Vođenje projekata -tradicionalni i suvremeni pristupi

Matulić, Lucija

Master's thesis / Diplomski rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Organization and Informatics / Sveučilište u Zagrebu, Fakultet organizacije i informatike**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:211:755803

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Lucija Matulić

VOĐENJE PROJEKATA – TRADICIONALNI I SUVREMENI PRISTUPI

DIPLOMSKI RAD

Varaždin, 2019.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Lucija Matulić

Matični broj: 0016104947

Studij: Organizacija poslovnih sustava

VOĐENJE PROJEKATA – TRADICIONALNI I SUVREMENI PRISTUPI

DIPLOMSKI RAD

Mentor:

Prof. dr. sc. Robert Fabac

Varaždin, rujan 2019.

Lucija Matulić

Izjava o izvornosti

Izjavljujem da je moj završni/diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autorica potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

_____

Sažetak

Projekti i njihovo kvalitetno upravljanje poprimaju sve važniju ulogu u organizacijama svih veličina i grana industrije. Zbog sve bržih procesa i složenosti poslovnih sustava, poduzeća su primorana odmaknuti se od tradicionalnih metoda vođenja projekata te se preusmjeriti na suvremene - agilne metode. U radu se prikazuju glavne odrednice tradicionalnih i suvremenih metoda vođenja projekata, razlike te prednosti i mane. S obzirom na sve veću popularnost agilnog pristupa, razvijen je velik broj alata za upravljanje projektima koristeći suvremene metode. Na kraju se predstavlja ciklus i način upravljanja projektom pri izradi mobilne aplikacije u alatu Jira primjenjujući SCRUM metodu.

Ključne riječi: projekt, vođenje projekata, tradicionalne metode, agilne metode, *waterfall*, SCRUM

Sadržaj

1. Uvod

Projekti i njihovo kvalitetno vođenje postaju sve značajniji u poduzećima gotovo svih grana i veličina industrije. Zbog naglih i intenzivnih tehnoloških promjena u svijetu, pa tako i poslovanju, počela se razvijati sve veća potreba za kvalitetnim upravljanjem tako velikog broja kompleksnih poslovnih procesa. Kao relativno nova branša, vođenje projekata počinje se više razvijati i dobivati sve veću pažnju; ponekad nazivajući je i najvažnijim zanimanjem 21. stoljeća. Velik značaj projektnog upravljanja jest u korištenju znanja, vještina, tehnika i pravila kojima se omogućava nesmetano i efikasno provođenje projektnih zadataka što dovodi do uspješno isporučenih projekata u cjelini.

Nagli tehnološki razvitak u svijetu i poslovanju donio je naravno i sve veći broj informatičkih rješenja i projekata. Ti projekti isto tako postaju sve opsežniji i kompleksniji te je tome razlog zašto vođenje projekata poprima sve veću važnost u informatičkoj industriji. Digitalizacijom, jakim tehnološkim razvojem poslovanja i sve kompleksnijim projektima, tradicionalni oblici i načini vođenja projekata više nisu mogli pratiti koncepte upravljanja u modernom dobu. Stoga su razvijene i sve su popularnije takozvane agilne metodologije upravljanja projektima.

Agilnost se sada sve više primjenjuje u raznim dijelovima poslovanja, od menadžmenta, upravljanja projektima do vođenja tima. Ono što je velika promjena i značaj agilnih metodologija jest da se na prvo mjesto stavlja najvažniji resurs - čovjek, dok su prethodne metode bile su usmjerene na procese te njihovo nesmetano izvršavanje. Zadovoljstvo svih dionika na projektima pokazala se kao puno efikasnija metoda, gdje se fokus okreće na potrebe, zahtjeve i mogućnosti onih koji u projektu sudjeluju.

Kroz rad opisat će se fenomen projekata i njihova vođenja, proći će se kroz najpoznatije tradicionalne metode, njihove prednosti, mane i sveukupne razlike. Nakon toga će se opisati nove, suvremene - agilne metode u vođenju projekata s naglaskom na vođenja projekata pri razvoju softverskih rješenja. Ponovo će se opisati njihove prednosti i mane upravo u okolnostima razvijanja softvera te ih se usporediti po glavnim karakteristikama.

Na kraju će se na temelju stvarnog primjera izrade jednostavne android mobilne aplikacije prikazati SCRUM proces praćenja i rada na izradi aplikacije. Zbog trenutno najveće popularnosti i najboljih recenzija, proces će biti prikazan u alatu Jira.

2. Definiranje projekta i projektnog menadžmenta

Područje upravljanja projektima je standardizirano od strane PMI udruge (eng. *Project Management Institute*). PMI je međunarodna udruga za upravljanje projektima, pružanje potpore i edukacije u projektnom menadžmentu te razvitak i unaprjeđenje projektnog menadžmenta u svijetu. Sjedište joj je u SAD-u, a podružnice udruge nalaze se u više od 200 zemalja svijeta među kojima je i Hrvatska. Članovi PMI udruge su autori vodiča PMBOK (eng. *Project Management Body of Knowledge*) koja predstavlja temeljnu literaturu za izučavanje i primjenu metodologija upravljanja projektima te certifikaciju.

Kako bismo mogli definirati upravljanje projektima potrebno je prije svega definirati pojam projekta, stoga se u ovom poglavlju obrađuju osnovni pojmovi kao i počeci i razvoj projekata i njihova vođenja.

2.1. Projekt

Projekt (od engl. riječi *Project*) dolazi od latinske riječi *projectum* i njezine izvedenice *projicere* koja znači - “baciti nešto naprijed”. Riječ upućuje na nešto što se unaprijed osmišljava i planira te nakon toga djeluje u svrhu ostvarivanja prvotnih ciljeva.

Prema (Project Management Institute, 2008) projekt je privremeni pothvat kojim se stvara jedinstveni proizvod, usluga ili rezultat. Privremena priroda projekta označava da ima točno određen početak i kraj a jedinstvenost se očituje u izradi proizvoda i provedbi projekta koji su svaki puta drugačiji.

Prema (Dujanić, 2010) projekt osim resursa obuhvaća velik broj aktivnosti koje se izvode slijedno i/ili simultano koje je potrebno kvalitetno koordinirati i usklađivati kako bi projekt bio izvršen unutar predviđenog vremenskog perioda, zadanih troškova i izvođačkih smjernica.

Kritični su za realizaciju poslovnih strategija jer se upravo njima postižu ciljevi kojima se sveukupno ostvaruje zacrtani poslovni uspjeh. Oni su ono što predstavlja jaz između postojećeg i željenog stanja. Prema (Project Management Institute, 2008) primjeri pokretanja projekata uključuju ali nisu ograničeni na sljedeće:

- razvoj novog proizvoda ili usluge,
- djelovanje na promjenu unutar strukture, na sastav zaposlenika ili stil neke organizacije
- razvoj ili prihvaćanje novog ili modificiranog informacijskog sustava, izgradnju zgrade ili infrastrukture, ili

- uvođenje novog poslovnog procesa ili procedure.

Glavni cilj projekta je ostvarivanje početno zacrtane ideje. Većina projekata se pokreće za neki trajni ishod, dakle nešto što će postojati i donositi značaj duži niz godina. Prema (Project Management Institute, 2008) projekt može stvoriti:

- proizvod koji može biti komponenta neke druge cjeline, ili konačna cjelina sama po sebi
- sposobnost da se pruži neka usluga (npr. poslovna funkcija kao podrška organizacijskoj jedinici)
- rezultat kao npr. neki ishod ili dokument (npr. istraživački projekti ili procesi koji koriste napretku društva)

Projekti se mogu provoditi na više razina, npr. unutar neke institucije ili organizacije, kroz suradnju nekoliko subjekata, na razini lokalne ili gradske samouprave, na razini države pa i šire. Također, projekt može uključivati jednu osobu ili više stotina ili tisuća ljudi. (Babić, 2018).

Projekti mogu biti dio bilo koje industrije i područja ljudske djelatnosti. Možemo govoriti o projektima za usklađivanje zakona s EU-om, izgradnja željeznice, provođenje reforme školstva, izgradnja objekata, naselja, informacijskih sustava, zbrinjavanje otpada,... ograničenja za projektnu okolinu zaista ne postoje. Ono po čemu se razlikuju je područje u kojem pripadaju, ciljevi, veličina, način financiranja, mjesto izvođenja, učestalosti ponavljanja, stupnju tehnologije itd.

