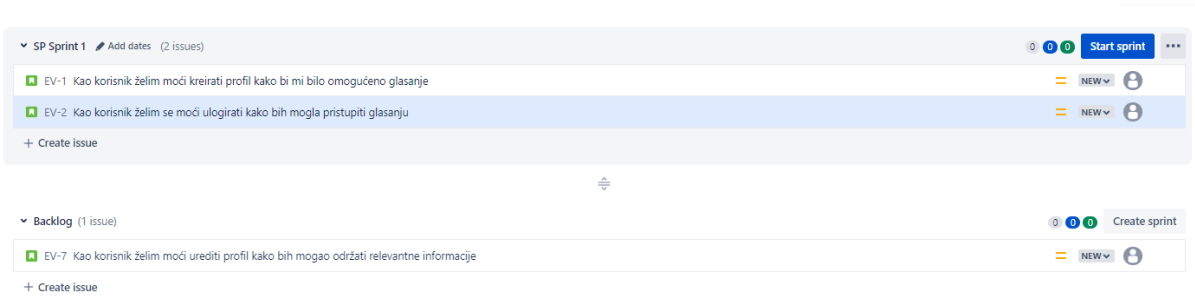
Slika zaslona 6: Epovi izrađeni u Jira softveru (https://www.atlassian.com/software/jira) (izrada autora)

Nakon što smo kreirali epove u svaki od epova dodajemo korisniče priče. Korisničke priče napisane su po dobroj praksi u obliku „Kao <uloga> želim moći <cilj> kako bih <korist>“.



Slika zaslona 7: Korisničke priče izrađene u Jira softveru (https://www.atlassian.com/software/jira) (izrada autora)

Kada smo dodali sve željene epove, korisničke priče i potencijalne nove funkcionalnosti spremni smo za početak projekta te korisničke priče možemo preseliti u prvi sprint. Time službeno možemo započeti rad na našem Scrum projektu.



Slika zaslona 8: Korisničke priče u prvom sprintu izrađenom u Jira softveru ((https://www.atlassian.com/software/jira) (izrada autora)

Iako se razvojem tehnologije, ali i metoda upravljanja projektima razvilo mnogo alata za online upravljanje projektima Jira je i dalje jedan od vodećih alata. Ono što izdvaja Jiru od konkurencije su jednostavnost i intuitivnost korištenja. Smatram da se svatko može samostalno snaći i prilagoditi Jira ploču za ono što mu je potrebna. Ako to nije slučaj, tvrtka Atlassian omogućila je na svojim web stranicama razne pisane i video upute koje podučavaju korisnike kako koristiti i unaprijediti svoje iskustvo s Jira alatom. Također velika prednost su i već gotovi predlošci što timovima omogućava da odmah započnu s praćenjem svojih projekata. Upravo zbog navedenih prednosti Jiru danas koriste mnogi timovi te njihova zajednica iz dana u dan raste.

6. Studija slučaja - Korištenje agilnih metoda i Scrum okvira u praksi

Iako teorija dosta detaljno opisuje kako bi određeni koncepti trebali funkcionirati u praksi, to često nije slučaj. Uzrok tome su razni vanjski i unutarnji čimbenici na koje ponekad tvrtka i zaposlenici ne mogu utjecati. Kako bi se prikazalo kako teorija Scruma funkcionira u praksi u ovom će radu biti korištena metoda studije slučaja.

6.1. Metodologija istraživanja

Metodom studije slučaja nastoji se analizirati određena pojava, proces, osoba, grupa ili događaj te se nastoji odgovoriti na pitanja „kako“ i „zašto“ su se određene situacije ili okolnosti dogodile (Heale, Twycross, 2018.).

U ovom radu studija slučaja omogućit će usporedbu Scrum timova, artefakata i događaja teorijski opisanih u prethodnom poglavlju s njihovim stvarnim izvođenjem. S obzirom na to da je Scrum samo okvir unutar kojeg tvrtke mogu koristiti metode i tehnike koje im odgovaraju, studija slučaja također će omogućiti da uvidimo primjere nekih metoda i tehnika koje se koriste u praksi.

Istraživanje se provodilo u suradnji sa zaposlenicima jedne hrvatske IT agencije za razvoj softvera, kao dio pilot prakse na preddiplomskom studiju u sklopu projekta Study4Career koji provodi Fakultet organizacije i informatike Sveučilišta u Zagrebu.

Navedena tvrtka odabrana je za istraživanje jer zadovoljava unaprijed postavljene kriterije. S obzirom da se studija slučaja temelji na istraživanju kako Scrum kao okvir za vođenje projekata razvoja softvera funkcionira u praksi, osnovni kriteriji bili su primarna djelatnost poduzeća i stupanj korištenja agilnih metoda (posebice Scrum okvira) na projektima razvoja softverskih proizvoda. Ova IT tvrtka je agencija za razvoj softverskih rješenja, a to čini kroz niz projekata vođenih na agilni način te time zadovoljava oba kriterija.

Cilj istraživanja je dobiti odgovore na pitanja kako teorija vezana za Scrum okvir i njegova glavna obilježja (tim, događaji i artefakti) predstavljena kroz ovaj rad funkcioniraju u praksi. Kroz istraživanje nastojat će se dobiti odgovore na pitanja vezana za korištenje Scruma u projektima izrade softverskih rješenja te na nekoliko pitanja vezanih za korištenje agilnih metoda općenito.