Zaključujemo da projekt ima značenje unaprijed osmišljenog, jedinstvenog poslovnog pothvata koji želimo izvršiti u određenom vremenu na određenom mjestu kako bi uz utrošak materijalnih i nematerijalnih resursa postigli njegove ciljeve. Ima sve elemente poslovnog procesa te se uvijek veže za svoje međusobno zavisne elemente - zadatke, resurse i vrijeme. Velika odlika koja definira projekt jest da je to pothvat koji se odvija s odgovarajućim rizikom i neizvjesnošću, a postupci koji se provode tijekom njegove provedbe određeni su procedurama i pravilima organizacije koja ga pokreće. Za uspješnu provedbu projekta i ostvarenje njegovih ciljeva potreban je velik trud i kvalitetna koordinacija aktivnosti gdje se ističe važnost uloge voditelja projekata.

2.1.1. Odrednice projekata

Iz navedenih definicija možemo zaključiti da se za projekte uvijek vežu slijedeće zajedničke odrednice:

1. Jedan cilj i svrha

Razlog pokretanja projekta može biti neki novi produkt ili usluga koji želimo stvoriti, uvođenje novih tehnologija, promjene u organizacijskoj strukturi i dr. Sve su to ciljevi koje želimo ostvariti i prema kojima usmjeravamo sve operacije i aktivnosti.

2. Vremenska i prostorna određenost

Svaki projekt ima definirano mjesto i vrijeme izvođenja, te vrijeme i mjesto njegova planiranja. Vrlo je važna odrednica vremenske određenosti projekta zbog kvalitetnog raspoređivanja aktivnosti i saznanja rokova te djelovanje temeljeno na istima.

3. Jedinstvenost

Uz vremensko ograničenje i trošenje resursa, jedinstvenost je jedna od važnijih karakteristika projekta. Ishod svakog projekta je produkt novog proizvoda ili usluge, promjene, reorganizacije te bilo koje drugo postizanje i kreiranje nečega što do sada nije postojalo. Rezultat i procesi projekta mogu biti ponavljajući (isti) ali će uvijek postojati univerzalne karakteristike u radu ili rezultatu koje razlikuju svaki zasebni projekt.

4. Zahtjeva i troši resurse

Velik i vrlo važan aspekt svakog projekta su resursi. O njima najviše ovisi uspješnost izvođenja i provedbe projekta. Najvažniji je naravno ljudski resurs, uz financijski i materijalni. Velik dio planiranja fokusira se na određivanje potrebe i raspodjelu resursa kako ne bi u određenom trenutku nedostajali.

5. Fazno djelovanje

Usko je povezano uz vremensku determinaciju i rezultat. U projektu se napreduje kroz faze odnosno korake koji se definiraju. Svaka faza sastoji se od niza aktivnosti koje je potrebno izvršiti kako bi faza bila finalizirana. Bilo da se radi o tradicionalno ili agilno vođenim projektima, on je uvijek razložen na faze. Razlika je u tome što se kod tradicionalnog pristupa faze odvijaju slijedno a kod agilnog iterativno.

6. Ima početak i kraj

Jedna od osnovnih karakteristika projekta je da ima definiran početak i završetak. Kraj projekta postignut je kada su ostvareni svi ciljevi zadani projektom ili kada odgovorne osobe

zaključe kako ciljevi projekta iz nekog razloga neće biti postignuti ili je njihovo ostvarivanje nemoguće.

2.2. Povijesni razvoj Projektnog menadžmenta

Kada gledamo povijesno kroz razvoj čovječanstva, možemo reći da je vođenje projekata oduvijek bilo prisutno. Najpoznatiji primjeri iz ljudske povijesti koji se često spominju za uspješno provedene “projekte” je izgradnja piramida, Panteona ili akvaduktima drevnog Rima. Tada ih naravno nisu tako nazivali, ali je činjenica da su postojali ljudi koji su bili zaduženi za uspješnu izvedbu tih pothvata, sa organizacijskim načelima iz kojih i danas učimo. Za definirani i teorijski određeni projektni menadžment možemo reći da je poprilično nova grana unutar organizacijsko - ekonomske znanosti.

Kao disciplina se počinje razvijati sredinom 20. stoljeća kada su tadašnji stručnjaci za upravljanje projektima osnovali AACE (eng. *The American Association of Cost Engineers).* Polazište je bilo s inženjersko - matematičkog aspekta gdje su koristili razne formule i tehnike za donošenje odluka. Danas ostaje vodeće profesionalno društvo za procjenitelje i inženjere troškova, planere, voditelje projekata i stručnjake za kontrolu projekta (Haughey, 2014).

Moderni projektni menadžment nastao je u SAD-u kada je uspješnost visokobudžetnih projekata (od NASA-e i Američke mornarice) iznimno ovisila o njihovom kvalitetnom vođenju i egzekuciji. Tada je velik naglasak bio na zaduženju i praćenju rokova, troškova i konstantnoj reviziji dosad odrađenih zadataka.

2.2.1. Značajni događaji razvoja Projektnog menadžmenta

U nastavku su navedeni ključni povijesni događaji u evoluciji Projektnog menadžmenta. Događaji koji su doprinijeli razvoju su većinom izrade metoda koje su postale podrška u upravljanju projektima, znanstveni članci, radovi i vodiči koji su definirali standarde i načela te nastanak udruga i profesionalnih organizacija koje se bave temom upravljanja projekata.

1. **Načela znanstvenog upravljanja - 1911. godine**

Frederic Taylor objavio je znanstveni članak Načela znanstvenog upravljanja (eng. *The Principles of Scientific Management*) kojim je opisao tehnike i načela učenja za nekvalificirane radnike u industriji čelika. Cilj je bio definirati jednostavne tehnike učenja uz pomoć kojih bi radnici mogli prijeći na složenije projekte. Uveo je potrebu za sustavima plaća na temelju poticaja i kako iskoristiti tehnike štednje vremena.

2. **Ganttov dijagram - 1917. godine**

Vrlo značajna inovacija koja se veže za nastanak vođenja projekata jest Ganttov dijagram. Njegovog izumitelja, Henrija Gantta, često se naziva i ocem modernog upravljanja projektima koji je 1917. godine razvio je istoimeni dijagram za raspoređivanje i prikaz projektnih zadataka. Upotrijebio je vizualni aspekt postavljanja linija koje su označavale trajanje pojedinog zadatka te se time moglo dočarati i njihov odnos (ako se kojim slučajem preklapaju). Iako se Ganttov dijagram danas ponegdje još uvijek i koristi, za mnoge današnje projekte nije dovoljno razrađen te se uz njega koriste još neki dodatni alati ili ga se nadograđuje i ažurira s obzirom na potrebe specifičnih projekata.

3. **Metoda kritičnog puta *(eng. Critical Path Method)* - 1957.**

Razvijen od strane tvrtke Dupont, CPM je tehnika kojom su htjeli predvidjeti trajanje projekta tako što su analizirali aktivnosti koje su najmanje fleksibilne pri planiranju. Tvrtka ga je osmislila kako bi olakšali vrlo kompleksan proces zatvaranja kemijskih postrojenja radi održavanja te njihovog ponovnog pokretanja. Tehnika je bila toliko uspješna da je korporaciji uštedio milion dolara u prvoj godini implementacije (Haughey, 2014).

4. **PERT metoda - 1958. godine**

Metodu je osmislio Ured za specijalne projekte američke mornarice iz Ministarstva obrane. Potreba za ovom metodom javila se zbog njihovog vrlo velikog i složenog projekta - razvojni programa za svemirski program tj. raketu „Polaris“. Fokus je bio na vremenu potrebnom da se svaki zadatak izvrši i identifikacija minimalno potrebnog vremena za izvršenje kompletnog projekta (Westland, 2018).

5. **Raščlamba strukture rada *(engl. Work Breakdown Structure)* - 1962. godina**

Vrlo poznata i raširena metoda u projektnom menadžmentu koju i danas poznajemo, WBS nastala je strane Ministarstva obrane Sjedinjenih Američkih Država za isti projekt “Polaris” kao i PERT metoda. WBS je cjelovita hijerarhijska struktura rezultata i zadataka potrebnih za dovršavanje projekta. Nakon završetka projekta, Ministarstvo obrane objavilo je WBS i 1962. godine ovlastilo korištenje procedure za buduće projekte (Haughey, 2014).

6. **Nastanak PMI-a - 1969. godine**

Project Management Institution, neprofitni je institut za upravljanje projektima kojeg je osnovalo pet volontera u Pensilvaniji. Za vrijeme svog mandata objavili su vrlo važan vodič za upravljanje projektima - *A Guide to the Project Management Body of Knowledge (PMBOK)* u kojem su detaljno opisani i razrađeni procesi i znanja o upravljanju projektima. PMBOK je 1988. godine postao standard za vođenje projekata koji do danas izlazi u novim izdanjima s ažuriranim sadržajem. Institut je postao i certifikacijsko tijelo, odnosno preko njih se i danas

mogu polagati ispiti i postati certificirani suradnik u upravljanju projektima ili stručnjak u vođenju projekata.

7. SCRUM - 1986. godine

SCRUM proces predstavili su 1993. godine njegovi “osnivači” Jeff Stherland i Ken Schwaber, iako je metoda zapravo nastala 80-ih godina prošlog stoljeća u Japanu. Prema Tridibesh (2003) opisali su inovaciju kao pristup za razvoj proizvoda kojeg su nazvali holistički „rugby“ koncept, gdje jedan tim pokušava prijeći jednu razdaljinu kao cjelina, dodajući loptu nazad i naprijed. Bazirali su svoj koncept na proizvodnji i analizi pojedinih slučaja iz različitih industrija. Zbog jako dobrih tehnika i principa koji su se uvelike razlikovali od tadašnjih metoda u vođenju projekata, SCRUM je danas predstavnik agilnosti i prilagodljivog vođenja projekata. Više o SCRUM-u bit će opisano u nastavku rada.

8. PRINCE (1989.) i PRINCE 2 (1996.)

PRINCE (*engl. Projects in Controlled Environments*), razvila je vlada Velike Britanije kao svoj standard za vođenje projekata informacijskog sustava 1989. godine. Revidiran je 1996. kao PRINCE2 zbog kritike da metoda nije dovoljno prilagodljiva i stoga su pogodna samo za velike projekte. Cilj metode je da se smanje troškovi te da se ne prelaze rokovi zadani za izvršenje pojedinih stadija projekta.

9. Upravljanje projektima kritičnim lancem - 1997. godina

Eliyahu M. Goldratt razvio je *Critical Chain Project Management* (CCPM) koji se temelji na metodama i algoritmima njegove Teorije ograničenja. Održava resurse ravnomjerno opterećenima, a istovremeno ostaje fleksibilan prema njihovim početnim vremenima i prebacuje se između zadataka po potrebi kako bi se projekt držao rasporeda (Haughey, 2014).

10. Agilni manifest - 2001. godina

Zbog sve većeg razvoja sofverskih rješenja a time i upravljanja projektima u navedenom području, skupina softverskih inženjera sastala se kako bi sastavili metode za uspješno provođenje takvih projekata. Agilni principi i metode pokazali su se kao najmodernijima i najefikasnijima za ovo područje, stoga su sastavili Agilni manifest. Neki od autora manifesta osnovali su i *Agile Alliance*, neprofitnu organizaciju koja promiče razvoj softvera u skladu s 12 temeljnih načela manifesta.

2.3. Projektni menadžment

Projektni menadžment podrazumijeva primijenjeno znanje, vještine, alate i tehnike na projektnim aktivnostima kako bi se postigli ciljevi i zahtjevi postavljeni pred projekt od strane interesno - utjecajnih skupina (Omazić i Baljkas, 2005, str. 43).

Prema (Dujanić, 2010, str. 18) projektni menadžment predstavlja aktivnosti usmjerene na izradu i ostvarivanje projekta od strane njegovih naručitelja, a to znači na postizanje unaprijed zacrtanih ciljeva radom ostalih odgovarajućih stručnjaka. Projektni menadžment pribavlja, angažira i kombinira sve *inpute* projektnog procesa da bi uspješno realizirao projektni obuhvat.

Projektni menadžment obuhvaća planiranje, organizaciju, praćenje i kontrolu svih aspekata projekta. S obzirom da su ljudi (dionici) projekta njegov gotovo najvažniji aspekt, ono podrazumijeva i motivaciju, pomoć te olakšavanje provedbe procesa i aktivnosti koji se moraju izvoditi tijekom provedbe projekta. Ključan segment je postizanje ciljeva unutar planiranog budžeta, u planiranom vremenu te na siguran i efikasan način.

Projekti mogu imati pozitivan ili negativan utjecaj na okolinu, a neki od tipova okoline su:

- **Kulturalna/sociološka okolina** – način na koji projekt utječe na ljude i kako ljudi utječu na projekt; sociološki aspekt okoline
- **Međunarodna/politička okolina** – određuju ju ili obilježavaju zakoni, karakteristični običaji, politika, nacionalni praznici itd.
- **Fizikalna okolina** – okoliš, geografski kontekst.

Dionici projekta su interesni sudionici, a to su članovi projektnog tima kao i svi drugi subjekti koji su uključeni u proces razvoja projekta i imaju koristi od njegove uspješne provedbe. Najčešće su to slijedeće osobe:

- Sponzor projekta/investitor – osoba ili grupa koja osigurava (financijske) resurse za projekt
- Projektni tim
- Potporno, prateće osoblje
- Naručitelj, klijent i krajnji korisnici
- Dobavljači, poslovni partneri
- Drugi sudionici, uključujući oponente.

S obzirom da projekt može imati velik broj dionika, sadržavati mnogo faza, trajati ponekad i nekoliko godina, uloga voditelja projekata je da svojim pravilnim nahođenjem prema svim aspektima projekta izvrši zadovoljstvo svih strana. Kada su projekti tako zahtjevni i veliki, postoji potreba za nekolicinom voditelja projekata i njihovim asistentima, te tada to nazivamo projektnim timom.

2.3.1. Uloge voditelja projekata

Projektni menadžer je odgovoran za ostvarenje ciljeva projekata koji imaju direktan utjecaj na ciljeve organizacije; odgovoran je za planiranje i organiziranje posla na projektu, komunikaciju s višim menadžmentom i ostalim projektnim sudionicima, upravljanje aktivnostima te isporučivanje projektnog proizvoda naručitelju u okvirima zacrtane kvalitete, budžeta i vremenskih okvira (Omazić i Baljkas, 2005, str.107).

Voditelj projekata mora imati jasan i detaljan uvid u organizaciju i strukturu u kojoj radi. S obzirom da je izravno odgovoran poduzeću za kojeg radi, mora djelovati prema njihovim ciljevima, načinu rada i kulturi. Mora efikasno koristiti resurse, pratiti rokove, kontrolirati rad, obavještavati vrhovni menadžment i klijente o napretku projekta.

Danas voditelji projekata moraju imati velik set vještina i znanja kako bi mogli efikasno obavljati dužnosti. Ovisno o industriji, metodologiji vođenja i vrsti projekta, projektni menadžer mora baratati s pojmovima, procesima i funkcionalnostima nekolicine različitih područja djelovanja.

U kontekstu voditelja projekata za informacijske sustave, to je osoba koja mora znati aspekte izrade IS-a te barem osnovne tehnologije koje će se u istom koristiti. To je osoba koja ukoliko radi u informatičkoj industriji, mora držati korak s razvojem novih tehnologija što zahtjeva konstantno učenje i sposobnost brzog prilagođavanja. S obzirom da se u većini slučaja danas koriste digitalni alati za praćenje informatičkih projekata, znanje i korištenje tih programa obavezan su dio spektra vještina koji voditelji projekata moraju imati.

Područja kojim svaki voditelj projekata upravlja jesu: ljudskim resursima, obuhvatom projekta, rizikom, vremenom, troškovima, kvalitetom komunikacijom (razmjenom informacija) i nabavom.

3. Tradicionalni ili fazni pristup vođenju projekata

Metodologija je, prema PMI definiciji, sustav praksi, tehnika, procedura i pravila koje rabi onaj tko radi na području određene discipline, gdje je procedura niz koraka koji se odvijaju po redoslijedu da bi se nešto postiglo.

Kerzner (kako citira Špundak, 2006) navodi da su karakteristike dobre metodologije preporučen stupanj detalja, uporaba predložaka, standardizirane tehnike planiranja, vremenskoga određivanja i kontrole troškova, standardizirani oblik izvještavanja, fleksibilnost za primjenu na svim projektima, fleksibilnost za brzi razvoj, razumljivost korisniku, prihvaćenost i uporabljivost u organizaciji, uporaba standardiziranih faza životnog ciklusa te temeljenost na smjernicama (umjesto na procedurama) i na etici dobro obavljenog posla.