Primjeri istraživačkih pitanja na koja će se nastojati pronaći odgovor su:

- Koji se Scrum događaji odvijaju i koliko često,
- Tko dolazi na događaje,
- Koje su odgovornosti članova Scrum tima,
- Što to tvrtka čini drukčije od onoga kako je to opisano u Vodiču kroz Scrum
- Koji su često korišteni alati i tehnike u primjeni Scrum okvira.

Proučavajući literaturu, a posebno knjigu „Case Study Research: Design and Methods“ autora Roberta K. Yina (2004.) za mjerne instrumente korištene u sklopu ove studije slučaja odabrani su polustrukturirani intervju i neposredno promatranje.

Polustrukturirani intervju odabran je jer omogućava intervju sa zaposlenicima po unaprijed određenom setu pitanja, ali uz mogućnost postavljanja potpitanja u svrhu dodatnog objašnjenja pojedinih tema.

Neposredno promatranje odabrano je jer omogućava direktan uvid u to kako agilne metode, a posebice Scrum, funkcioniraju u svom prirodnom okruženju. Promatranje će se provoditi prisustvom autora rada na sastancima vezanim uz agilno upravljanje projektima te promatranjem svakodnevnog rada Scrum tima.

Prije intervjua svaki ispitanik (dva zaposlenika agencije na poziciji voditelja projekta i Scrum Mastera) potpisao obrazac za pristanak na sudjelovanje u istraživanju kojim potvrđuje da se slaže s korištenjem njegovih odgovora u svrhu izrade ovog rada. Spomenuti obrazac također se može pronaći u prilogu ovog rada (Prilog 1). Tokom polustrukturiranog intervjua svakom od zaposlenika postavljena su ista pitanja otvorenog tipa, a ona su dostupna u prilogu ovog rada (Prilog 2).

Intervjui i neposredno promatranje provedeni su u srpnju 2022. godine u poslovnim prostorima tvrtke. Prilikom provođenja intervjua pratila su se pitanja zapisana u popisu pitanja vidljivih u prilogu ovog rada te su, po potrebi, postavljena dodatna potpitanja u svrhu lakšeg razumijevanja određenih tema. Svaki je intervju, uz pristanak zaposlenika, snimljen što je pomoglo u skraćivanju vremena intervjua te olakšavanju daljnje analize dobivenih odgovora. Neposredno promatranje provodilo se prisustvom autora ovog rada u svakodnevnom radu i sastancima Scrum timova te vođenjem privatnih bilješki koje su korištene za daljnju analizu.

Kasnijom analizom dobivenih odgovora te na temelju podataka dobivenih neposrednim promatranjem formiran je finalni zaključak o tome kako određeni elementi funkcioniraju u praksi te je to uspoređeno s teorijom opisanom u radu.

6.2. Studija slučaja

Studija slučaja provedena je u suradnji s jednom poznatom IT agencijom (u daljnjem tekstu Agencija) za razvoj softvera osnovanom 2005. godine. Svoj rad na projektima započeli su oslanjajući se na tradicionalnu, vodopadnu metodu što je značilo da su softveri zatraženi od klijenta morali biti ostvareni u određenom budžetu, određenom vremenskom periodu i po unaprijed određenim zahtjevima. Razvojem tržišta, ali i same Agencije, polako su počeli uočavati probleme koje donosi vodopadna metoda te su ih nastojali riješiti na što jednostavniji način. Jedan od primjera problema s kojima su se svakodnevno susretali je niska uključenost klijenta u izradi softvera. To je često rezultiralo time da klijent na kraju projekta zaključi da ono što je napravljeno nije ono što je on zamislio. Kako bi navedeni problem riješili voditelji projekata u Agenciji složili su se kako je potrebno što više uključiti klijenta u proces razvoja te nastojati biti što više transparentni s klijentom i članovima tima koji rade na projektu. Krenuli su tražiti metode koje mogu riješiti navedene probleme te su se susreli s agilnim metodama koje su kroz godine postupno uveli na gotovo sve projekte. Danas svoje projekte provode u obliku kontinuiranih suradnji s raznim klijentima gdje se vremenski okvir i budžet projekta mijenjaju prema potrebama tržišta.

Fleksibilnost koju im je pružilo uvođenje agilnih metoda opisuju kao dvosjekli mač. Velika prednost fleksibilnosti koju pružaju agilne metode je zadovoljan klijent koji dobiva ono što treba i želi, konstantno je uključen u proces razvoja (u svakom trenutku zna što se događa na projektu) te ima slobodu u bilo kojem trenutku zatražiti promjenu. S druge strane, prevelika fleksibilnost uzrokuje da razvoj proizvoda traje duže nego što bi trebao te su troškovi, koji proizlaze iz potrebe za članovima tima koji će konstantno raditi na projektu do njegovog kraja, osjetno veći.

Kako bi sve svoje projekte imali pod kontrolom i mogli pratiti tijek svakog od njih koriste razne alate kao što su Jira, Productive, DevOps, Trello, Monday i slični. Napredak projekta „drže pod kontrolom“ - koristeći razne ceremonije koje propisuje Scrum okvir unutar kojeg se provodi većina trenutno aktivnih projekta. Scrum su počeli aktivno koristiti i istraživati 2017. godine te su ga do danas prilagodili svojim potrebama.