Dobra primjena pripadajuće metodologije smanjuje nam rizik neuspjeha, bolji pregled aktivnosti, brži i kvalitetniji proces donošenja odluka, obavljanje zadataka na vrijeme a svime time cjelokupno zadovoljstvo dionika projekta.

Gledajući odrednice i načela, tradicionalni pristup vođenju projekata temelji se na smjernicama opisanim u PMBOK-u. Projekti se sastoje od procesnih faza koje se detaljno opisuju u vodiču: inicijalizacija, planiranje, izvršenje, kontrola i zatvaranje. Faze se ne ponavljaju te se izvršavaju slijedno, jedna za dugom. Važnost slijednosti je u tome što su faze povezane - *output* jedne faze postaje *input* slijedeće.

U poglavlju će se opisati procesne faze tradicionalnog pristupa, životni ciklus projekta, njegovi parametri te će se objasniti vodopadni model koji se temelji na faznom pristupu. Nakon toga će biti objašnjene najpoznatije metode tradicionalnog pristupa, njihove prednosti i nedostaci.

3.1. Životni ciklus projekta

U tradicionalnom svijetu vođenja projekata, projekt prolazi kroz početnu fazu, fazu provedbe te završnu fazu. Kao vremenski ograničen subjekt, određuju ga poput svih organskih sustava, stvari i pojava - nešto što se rađa, živi i umire. U tradicionalnim metodama provedbe projekta, faze i procesi se najčešće izvršavaju slijedno, linearno - bez povratka na prethodnu fazu jednom kada ona biva dovršena.

Prema PMBOK vodiču izdanom od *Project Management Institutiona* kao prihvaćenu vodilju za vođenje projekata, životni ciklus cijelimo na tri faze opisane u slijedećoj tablici.

Tablica 1: Životni ciklus projekta

Faza	Ključni zadaci i odluke	Temeljna pitanja
1. Početna faza - faza dizajniranja	formuliranje vizije i strategije projekta, definiranje ciljeva, modeliranje i planiranje, evaluacija financijskih troškova i koristi, analiza ključnih resursa, budžetiranje	- Što treba uraditi? - Zašto to treba učiniti? - Kako će se to ostvariti? - Tko će što raditi i tko će sve biti uključen u projekt? - Tko će biti sponzor projekta i projektni menadžer? - Kad je početak, a kad završetak projekta? - Koliko će to stajati?
2. Faza provedbe - faza implementacije (provedbe)	prikupljanje tima, organizacija, kontrola, vođenje, donošenje odluka i rješavanje problema, rješavanje konflikata, ugovaranje, provedba, predaja projekta	- Na koji način će se rukovoditi projektom? - Tko će obavljati kontrolu nad projektom? - Hoće li projekt biti završen na vrijeme i u okvirima budžeta?
3. Završna faza (zaključivanje)	procjena procesa i učinkovitosti projekta, evaluacija, prikupljanje i implementacija znanja u sustav, promjene za budućnost	- Kakvi su rezultati ostvareni projektom? - Kako kontinuirano poboljšavati i razvijati projektni menadžment? - Je li korisnik zadovoljan rezultatom?

(Izvor: Dujanić, 2010)

Prema stavkama tablice možemo vidjeti glavne odlike pojedine faze. S obzirom da su vrlo općenito i jednostavno raspoređene aktivnosti, ovaj model može se primijeniti na gotovo svaki projekt. Dobro strukturiranje i podjela projekta na faze je u njegovom provođenju jako važno, no možemo zaključiti zašto su s vremenom metodologije postale specifičnije i razvijenije. Iako ovo kao i procesi koji se provode u tradicionalnim modelima vođenja projekata imaju svoju korist i značaj, nisu dovoljno definirani procesi i današnji kompleksni projekti zahtijevaju bolje razrađene metodologije.

3.2. Parametri tradicionalnih projekta

Glavni parametri tradicionalnih projekta su doseg (opseg), kvaliteta, trošak, vrijeme i resursi. Oni izravno i snažno utječu na postizanje projektnih ciljeva i njegovu uspješnost. (Fertalj i sur., 2016, str. 7).

Doseg projekta je parametar kojim se određuju njegove granice. Definiraju se elementi koji ulaze te elementi koji neće biti dio projekta. Cilj dosega je jasna vizija što projekt treba postići i što od njega očekuje. To je dokument koji služi kao osnova i baza za daljnje provođenje te je stoga iznimno važno da bude napisan točno i nedvosmisleno. Prema (Projektura, 2007)

planiranje dosega možemo promatrati kao pretfazu ispred procesa planiranja, gdje tim razjašnjava sve pretpostavke, ograničenja, objašnjenja, specifikacije, nedoumice, opise radnih procesa te konačne isporuke projekta.

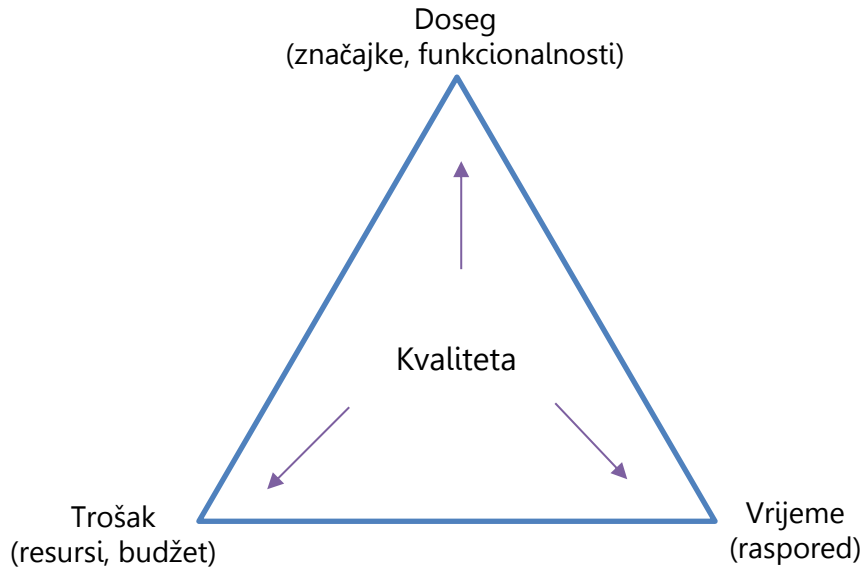
Kvaliteta projekta nešto je čime se teži kako bi se što lakše ostvarili ciljevi. Kvalitetno izvršenje projekta dovodi do kvalitetnog proizvoda a time i zadovoljstva svih dionika. Kada dođe do neočekivanih situacija u kojima je potrebno napraviti izmjene u planu, teži se tome da kvaliteta ne podliježe kompromisu. Prema (Fertalj i sur., 2016) kvaliteta projekta se dijeli na kvalitetu proizvoda i kvalitetu procesa. Kvaliteta proizvoda odnosi se na kvalitetu isporuke projekta a kvaliteta procesa odnosi se na kvalitetu samog procesa upravljanja i njeno poboljšanje. Teži se naravno postizanju što više kvalitete i kohezije između ta dva elementa.

Trošak projekta parametar je koji se odnosi na financiranje projekta. To je budžet s kojim se raspolaže i temeljeno tome se mogu odrediti mogućnosti i planovi. Vrlo važan parametar kod kojeg se uvijek teži da bude manji od očekivane povratne vrijednosti. S koliko novaca projekt raspolaže može utjecati na njegovu kvalitetu i kompleksnost. Ukoliko budžet nije dovoljno velik voditelji projekta primorani su iskoristiti alternativna rješenja ili pronaći način kako da se kompenzira nedostatak.

Vrijeme je u organizacijskim okvirima često nazivan jedan od najvažnijih resursa zato što utječe na sve ostale parametre, ne može ga se vratiti i kada ga se želi dobiti „više“ često generira troškove. Vremenskim rokovima i okvirima određujemo završetke određene faze projekta kao i završetak projekta u cijelosti. Imati ciljeve beskorisno je ukoliko ne znamo do kada želimo da se oni ostvare. Tako je jasniji pregled i plan izvršavanja aktivnosti.

Resursi su sve ono što koristimo (trošimo) u projektu da bi proizveli željeni cilj. Igraju isto jedu od glavnih uloga jer bez resursa nema ni projekta. Najvažniji resurs je ljudski rad, a osim njega može biti i oprema, uređaji, informatički softver, proizvodi, itd.

Od parametara se spominje i rizik jer utječe na sve druge parametre. Na sljedećoj slici prikazan je međuodnos parametara projekta zvan trokut dosega ili željezni trokut. Trokut je tako nazvan zato što, iako se strane mogu skratiti ili produljiti, on i dalje ostaje neraskidiv - upućujući na usku povezanost parametara (Kerzner, 2013).



Slika 1: Trokut dosega (Prema Kerzner, 2013)

Odnos među parametrima prikazan je trostrukim ograničenjem, odnosno način na koji parametri dinamički utječu jedan na drugoga. Važno je uspostaviti ravnotežu među parametrima što znači da treba postizati kvalitetu pravilnim ophođenjem s resursima, pazeći na vrijeme i doseg projekta. Nedostatak ili nemar samo jednog parametra dovodi u disbalans cjelokupne ravnoteže projekta i on tako gubi na svojoj kvaliteti.

3.3. Fazni procesi upravljanja projektima

Prema (Project Management Institute, 2008) upravljanje projektima je primjena znanja, vještina, alata i tehnika na projektne aktivnosti kako bi se zadovoljili projektni zahtjevi. Ova primjena znanja zahtijeva učinkovito upravljanje procesima. U ovom poglavlju definirat će se pojam procesa, razlike u odnosu na ulogu projekta u organizaciji te će se objasniti procesi koji se izvode tijekom vođenja tradicionalnih projekata.

Pojam proces (lat. *processus*: napredak, razvoj) definira se kao zakonomjeran slijed pojava i zbivanja, odnosno put i način na koji se nešto odvija; razvoj nečega. Drugim riječima, proces je skup međusobno povezanih aktivnosti koje se provode s određenom svrhom, da bi se postigao određeni skup rezultata, proizvoda ili usluga. Svaki proces odlikuje se nekim karakteristikama, odnosno ulaznim parametrima, alatima i tehnikama koje se mogu primijeniti te izlaznim parametrima (Fertalj i sur., 2016)

Procesi se u organizacijama izvode svakodnevno na svim razinama, neovisno o projektima. To je aktivnost ili niz aktivnosti kojima se postiže specifični cilj organizacije (u vidu stvaranja proizvoda ili usluge). Za razliku od projekta koji je kratkotrajan (ima početak i kraj),

procesi su dugotrajni – neprekidno se ponavljaju i nemaju određeni vremenski period. Projekt smo definirali kao jedinstven pothvat koji je svaki puta drugačiji, dok su procesi gotovo uvijek isti, određeni standardima i pravilima poduzeća. Tako isto možemo govoriti i za *outpute* – rezultat projekta je uvijek po nečemu drugačiji, poseban i jedinstven, dok je rezultat procesa uvijek isti svaki puta kada se pokrene ciklus.

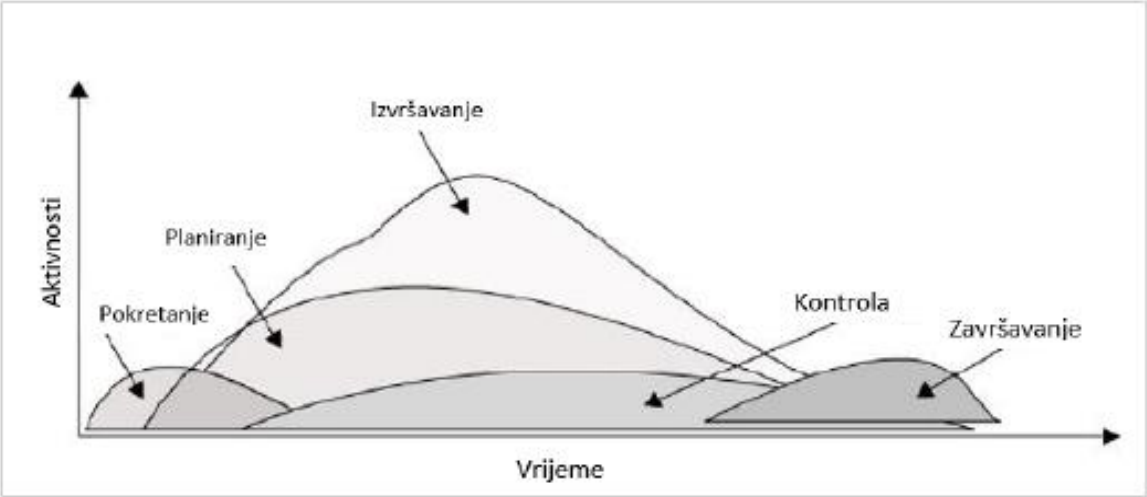
Procesi koji se izvode tijekom provedbe projekata su vrlo važna stavka jer su upravo oni ono čime postižemo slijed i prelazak na nove faze. Voditelj projekata mora biti upoznat sa svim procesima kako bi ih mogao koordinirati, usklađivati te pravilno i pravodobno reagirati u slučaju neočekivanih situacija.

Prema tipu, procesi se mogu podijeliti na sljedeći način (Project Management Institute, 2008):

Procesi orijentirani prema proizvodu (eng. *product-oriented processes*) – specifični za određenu domenu, životni ciklus proizvoda (npr. procesi slijednog modela razvoja programske podrške, ili iterativnog modela razvoja).

Procesi upravljanja projektima (eng. *project management processes*) – procesi koji su zajednički za sve projekte. Poredak i slijed glavnih procesa pri provođenju projekata su slijedeći:

- Procesi pokretanja (inicijalizacija)
- Procesi planiranja
- Procesi izvršavanja
- Procesi kontrole
- Procesi zatvaranja



Slika 2: Procesi u vođenju projekata (Izvor: Phillips, 2004)

Na slici su prikazani projektni procesi s obzirom na vrijeme i aktivnosti. Primjećujemo da su međuovisni te da se preklapaju (osim pokretanja i zatvaranja). Procesi izvršavanja, planiranja i kontrole provode se za vrijeme gotovo čitavog trajanja projekta zato što su to aktivnosti koje se konstantno prilagođavaju, ponavljaju i nadograđuju.

3.3.1. Proces pokretanja (inicijalizacija)

Proces inicijalizacije primarni je i prvotni proces pri odluci o provođenju projekta. Tipični dokument koji nastaje kao *output* ovog procesa je tzv. Prijedlog projekta, gdje se određuju ideje i ciljevi za potencijalno pokretanje nekog projekta. (Buble, 2010, str. 13) definira četiri glavne faze unutar ovoga procesa:

1. Definiranje projekta
2. Formiranje projektnog tima
3. Izrada studije izvodljivosti
4. Selekcija projekta

Definiranjem projekta opisujemo njegove najznačajnije karakteristike - ciljeve i svrhu, provodimo analizu financijske isplativosti, određujemo ulagače i sl. Nakon toga, ukoliko je projekt odobren, formira se tim koji svojim radom doprinosi uspješnosti provedbe projekta. Izrađuje se studija izvodljivosti koja služi kao polazni dokument u kojem se analiziraju troškovi te financijska izvodljivost projekta. Selekcija projekta odnosi se na situaciju u kojoj je predloženo više projekata te se tada odabiru oni koji imaju najveći izgled za uspjeh i postizanje ciljeva organizacije.

Nakon što se počne planirati, izvršavati i već i kontrolirati začeci projekta, proces pokretanja završava. S obzirom da je nešto što se neće mijenjati i projekt je već u punom pogonu, ponavljanjem ovog procesa samo možemo započeti novi projekt, a ne utjecati na postojeći.

3.3.2. Proces planiranja

Grupa procesa planiranja (eng. *planning processes*) definira i precizno iznosi ciljeve (svrhu), planira smjer i akcije koje će se primjenjivati u svrhu ispunjenja ciljeva. Planiranje je od velikog značaja za projekt budući da projekt obično uključuje ono što nije još bilo izvršeno.

Planiranjem detaljno definiramo svaku aktivnost i fazu u projektu, osobe koje će ih izvršavati, detaljne rokove i resurse unutar zadanog vremena s dostupnim budžetom. Ovo je proces koji se treba obavljati s puno pažnje i truda zato što nam služi kao vodič za uspješnu provedbu čitavog projekta. Najmanje greške, nepredviđene situacije i nedefinirani rizici u procesu planiranja mogu dovesti do većih i kobnijih problema kasnije što može rezultirati većim troškovima i nezadovoljstvom sudionika.