S obzirom da se radi o IT agenciji ova tvrtka ima timove koji vrlo često nisu stabilni (povremene izmjene ljudi u timovima). Upravo zato njihovi timovi ne mogu u potpunosti raditi po Scrum okviru onako kako je propisano već sa sitnim preinakama. Važno je naglasiti kako je to u potpunosti prihvatljivo, s obzirom na to da je Scrum samo okvir unutar kojeg tim ima slobodu odabrati alate i tehnike koje će im pomoći dostići željeni cilj.

Od alata i tehnika koje koriste u svom svakodnevnom radu, a nisu propisani u Vodiču kroz Scrum, Scrum timovi ove Agencije koriste se Kanban pločom koju se preuzeli iz Kanbana, korisničkim pričama koje su preuzeli iz Ekstremnog programiranja te raznim dodatnim alatima i tehnikama kao što su unaprijed planirani sastanci ažuriranja popisa stavki, raspisivanje testnih slučajeva, itd.

U nastavku ovog poglavlja ukratko ćemo usporediti propisanu teoriju i praksu za svaki od tri važna koncepta Scruma: Scrum tim, Scrum događaji i Scrum artefakti.

Scrum tim

Tablica 2: Usporedba teorije i prakse u Scrum timu

	TEORIJA	**PRAKSA**
BROJ ČLANOVA TIMA	Maksimalno 10	8-10
ULOGE U TIMU	Jedan Scrum master, jedan vlasnik proizvoda i razvojni tim	Jedan Scrum master, jedan vlasnik proizvoda i razvojni tim
TKO ČINI RAZVOJNI TIM	Svi koji pridonose razvoju upotrebljivog inkrementa (analitičari, dizajneri, testeri)	Dizajner, jedan android programer, jedan ios programer, jedan ili dva back-end programera
GLAVNE ODGOVORNOSTI SCRUM MASTERA	Uspostavljanje Scruma kako je definirano vodičem kroz Scrum i educiranje tima i organizacije o Scrum okviru	Educira zaposlenike o korištenju Scruma, miče prepreke, pomaže razvojnom timu i vlasniku proizvoda
GLAVNE ODGOVORNOSTI VLASNIKA PROIZVODA	Brine o popisu stavki, odgovoran za maksimiziranje vrijednosti proizvoda koji je rezultat rada tima	Priprema popis stavki (određuje prioritete, poslovnu vrijednost stavki,..)
GLAVNE ODGOVORNOSTI RAZVOJNOG TIMA	Izrada popisa stavki sprinta, izrada kvalitetnog softvera	Izrada popisa stavki sprinta, preuzimanje odgovornosti za preuzeti dio posla, dijeljenje znanja

(izrada autora)

Kao što se jasno može pročitati iz tablice, teorija i praksa u području koncepta Scrum tima dosta se poklapaju. Važno je napomenuti kako svi ovi koncepti ovise o projektima, a gore u tablici su navedene informacije koje vrijede u većini slučajeva.

Još jedan koncept koji je vrlo važan kada govorimo o Scrum timu je njihova razina multifunkcionalnosti i autonomnosti. To naravno varira od tima do tima. Što se tiče multifunkcionalnosti ona je uvijek prisutna upravo zbog prirode posla. Što se tiče autonomnosti, postoje određeni timovi u kojima ima puno iskusnih programera koji nastoje raspodijeliti posao manje iskusnim programerima te im pomoći u njihovom profesionalnom razvoju. Takvi timovi tek započinju svoj put prema autonomnosti. S druge strane, postoje timovi koji već godinama rade zajedno te su time na istoj razini domenskog i tehnološkog znanja. U takvim timovima svi programeri kao pojedinci preuzimaju posao za koji su sigurni da mogu samostalno odraditi u tom sprintu. Upravo je to ono prema čemu teže sljedbenici Scruma.

Scrum događaji

Tablica 3: Usporedba teorije i prakse u Scrum događajima

	TEORIJA	**PRAKSA**
SCRUM DOGAĐAJI KOJI SE PROVODE U TOKU JEDNOG SPRINTA	Planiranje sprinta, dnevni sastanci, pregled sprinta, retrospektiva, ažuriranje popisa stavki.	Planiranje sprinta, retrospektiva, pregled sprinta, dnevni sastanci, ažuriranje popisa stavki, jedan na jedan sastanci s voditeljima tehnoloških timova, interni sastanci tehnoloških timova.
TRAJANJE SPRINTA	2 – 4 tjedna	2 tjedna
TKO PRISUSTVUJE NA DOGAĐAJIMA	Na svim sastancima prisustvuje cijeli Scrum tim, uz iznimku: • dnevnih sastanaka na koje Scrum master i vlasnik proizvoda nisu dužni doći • pregledu sprinta na kojem se timu pridružuju i zainteresirane strane	Na svim sastancima prisustvuje cijeli Scrum tim, uz iznimku: • dnevnih sastanaka na koje Scrum master i vlasnik proizvoda nisu dužni doći • pregledu sprinta na kojem se timu pridružuju i zainteresirane strane (klijenti)

(izrada autora)

Ponovno vidimo da se teorija i praksa dosta poklapaju. Iako su u tablici na strani prakse navedeni svi Scrum događaji, ponekad se dogodi da se za određene događaje jednostavno nema

vremena. Iz Agencije ističu kako je događaj koji najčešće preskoče retrospektiva. Veliku ulogu u tome igra stabilnost tima. Timovi koji duže rade zajedno, bez promjena članova tima imaju tendenciju lakše se usuglasiti oko toga što je potrebno za kolektivni napredak.