Neke od najvažnijih aktivnosti koje se izvršavaju tijekom ovog procesa su definiranje opsega projekta, vrijeme i rokove, predviđanje troškova, procjena potrebnih resursa, upravljanje ljudskim potencijalima, definiranje rizika i vanjsko/unutarnjih utjecaja, logistika te brojne druge.

Tipični dokumenti koji proizlaze iz ovog procesa su izrada projektnog zadatka i izrada liste zadataka (eng. *Work Breakdown Structure*). Dokument projektnog zadatka sadrži ciljeve i zadatke projekta, strukturu i opis posla, organizaciju posla, odgovornosti, resurse, rizike i dr. (Požgaj, bez dat.).

3.3.3. Proces izvršenja

Grupa procesa izvršavanja sastoje se od procesa potrebnih da se projekt dovrši prema predviđenom planu s uspješno ostvarenim projektnim ciljevima. Uključuje koordinaciju ljudskih i materijalnih resursa, izvršavanje projektnih aktivnosti i integraciju aktivnosti u skladu s prethodno definiranim projektnim planom. Zbog toga, ovo je proces na koji se troši najveći dio proračuna.

Procesima izvršavanja uspostavljaju se operativni odnosi između svih dionika neovisno jesu li uključeni u upravljanje ili ne. Kod procesa izvršavanja koriste se sljedeće tehnike i pomagala (Fertalj i sur., 2016):

- Općenita vještina upravljanja

- Znanja i vještine vezane uz domenu projekta
- Sastanci za praćenje statusa projekta
- Informacijski sustav za upravljanje projektom/autorizaciju rada
- Organizacijske procedure – poslovni postupci, temeljem pravila.

Danas veliku ulogu u procesu izvršenja kao potpora imaju informacijski sustavi za kontrolu rada. Drugi najveći dio su znanja i sposobnosti voditelja projekta oko upravljanja ljudima, resursima i procesima. Važna vještina koju voditelji projekata moraju imati jest komunikacijska, kako je balansiranje zadovoljstva između svih sudionika na projektu najvažnija stavka provedbe projekta.

3.3.4. Proces kontrole

Procesi praćenja i kontrole služe da bi mjerili i provjeravali svaku aktivnost zbog mogućih ljudskih greška i odstupanja od planiranih rezultata. Nadziranje podrazumijeva prikupljanje podataka, mjerenje učinka i izvještavanje, a kontrola akcije temeljem nadziranja (regulaciju). Cilj ovog procesa je poduzimanje korektivnih mjera i pravodobno reagiranje na nepravilnosti koje su se dogodile.

Izvedba projekta prati se konstantno i redovito kako bi se na vrijeme prepoznala odstupanja u odnosu na zacrtani plan. Kako bi se izbjeglo ugrožavanje projekta, na primijećena variranja provodi se prilagodba i ponavljanje procesa planiranja. Zbog neke greške ili nepredviđene situacije potrebno je ažurirati plan kako nastaviti izvršenje projekta sa što manje štete.

Neki od dokumenta koji nastaju tijekom ovog procesa su:

- **Izvješća za aktualno razdoblje** - rade se za neposredno završeni period te navode napredak aktivnosti koje su bile otvorene ili planirane u promatranom periodu
- **Kumulativna izvješća** - izvješća koja sadrže povijest od početka projekta do aktualnog razdoblja te donose informacije o trendovima
- **Izvješća o iznimkama** - ukazuju na odstupanja od plana i u pravilu su oblikovana da ih više rukovodstvo može brzo shvatiti
- **Izvješća upozorenja** - oznaka na vrhu prve stranice izvješća koja nastaju tijekom ovog procesa, zelena označuje projekt koji se razvija prema planiranom, žuta kada zaostaje a crvena kada se kontrola gubi
- **Izvješća o odstupanjima** - dojava odstupanja između realizacije u odnosu na plan; najčešće tabličnog oblika

3.3.5. Proces zatvaranja

Procesi završavanja ili zatvaranja formaliziraju prihvaćanje usluga, proizvoda ili drugih rezultata projekta (ili faze projekta) te dovode do završetka projekta, odnosno faze projekta.

Dobivanjem gotovog outputa na kraju projekta označava da je projekt završen. Tada je potrebno dobiti potvrdu klijenta (ili nekog drugog važnog dionika) o završetku, sažeti naučene lekcije te dovršit svu preostalu administrativnu dokumentaciju.

Prema (TenSteps Croatia, 2011) slijedeće aktivnosti su potrebne za zatvaranje projekta:

- Održavanje završnog sastanka
- Proglašenje uspjeha ili neuspjeha
- Tranzicija rješenja u podršku (ako je primjenljivo)
- Predaja datoteka projekta (ako je primjenljivo)
- Izvođenje pregleda performansi
- Ponovo pridruživanje preostalog projektnog tima

3.4. Vodopadni model

Jedan primjer tradicionalne metodologije upravljanja projektima je metodologija vodopada (engl. *Waterfall model*). Popularizirao ju je američki računalni znanstvenik Winston W. Royce u članku „*Managing Development of Large Scale Software"* 1970. godine. Pokušao je postići realizaciju projekata u operativnom stanju, na vrijeme i unutar troškova.

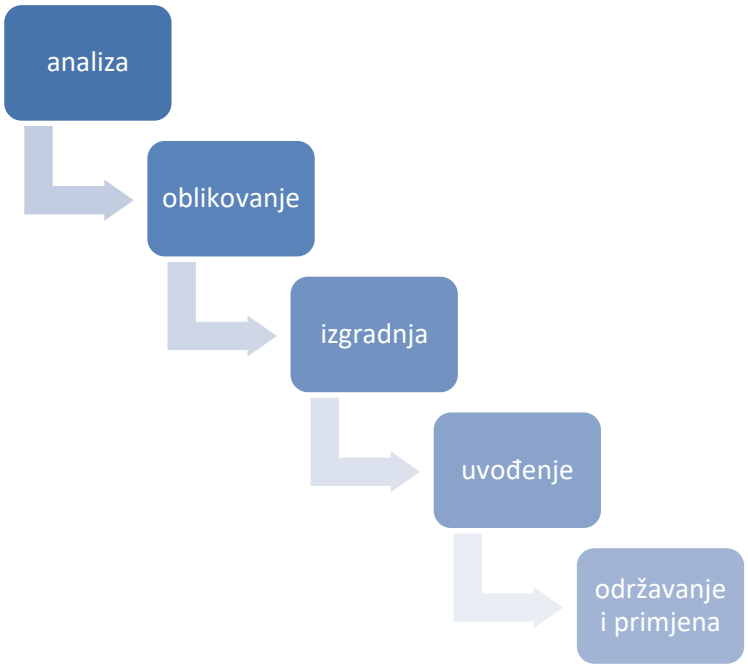
Vodopadni model dobio je ime zbog svog slijednog načina izvođenja faza upravljanja projektima. Grafički reprezentirano, zbog početne faze na najvišoj razini i slijedno spuštanje prema ostalim fazama, podsjeća na protok vode u slapovima.

Waterfall metodologija glavna je predstavnica tradicionalnih modela vođenja projekata iz koje su, njezinom modifikacijom i prilagođavanjem, nastale i ostale metodologije u tradicionalnom kontekstu. Njihova sličnost najviše se može primijetiti po istim ili sličnim fazama te njihovoj reorganizaciji i slijedu izvođenja.

Ovo je tradicionalna linearna metodologija u kojoj se na sljedeću fazu ne može prijeći dok prethodna faza nije završena. U sladu s tim, nema niti povratka na prethodnu fazu. Te su karakteristike stvarale probleme u izvršavanju projekata, prvenstveno zbog nedostatka

povratne informacije o napretku projekta, zbog čega su napravljene dorade *Waterfall* metodologije (Charvat, 2003).

Prema (Pavlić, 2011, str. 120) vodopadni model se sastoji od faza analize zahtjeva, oblikovanja, izgradnje, uvođenja te održavanja i primjene. Prikladan je za veće projekte i za dobro definirano okruženje, gdje postoje propisane procedure ručne obrade informacija ili računalni sustav, a rezultat je poznat iz niza sličnih ranije uspješnih projekata. Na sljedećoj slici možemo vidjeti faze vodopadnog modela te njihovo kaskadno izvođenje.



Slika 3: *Waterfall* metodologija (Prema: Pavlić, 2011)

Prije analize zahtjeva često se izrađuju konceptualni modeli. Njima se određuju granice poduhvata, opseg i izvodljivost planiranog projekta. To su modeli podataka i procesa kojima se ugrubo prikazuju koncepti zamišljenog projekta. Nakon toga slijedi analiza zahtjeva i predlažu se optimalna rješenja za te zahtjeve. Nakon toga kreće izrada dokumentacije gdje se grubi model podataka i procesa detaljnije razrađuje, pri čemu se najveći fokus stavlja na podatkovne sadržaje, funkcionalnost i tehnologiju rada te vizualni izgled sustava. Nakon toga slijedi implementacija. Ona naravno ovisi o vrsti i veličini projekta. Gledano iz aspekta razvoja informacijskih sustava (ili bilo kojeg softverskog rješenja) to bi bila faza u kojoj se programira i izrađuje sučelje tj. direktno izrađuje programski proizvod. Nakon toga slijedi testiranje i verifikacija, odnosno provjera i potvrda svega. Na kraju naravno slijedi održavanje odrađenog projekta.

Jedna od glavnih odlika vodopadnog modela jest da se s prelaskom na sljedeću fazu na prethodnu ne vraća. To se s današnjim zahtjevnim projektima pokazalo nedostatno, zato

što se kako bi se faza u potpunosti završila na nju potroši jako puno vremena. Ponekad u procesu razvoja projekta kasnije možemo primijetiti grešku ili potrebu za promjenom nečega što smo radili prije, što nije odlika ove metode. Iako ima svojih pozitivnih strana, za današnje projekte (a posebno informatičke) nije efikasna, i stoga su nastale danas mnogo korištenije agilne metodologije.

3.4.1. Prednosti i nedostaci vodopadnog modela

Vodopadni model u današnje vrijeme slovi za zastarjelu metodologiju sa sve manjim opsegom primjene. Usprkos tome, ona ima neke određene prednosti. Prema (Palmquist i sur., 2013) prednosti vodopadnog modela su:

- zahtjevi projekta su u potpunosti definirani prije faze implementacije,
- klijent već na početku projekta zna njegov opseg, vremensko trajanje i cijenu, koji su precizno definirani zahvaljujući detaljnom planiraju aktivnosti i zahtjeva projekta,
- potencijalni problemi prepoznati su u početnim fazama planiranja i specificiranja projekta, što osigurava uštedu vremena i povećava efikasnost u fazi implementacije,
- detaljna tehnička dokumentacija cijelog projekta olakšava osobama koje nisu sudjelovale u njegovoj implementaciji održavanje sustava nakon isporuke klijentu,
- olakšano testiranje sustava radi već unaprijed definiranih scenarija u tehničkoj dokumentaciji.

Upravo zbog nedostataka vodopadnog modela njegova primjena bilježi sve manji udio korištenja navedene metodologije. Nedostaci vodopadnog modela (Palmquist i sur., 2013):

- jednom kada je faza završila i počela nova, projektni tim se rijetko može vratit na prethodnu i učiniti neku promjenu/doradu, čime se potencijalne greške u prethodnim fazama ugrađuju u finalni proizvod,
- ukoliko se učini iznimka, i vrati se neku od prethodnih faza, takva vrsta pothvata uvelike podiže cijenu projekta, ali i samo trajanje te samim time projekt gubi na svojoj isplativosti,
- klijentima je problem u samom početku iznijeti sve specifikacije i zahtjeve projekta, bez mogućnosti kasnijeg dodavanja ili mijenjanja.

Vodopadni model iako je dosta izgubio na popularnosti, i danas se koristi. Svaka industrija, svako poduzeće i projekt su individualni, posebni i zahtijevaju drugačiji pristup. Dok za većinu informatičkih projekata vodopadni model nije efikasno rješenje, za neke projekte

drugog tipa može biti upotrebljiv. Važno je za svaki projektni tim i organizaciju da prouče i odrede prednosti i mane određenog modela (i metodologije) te da odaberu one koji najbolje odgovaraju potrebama njihovog projekta.

3.5. Tradicionalne metodologije

Tradicionalnim metodologijama smatraju se sve one koje imaju slijedni (fazni) pristup u kojem se prelaskom na sljedeći proces ne vraća na prethodni. To su sve one koje imaju uporište u vodopadnom modelu, nisu prilagodljive i jasno su strukturirane. Opseg projekta je jasno definiran, faze i procesi su strogo razloženi te ne nisu podložne promjenama. Vremenski okvir je jasno definiran s određenim rokovima, dok su rizici poznati i rijetko će se dogoditi neočekivane situacije.

Koju metodologiju za provedbu projekta odabrati jedna je od najvažnijih i najtežih odluka pri njihovom pokretanju. Svako poduzeće odabrat će metodologiju koja njihovoj vrsti projekta i načinu rada najviše odgovara. Svaka metodologija ima svoje prednosti i mane za određene vrste projekata.

3.5.1. PRINCE 2

Projekti u kontroliranom okruženju (eng. *Projects in controlled environments)* verzija 2 nastala je kao prepravljena prva verzija istoimene metode zbog kritika nedovoljne fleksibilnosti. Metodologija nastala je od strane vlade Ujedinjenog Kraljevstva primarno kao standard za razvoj informacijskih sustava a zatim postaje standard za vođenje svih projekata u vladi UK-a.

Projekt se razlaže na faze za lakše upravljanje i mogućnost učinkovite kontrole nad resursima i praćenje rada i razvoja projekta. Definirane su različite uloge i odgovornosti za vođenje projekta koje su detaljno opisane i prilagodljive tako da odgovaraju različitim veličinama i složenostima projekta. Upravljanje projektima korištenjem PRINCE2 metodologije označava baziranje na proizvod, što znači da ona koristi postupak identifikacije svih proizvoda (iz projekta) koji čine ili pridonose ostvarivanju ciljeva projekta.

PRINCE2 metodologija je temeljena na sedam principa koji omogućuju kontrolu od početka do kraja. Projekt se temeljito planira prije nego što se počinje izvršavati te su svaka faza i svi procesi jasno strukturirani. Principi su slijedeći (Trainer, 2013):

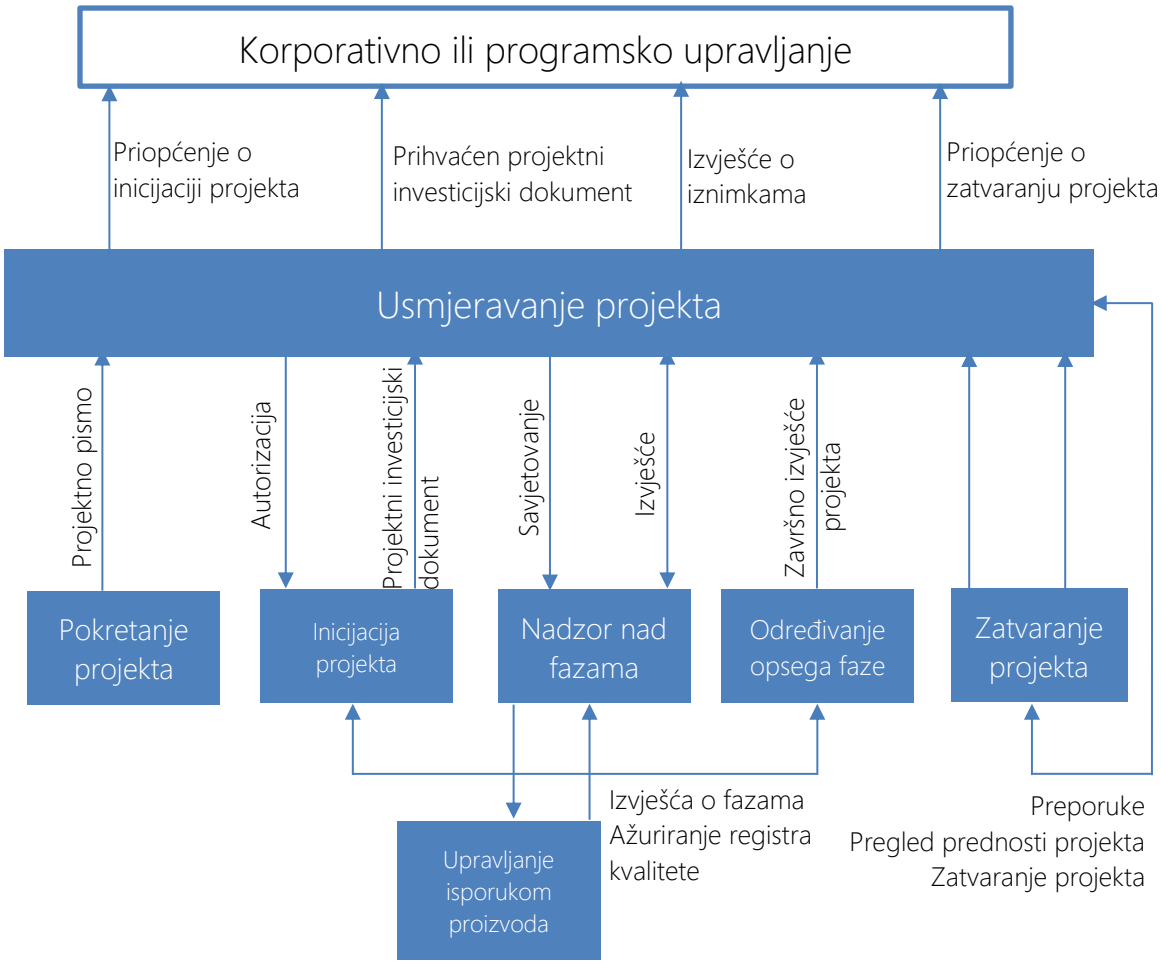
1. Projekti moraju biti poslovno opravdani
2. Projektni timovi konstanto uče, tijekom svake faze projekta
3. Uloge i odgovornosti su jasno i jednoznačno definirani