Još jedan od dodataka jesu i vlastite ceremonije timova kao što su jedan na jedan sastanci s voditeljima tehnoloških timova, interni sastanci tehnoloških timova i slični. Takvi sastanci služe kako bi se utvrdilo da su svi članovi na jednakoj razini i kako bi se razjasnilo ili raspravilo o važnim elementima.

Još jedna specifičnost u Agenciji jest da se događaji ažuriranja popisa stavki planiranju unaprijed. Određuje se točan datum i tko sve treba biti prisutan na navedenom sastanku. Tokom sastanka se razjašnjavaju i detaljnije opisuju stavke koje će biti uvrštene u neki od budućih sprintova te se rade procjene.

Scrum artefakti

Tablica 4: Usporedba teorije i prakse u Scrum artefaktima

	TEORIJA	**PRAKSA**
DEFINICIJA GOTOVOG	Na razini timova ili na razini organizacije	Na razini tima
POPIS STAVKI PROIZVODA	Nove funkcionalnosti, popravci grešaka, zadaci i poslovni zahtjevi koji su potrebni za razvoj finalnog softvera, cilj proizvoda.	Korisničke priče, zadaci, epovi, nefunkcionalni zahtjevi, dokumentacija, greške, funkcionalnosti.
ODABIR STAVKI ZA POPIS STAVKI SPRINTA	Sastavlja razvojni tim kako bi prikazali posao koji planiraju obaviti tijekom sprinta sa svrhom postizanja cilj sprinta, a sadrži zadatke, popravke i rad koji će se odraditi u tom sprintu te plan za isporuku inkrementa.	Bira ih razvojni tim, a pod njih spada sve što je preostalo iz prošlog sprinta te stavke iz popisa stavki proizvoda koje imaju najveći prioritet, a dobro su opisani.
CILJ PROIZVODA	Dugoročni cilj Scrum tima koji se bilježi u popis stavki proizvoda.	Cilj koji Scrum tim želi postići izradom proizvoda. Određuje se prije nego projekt započne, a mijenja se ovisno o fazi u kojoj se nalazi projekt. Bilježi se u zajednički

		dokument ili u popis stavki proizvoda.
CILJ SPRINTA	Cilj koji opisuje na koji način proizvod može povećati svoju vrijednost.	Opisuje kako tim namjerava postići povećanje vrijednosti proizvoda tokom sprinta. Definira se na planiranju sprinta i bilježi se u zajednički dokument ili u popis stavki sprinta.

(izrada autora)

Veliku ulogu kod toga kako se teorija Scrum artefakta provodi u praksi igra već spomenuti prepreka, a to je da se radi o agenciji. Razilaženje teorije i prakse najbolje se vidi na definiciji gotovog. Za ovu IT agenciju „Gotovo“ znači da su oni napravili sve što je u njihovoj moći (sve što je potrebno „s njihove strane“) kako bi isporučili klijentu gotov inkrement koji se slaže s dogovorenim kriterijima za prihvaćanje. Taj inkrement je gotov u očima Scrum tima u agenciji no u trenutku kada se on preda dalje, klijenti najčešće preferiraju to još testirati i provjeriti sami. Isto tako, po definiciji, inkrement koji je gotov u sprintu trebao bi se odmah moći izdati i dati krajnjim korisnicima na korištenje, no često to nije slučaj jer klijenti imaju određena ograničenja i nekada im ne odgovara da se aplikacija izdaje na kraju svakog sprinta.

Kao što je vidljivo velik dio teorije Scruma poklapa se s praksom u današnjim IT kompanijama. Sva velika vidljiva odstupanja u primjeru ove IT tvrtke najčešće se događaju zbog činjenice da se radi o agenciji koja radi s klijentima te na neke stvari jednostavno nema utjecaj. Oni su uzeli Scrum okvir i prilagodili ga sebi i svojim potrebama kako bi osigurali isporuku najveće vrijednosti i ispunili jedinstveni cilj kompanije, a to je zadovoljiti potrebe klijenta.

7. Zaključak

Spontani sastanak mladih inženjera 2001. godine pokrenuo je nešto revolucionarno za IT industriju. Agilne metode i njihove prednosti olakšavaju upravljanje projektima razvoja IT usluga već 20-ak godina te se svakog dana razvijaju novije, bolje i praktičnije metode.

Sloboda koju pruža Scrum, IT kompanije znaju dobro iskoristiti. Iako se na prvu možda ne čini da se teorija propisana u poznatom Vodiču kroz Scrum u potpunosti prenosi u praksu to zapravo nije tako. Temelj tog vodiča je pokazati slobodu Scruma i činjenicu da se ne radi o „kuharici“ u kojoj morate specifično pratiti svaki korak kako bi uspjeli, već ga možete prilagoditi svojim potrebama i eksperimentirati. S druge strane Vodič također osigurava da ako u bilo kojem trenutku naiđete na prepreku Scrum okvir je ovdje da vas vrati na „pravi put“.

Navedeno smo mogli vidjeti i iz slučaja poznate IT kompanije. Oni su se kroz godine rada susreli s raznim preprekama koje im je pružalo dinamično tržište razvoja IT usluga te su nastojali pronaći rješenje za svoje probleme. Susreli su se sa Scrum okvirom i njegovom teorijom opisanom u Vodiču kroz Scrum te su tu teoriju nastojali prenijeti u praksu. U početku nisu bili svjesni koliku će prepreku u njihovom putu prema savršenom Scrum poslovanju predstavljati činjenica da se radi o IT agenciji, no to ih nije obeshrabrilo. Oni su odlučili vjerno pratiti teoriju Scruma i njegovih glavnih obilježja (timova, događaja i artefakata), ali istovremeno iskoristiti slobodu i fleksibilnost Scrum okvira koja im omogućava da navedena obilježja prilagode svojem slučaju. Na temelju toga možemo zaključiti kako se Scrum teorija zaista jednostavno i praktično prenosi u praksu, ali i da sloboda Scruma osigurava da svaka kompanija prilagodi Scrum svojim potrebama u svrhu razvoja proizvoda najveće kvalitete.

Razvojem ovog područja paralelno su se razvijali i razni alati koji pomažu timovima da dostignu ciljeve agilnosti, a primjer toga je i Jira kao jednostavan i intuitivan alat koji omogućava svojim korisnicima da vode projekte prema svojim potrebama i željama.

Sve u svemu, možemo zaključiti kako je područje upravljanja projektima razvoja IT usluga svakim danom sve veće i snažnije. Tržište zahtjeva jako puno od IT kompanija, a kako bi im se put prema uspjehu olakšao svakodnevno se razvijaju razne metode i tehnike koje im u tome pomažu. Agilne metode i njihova fleksibilnost i sloboda igraju veliku ulogu u napretku IT kompanija, a samo one kompanije koje su spremne eksperimentirati i uzeti ono najbolje što

pružaju pojedine agilne metode imaju nadu za opstanak na promjenjivom i brzorastućem tržištu razvoja IT usluga.

Literatura

3 Pillar Global (2021). *What is a software product?.* Preuzeto 20.05.2022. s https://www.3pillarglobal.com/insights/what-is-a-software-product/

Abrahamsson P., Ebert C., Oza N. (2012). *Lean Software Development.* IEEE Software, 25 (5), 22-25, DOI: 10.1109/MS.2012.116

Agile Alliance (bez dat). *Agile 101.* Preuzeto 20.05.2022. s https://www.agilealliance.org/agile101/

Ahmad M. O., Markkula J., Ovio M. (2013). *Kanban software development: A systematic literature review.* Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference. DOI: 10.1109/SEAA.2013.28

Alblas J . (2018). *Scrum from the Trenches - Product Backlog Refinement is a Scrum Team Responsibility.* Preuzeto 01.07.2022. s https://www.Scrum.org/resources/blog/Scrum-trenches-product-backlog-refinement-Scrum-team-responsibility

Alblas J. (2019). *Equality - accountabilities in Scrum.* Preuzeto 30.06.2022. s https://www.Scrum.org/resources/blog/equality-accountabilities-Scrum

Altvater A. (2017). *What is Agile Methodology? How It Works, Best Practices, Tools.* Preuzeto 28.06.2022. s https://stackify.com/agile-methodology/

Andrejić D. Marko, Đorović D. Boban, Pamučar D. Dragan (2011). *Upravljanje projektima po pristupu projekt menadžmenta*. Vojnotehnički glasnik, 59(2), 142-157

Arbulu R., Ballard G., Harper N. (2003). *Kanban in construction.* 11th Annual Conference of the International Group for Lean Construction, Virginia, SAD

Atlassian (bez dat). *What is Jira used for?* Preuzeto 20.07.2022. s https://www.atlassian.com/software/jira/guides/use-cases/what-is-jira-used-for

Brown L. (2021). *The Project Management Life Cycle, and Its 5 Phases.* Preuzeto 21.02.2022. s https://www.invensislearning.com/blog/5-phases-project-management-lifecycle/

Carr K. (2022.) *Agile Project Management Vs. Traditional Project Management.* Preuzeto 24.06.2022. s https://www.knowledgehut.com/blog/agile/agile-project-management-vs-traditional-project-management

Corona E., Pani F. E. (2013). *A Review of Lean-Kanban Approaches in the Software Development.* WSEAS Transactions on Information Science and Applications, 10(1), 1-13

Čubranić D., Kaluža M., Novak J. (2013). *Standardne metode u funkciji razvoja softvera u Republici Hrvatskoj.* Zbornik Veleučilišta u Rijeci, 1(1), 239-256

Digital.ai (bez dat). *15th Annual State Of Agile Report,* Preuzeto 20.07.2022. s https://digital.ai/resource-center/analyst-reports/state-of-agile-report

Drumond C. (bez dat). *Is the Agile Manifesto still a thing?* Preuzeto 20.05.2022. s https://www.atlassian.com/agile/manifesto

Đumić M. (2021). *Analiza primene agilnog pristupa projektnom menadžmentu u savremenom poslovanju*, Zbornik radova Fakulteta tehničkih nauka, 36(12), 2161-2164, DOI: https://doi.org/10.24867/15GI07Djumic

Fertalj K., Car Ž., Nižetić Kosović I. (2016). *Upravljanje projektima-skripta* (Skripta). FER, Zagreb

Fowler M. (2013). *ExtremeProgramming.* Preuzeto 28.06.2022. s https://martinfowler.com/bliki/ExtremeProgramming.html

Gartner Glossary (bez dat). *IT Services.* Preuzeto 19.05.2022. s https://www.gartner.com/en/information-technology/glossary/it-services

Gonçalves L. (2018). *Scrum: The methodology to become more agile.* Controlling & Management Review, 62(4), 40-42, DOI: 10.1007/s12176-018-0020-3

Hales L. (2021). *Pros & Cons of Agile Development Mehods.* Preuzeto 28.06.2022. s https://study.com/academy/lesson/pros-cons-of-agile-development-methods.html

Harris C. (bez dat). *Agile Scrum artifacts.* Preuzeto 01.07.2022. s https://www.atlassian.com/agile/Scrum/artifacts

Heale R., Twycross A. (2018). *What is a case study?.* Evid Based Nurs, 21(1), 7-8, DOI: http://dx.doi.org/10.1136/eb-2017-102845

Highsmith J. (2009). *Agile Project Management: Creating Innovative Products.* Pearson Education Inc., SAD

Indeed (2021). *13 Types of IT Services: What They Are and How They Help.* Preuzeto 19.05.2022. s https://www.indeed.com/career-advice/career-development/examples-of-it-services

Kientiz P. (2017). *The pros and cons of Waterfall Software Development.* Preuzeto 01.07.2022. s https://www.dcsl.com/pros-cons-waterfall-software-development/

Lapham, M.A. , Williams R., Hammons (Bud) C., Burton D., Schenker A. (2010). *Considerations for Using Agile in DoD Acquisition.* Software Engineering Institute, Carnegie Mellon University, DOI: 10.1184/R1/6585611.v1

Livermore J. A. (2007). *Factors that Impact Implementing an Agile Software Development Methodology.* Proceedings 2007 IEEE SoutheastCon, 82-86, DOI: 10.1109/SECON.2007.342860.

Lutkevich B. (2021). *Lean software development.* Preuzeto 29.06.2022. s https://www.techtarget.com/searchsoftwarequality/definition/lean-programming

Majstorović A., Majstorović V. (2019). *Metodologije i trendovi u području upravljanja projektima.,* Fakultet strojarstva, računarstva i elektrotehnike Sveučilišta u Mostaru, Bosna i Hercegovina

„Manifesto for Agile Software Development“ (2001). Preuzeto 28.06.2022. s https://agilemanifesto.org/

Marsden G., Maunder A., Parker M. (2008.) *People are people, but technology.* Philosophical Transactions of The Royal Society A Mathematical Physical and Engineering Sciences 366(1881), 3795-804, DOI: 10.1098/rsta.2008.0119

Milošević D. (2018). *Agilna metodologija.* Preuzeto 28.06.2022. s https://dusanmilosevic.com/agilna-metodologija/

Mordi A., Schoop M. (2020). *Making it tangible – creating a definition of agile mindset*. Twenty-Eigth European Conference on Information Systems (ECIS2020), Marrakesh, Morocco

Palmquist S., Lapham M. A., Gracia-Miller S., Chick A. T. (2013). *Parallel Worlds: Agile and Waterfall Differences and Similarities*., Software Engineering Institute, Carnegie Mellon University, DOI: 10.1184/R1/6576047.v1

Parsons D., Thorn R., Inkila M., MacCallum K. (2018). *Using Trello to Support Agile and Lean Learning with Scrum and Kanban in Teacher Professional Development.* 2018 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), 720-724, DOI: 10.1109/TALE.2018.8615399

PMI: Project Management Institute (2011). *Vodič kroz znanje o upravljanju projektima: (vodič kroz PMBOK)*. Zagreb : Mate

Pollack, J., Helm, J. & Adler, D. (2018). *What is the Iron Triangle, and how has it changed?.* International Journal of Managing Projects in Business, 11(2), 527-547. DOI:10.1108/IJMPB-09-2017-0107

Quiambao L. (2022). *How to Write a Project Plan in 8 Easy Steps.* Preuzeto 01.07.2022 s https://www.wrike.com/blog/how-to-write-a-project-plan-easy-steps/

Radoš A. (2021). *Izazovi upravljanja projektima razvoja softvera primjenom Scrum okvira za agilno upravljanje projektima* (Diplomski rad), Sveučilište u Zagrebu, Ekonomski fakultet

Rehkopf M. (bez dat). *Scrum Sprints*. Preuzeto 30.06.2022. s https://www.atlassian.com/agile/Scrum/sprints

Rožman P. (2021). *Faze upravljanja projektom* (Završni rad). Istarsko veleučilište - Universita Istriana di scienze applicate, Pula

Salameh H. (2014). *What, When, Why, and How? A Comparison between Agile Project Management and Traditional Project Management Methods.* International Journal of Management Reviews, 2(5), 52-74

Salazar Caraballo A. L. (2018). *From iron triangle to agile triangle.* Preuzeto 24.06.2022. s https://www.linkedin.com/pulse/from-iron-triangle-agile-extended-luis-antonio-salazar-caraballo

Sarangam A. (2022). *8 Important Types Of Agile Methodology.* Preuzeto 28.06.2022. s https://www.jigsawacademy.com/blogs/product-management/types-of-agile-methodology/

Schwaber K., Sutherland J. (2020). *Vodič kroz Scrum.* Preuzeto 24.06.2022. s https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Croatian.pdf

Sharma S., Sarkar D., Gupta D. (2012). *Agile Processes and Methodologies: A Conceptual Study.* International Journal on Computer Science and Engineering, 4(5)

Sinković G., Bevanda V. (2007). *Značajke kvalitete softverskog proizvoda.* Economic research - Ekonomska istraživanja, 20(2), 35-46

Stephan F. (2015.) *Is Agile The Answer?.* Preuzeto 24.06.2022. s https://www.kaizenko.com/top-4-benefits-of-agile/

Vresk A., Pihir I., Tomičić Furjan M. (2020). *Agilne vs tradicionalne metode za upravljanje IT projektima – studija slučaja.* Preuzeto 24.06.2022. s http://archive.ceciis.foi.hr/app/public/conferences/2020/Proceedings/Croatian/Cro4.pdf

Wells D. (1999). *The Rules of Extreme Programming.* Preuzeto 28.06.2022. s http://www.extremeprogramming.org/rules.html

Wells D. (2013). *Extreme Programming: A gentle introduction.* Preuzeto 28.06.2022. s http://www.extremeprogramming.org/

What is the Software Development Life Cycle (SDLC) (2013). Preuzeto 26.03.2022. s https://airbrake.io/blog/sdlc/what-is-the-software-development-life-cycle

Wysocki R. K. (2006). *Effective Software Project Management,* Indianapolis. SAD: Wiley Publishing Inc.

Yin, R. K. (2004). *Case Study Research: Design and Method*s. Thousand Oaks, CA, USA: Sage Publications.

Popis slika

Popis tablica

Prilog 1

OBRAZAC ZA PRISTANAK NA SUDJELOVANJE U ISTRAŽIVANJU

Ovaj obrazac sastoji se od dva dijela:

- Informativni list (osnovne informacije o istraživanju i osobama odgovornim za istraživanje)
- Potvrda o pristanku (informacije o elementima istraživanja na koje pristajete ako se odlučite sudjelovati u istraživanju)

INFORMATIVNI LIST

Istraživanje: primjena agilnih metoda, posebice Scrum okvira, u praksi

Autorica: Mia Maran, studentica treće godine preddiplomskog studija Poslovni sustavi na Fakultetu organizacije i informatike u Varaždinu, Sveučilišta u Zagrebu

Email autorice: mmaran@foi.hr

Mentorica: Doc. dr. sc. Katarina Pažur Aničić, docentica na Fakultetu organizacije i informatike u Varaždinu, Sveučilišta u Zagrebu

Email mentorice: kpazur@foi.unizg.hr

Organizacija: Fakultet organizacije i informatike u Varaždinu, Sveučilište u Zagrebu

Adresa organizacije: Pavlinska 2, 4200 Varaždin, Hrvatska

U slučaju da u bilo kojem trenutku imate bilo kakva pitanja, komentare ili nedoumice vezane za ovo istraživanje, molim Vas da se javite na jedan od gore navedenih e-mail-ova.

O istraživanju

Istraživanje se provodi u svrhu izrade studije slučaja u sklopu završnog rada autorice Mie Maran. Naziv završnog rada je „Korištenje agilnih metoda vođenja projekata u razvoju IT usluga“. Cilj istraživanja je uvidjeti kako se teorija na kojoj se temelje agilne metode, a posebice Scrum, provode u praksi.

Procedura i proces intervjuiranja

U slučaju Vašeg pristanka na sudjelovanje u ovom istraživanju dužni ste potpisati ovaj obrazac. Nakon potpisa obrasca dobit ćete listu pitanja koja će se pojaviti na intervjuu.

Intervju je polustrukturiranog tipa što znači da će se navedena pitanja sigurno postaviti na intervjuu no moguća je pojava i dodatnih potpitanja autorice kako bi se određene teme detaljnije razjasnile.

Ako u bilo kojem trenutku na pitanje ne želite ili ne možete odgovoriti imate pravo zatražiti da se to pitanje preskoči.

Cijeli intervju bit će sniman. Snimka će biti dostupna samo autorici te će biti korištena isključivo u svrhe daljnje obrade vaših odgovora. Odgovori će u finalnom radu biti u potpunosti anonimni.

Pristankom na sudjelovanje u ovom istraživanju od vas se očekuju iskreni i što je moguće više objektivni odgovori, osim u slučajevima kada se traži vaše subjektivno mišljenje.

Publiciranje

Rezultati istraživanja bit će objavljeni kao dio studije slučaja u sklopu završnog rada autorice Mie Maran.

Po završetku istraživanja te nakon obrade podataka od strane autorice, finalni će rezultati istraživanja biti dostupni svim sudionicima istraživanja. Svaki sudionik ima pravo provjeriti istinitost informacija te zatražiti dodavanje ili micanje određenih dijelova.

Sudjelovanje i izbor sudionika:

Tvrtka odabrana za ovo istraživanje odabrana je na temelju sljedećih kriterija:

- Tvrtka koja se bavi razvojem softverskih proizvoda
- Koristi agilne metode (uključujući i Scrum)

Zaposlenici tvrtke koji će sudjelovati u istraživanju čine to isključivo na dobrovoljnoj osnovi. Svaki sudionik ima pravo odustati u bilo kojem trenutku.

POTVRDA O PRISTANKU

Datum i mjesto: _____

Izjava sudionika

Pozvan sam da sudjelujem u istraživanju primjene agilnih metoda u praksi za potrebe izrade završnog rada na temu „Korištenje agilnih metoda vođenja projekata u razvoju IT usluga“ autorice Mie Maran.

Pristajem odgovoriti na pitanja postavljena od strane autorice. Nastojati ću na sva pitanja odgovoriti objektivno, osim u slučaju kada se od mene traži subjektivno mišljenje. Pristajem da se intervju snima i da se snimka koristi u svrhu daljnje obrade mojih odgovora.

Pročitao/la sam sve gore navedene podatke te dobrovoljno pristajem biti dio ovog istraživanja.

Ime i prezime sudionika:

_____

Vlastoručni potpis sudionika:

_____

Izjava autorice

Pružila sam točne podatke vezane uz ovo istraživanje, a vidljive na ovom listu. Nastojala sam objasniti cijeli proces istraživanja te omogućila sudionicima načine za pružanje podrške (odgovaranje na pitanja, komentare i nedoumice) tokom trajanja ovog istraživanja. Potvrđujem da će sudionik dobiti točne i iskrene informacije kao odgovore na sva njegova pitanja.

Potvrđujem da sudionik ni na koji način nije prisiljen na sudjelovanje u ovom istraživanju nego je ono zasnovano isključivo na njegovoj dobroj volji.

Kopija ovog obrasca dostavljena je sudioniku.

Ime i prezime autora:

_____

Vlastoručni potpis autora:

_____

Prilog 2

PITANJA ZA PROVOĐENJE INTERVJUA

Poštovani/a,

za početak Vam se želim zahvaliti na sudjelovanju u ovom istraživanju.

Moje ime je Mia Maran i studentica sam Fakulteta organizacije i informatike u Varaždinu, Sveučilišta u Zagrebu na trećoj godini preddiplomskog studija Poslovni sustavi. Ovaj intervju provodim sa svrhom izrade studije slučaja koja čini dio mog završnog rada na temu „Korištenje agilnih metoda vođenja projekata u razvoju IT usluga“. Cilj mi je ovim intervjuom dobiti uvid kako agilne metode, a posebice Scrum, funkcioniraju u praksi kako bih to u mogla usporediti s teorijom.

Prije početka ovog intervjua dobiti će te „Obrazac za pristanak na sudjelovanje u istraživanju“ kojim pristajete na korištenje vaših odgovora u svrhu daljnje obrade i izrade navedenog završnog rada. Svi će odgovori biti u potpunosti anonimni. Obrascem također pristajete na snimanje intervjua. Snimka će biti korištena isključivo u svrhe skraćivanja trajanja ovog intervjua te u svrhe kasnije lakše obrade vaših odgovora. Očekivano trajanje intervjua je 45 minuta.

Pitanja će biti podijeljena u 3 sekcije. Prva sekcija – „Općenito“ odnosi se na općenite podatke o vašoj poziciji, stažu i slično. Sljedeća sekcija – „Agilne metode“ odnosi se na pitanja vezana za vaše iskustvo s agilnim metodama, dok se zadnja sekcija – „Scrum“ odnosi na vaša iskustva sa Scrum okvirom.

Ako u bilo kojem trenutku ne želite ili ne možete odgovoriti na pitanje imate pravo odbiti odgovoriti. U nastavku možete vidjeti pregled pitanja koja vas očekuju, kako biste bili upoznati s tijekom intervjua i potrebnim informacijama.

Još se jednom zahvaljujem na sudjelovanju i uloženom vremenu!

S poštovanjem,

Mia Maran

Pitanja za intervju na temu primjene agilnih metoda i Scrum okvira u praksi

Općenito

1. Koja je vaša poslovna funkcija?
2. Koliko radnog iskustva imate na toj ili sličnoj poziciji?
3. Imate li diplome/certifikate za područje/funkciju kojom se bavite?

Agilne metode

1. Tko je zadužen za vođenje projekata u vašoj agenciji?
2. Koristite li agilne metode ili tradicionalne metode u vođenju projekata?
3. Ako koristite agilne metode, jeste li prije koristili neku tradicionalnu metodu? Kako ste prešli s tradicionalnog na agilno?
4. Koje su po vama prednosti agilnog vođenja projekata? Koji su izazovi s kojima se susrećete?
5. Koje alate koristite za upravljanje projektima?
6. Kako pratite napredak projekta?

Scrum

1. Koliko dugo koristite Scrum?
2. Koliko se projekata trenutno provodi koristeći Scrum okvir?
3. Što je vaš dodatak u Scrum okvir? Što vi koristite, a nije propisano u Vodiču kroz Scrum?

Scrum tim

1. Od koliko se članova sastoji scrum tim?
2. Od kojih se uloga sastoji scrum tim?
3. Koja je odgovornost scrum mastera?
4. Koja je odgovornost vlasnika proizvoda (engl. Product Owner)?
5. Tko čini razvojni tim (engl. Developers)?
6. Koja je odgovornost razvojnog tima?
7. Kako biste procijenili razinu krosfunkcinonalnosti i samoorganiziranosti članova tima?

Scrum događaji

1. Koje Scrum događaje provodite?
2. Koliko traje tipičan Sprint?
3. Imate li dodatnih događaja uz one propisane u Vodiču kroz Scrum? Ako da, koje?
4. Jeste li ikada bili primorani otkazati Sprint? Ako da, zašto?
5. Tko prisustvuje na:
    a. Planiranju sprinta (engl. Sprint planning)
    b. Dnevnim sastancima (engl. Daily Scrum)
    c. Pregledu sprinta (engl. Sprint Review)
    d. Retrospektivi (engl. Sprint Retrospective)
    e. Ažuriranju popisa stavki (engl. Backlog Refinement)

Scrum artefakti

1. Imate li definiciju gotovog (engl. Definition of Done) na razini organizacije ili na razini timova?
2. Što za Vas znači „Gotovo“ na nekom projektu?
3. Što čini popis stavki proizvoda (engl. Product Backlog)?
4. Kako odabirete koje će stavke ići u popis stavki sprinta (engl. Sprint backlog)?
5. Kako se rade procjene?
6. Kada i kako se određuje cilj proizvoda?