4. Rad se planira u fazama
5. Upravlja se projektnom iznimkom
6. Fokus je primarno postavljen na kvalitetu
7. Pristup je prilagođen za svaki individualni projekt

Metodologija se još naziva i strukturnim upravljanjem projekata zato što pomaže upravljati projektom na logičan i organiziran način, slijedeći definirane procese. PRINCE2 opisuje što bi projekt trebao raditi i kada i to čini nizom procesa koji pokrivaju aktivnosti potrebne za projekt, od pokretanja do zatvaranja. Zbog toga je PRINCE2 poznat kao procesno vođena metodologija, a sedam PRINCE2 procesa su kako slijedi (prema Trainer, 2013):

- **Pokretanje projekta** - provjera kvalitete i potencijala projekta
- **Usmjeravanje projekta** - proces kojim konstantno provjeravamo valjanost dokumentacije, izvješća, autorizaciju prelaska na novu fazu, odobrenje klijenta, itd.
- **Razumijevanje projekta** - konstantno razumijevanje projekta i uvid u vrijeme, troškove, kvalitetu, opseg, rizik i korist
- **Nadzor i kontrola nad fazama** - provjera napretka svake faze, potrebnih resursa i ispravnost rada
- **Upravljanje isporukom proizvoda** - osiguravanje da se proizvodi stvaraju i dostavljaju (na vrijeme uz što manji trošak)
- **Određivanje granica/opsega faze** - planiranje sljedeće faze kao i planiranje ukupnog projektnog plana i njegovog opsega.
- **Zatvaranje projekta** - isporuka gotovog projekta i dobivanje odobrenja za njegovo zatvaranje

Na slijedećoj slici možemo vidjeti njihove međuodnose i kako se razmjenjuju informacije.



Slika 4: PRINCE 2 model procesa (Prema: Trainer, 2013)

Na shemi vidimo kako procesi međusobno komuniciraju. Primjećujemo da su većinom slijedni, odnosno izvršavaju se jedan za drugim, kao i u većini tradicionalnih metoda. Proces koji se izvršava cijelo vrijeme je usmjeravanje projekta, odnosno to bismo nazvali vođenjem projekta. To su sve aktivnosti kojima pratimo sve aspekte projekta, usklađujemo komunikaciju s upravom, provjeravamo izvršava li se sve kako treba i na vrijeme. Pokretanje projekta je pretprojektni proces u kojem se skupljaju osnovne informacije o projektu. Iniciranje projekta osigurava da su svi dionici postigli sporazum prije značajnog trošenja. Treba sklopiti sporazume o onome što treba učiniti, kako, kada i zašto se to radi, očekivane koristi i kako će se postići kvaliteta. Nadzor nad fazama je postupak koji opisuje osnovne aktivnosti voditelja projekta i osigurava da faza ostane u okviru proračuna i rasporeda te traženoj kvaliteti korisnika. Upravljanje isporukom proizvoda opisuje vezu između voditelja projekata i rukovoditelja tima i posebno je važan kada su timovi vanjski i ne koriste PRINCE2. Dogovoreni

rad uključuje vrijeme i datume, kvalitetu i izvještavanje. Određivanje opsega faze je postupak u kojem se provjerava rad trenutnih faza i planira se sljedeća. Plan projekta i poslovni slučaj se ažuriraju i voditelj projekta obavještava projektni odbor o rezultatu završene faze, nakon čega odbor odobrava prelazak na slijedeću. Proces zatvaranja projekta događa se na prirodnom kraju projekta ili kada je potrebno prijevremeno zatvaranje. Zatvaranje projekta potvrđuje prihvaćanje projektnog proizvoda i da su postignuti ciljevi utvrđeni u dokumentu o pokretanju projekta.

3.5.1.1. Prednosti i nedostaci PRINCE2 metodologije

Prema (Pawar i Mahajan, 2017) prednosti PRINCE2 metodologije su sljedeće:

- **Predvidljivost**

Činjenica da PRINCE2 metodologija raščlanjuje projekte na faze znači da se ona može pomno nadzirati postupnom metodom od samog početka projekta do zatvaranja. Tako se zna što se očekuje i manje su šanse za greškama i neočekivanim situacijama. Predvidljivošću se projektom lakše i kvalitetnije upravlja a time se ostvaruju zacrtani ciljevi na vrijeme.

- **Korištenje najbolje prakse**

Projekt se smatra uspješnim samo ako daje kvalitetu proizvoda koja je ustanovljena pri njegovom pokretanju. Najbolji način da se to ostvari je korištenje najbolje prakse - jedan od čimbenika koji je metodologiju PRINCE2 zadržao jednako popularnom kao i danas.

- **Standardizacija**

Metodologija PRINCE2 standardizira svaki aspekt projekta kako bi se osiguralo da nema šanse za pogrešno komuniciranje ili zabludu. Standardiziranim postupkom jamči se da svi koji su uključeni u projekt znaju što treba učiniti i kada to treba učiniti.

- **Vremenska isplativost**

S obzirom na navedenu standardizaciju, vrijeme i novac se štede na obuci novih zaposlenika koji su već upoznati s PRINCE2 metodologijom. Manje vremena se troši na identificiranje rizika kako će biti istaknuto na početku projekta, a ako se pojavi neočekivani rizik, može se s vremenom rješavati kako faze projekta završavaju, a slijedeća započinje.

- **Isproban i testiran**

PRINCE2 je dostigao starost od 30 godina i još uvijek ide snažno, što je dokaz da ostaje pouzdana metoda u preko 50 zemalja širom svijeta. Koristi se u svakom sektoru poslovanja i smatra se metodologijom upravljanja projektima u kojoj se mjere sve ostale metode.

Usprkos mnogobrojnim prednostima, PRINCE2 sadrži i neke nedostatke, a to su (Pawar i Mahajan, 2017):

- **Mali broj alata i tehnika**

U PMBOK vodiču ima preko 100 alata i tehnika za vođenje projekata, dok je PRINCE2 zbog svoje standardizacije i nedovoljne fleksibilnosti ograničen na nekolicinu.

- **Ne obuhvaća meke vještine**

Danas vrlo važne “meke vještine” u vođenju projekata nisu obuhvaćene ovom metodologijom. To znači da bi voditelji samostalno trebali primjenjivati i učiti o ovim vještinama i time se gubi na vremenu te stvara dodatan trošak.

- **Za uspjeh je potrebna prijava višeg rukovodstva**

Kao što se može vidjeti i na shemi (slika *broj*) - uprava, odnosno viši menadžment mora biti dosta uključen. To znači da projektni timovi nisu u potpunosti samostalni i time se dodatno troši vrijeme i komplicira situacija.

- **Zahtijeva iskustvo kako bi se dobro primijenio**

Ova metodologija nije *“beginner-friendly”*, odnosno kako bi ju se moglo primjenjivati voditelji trebaju imati određeno iskustvo i praksu koristeći PRINCE2

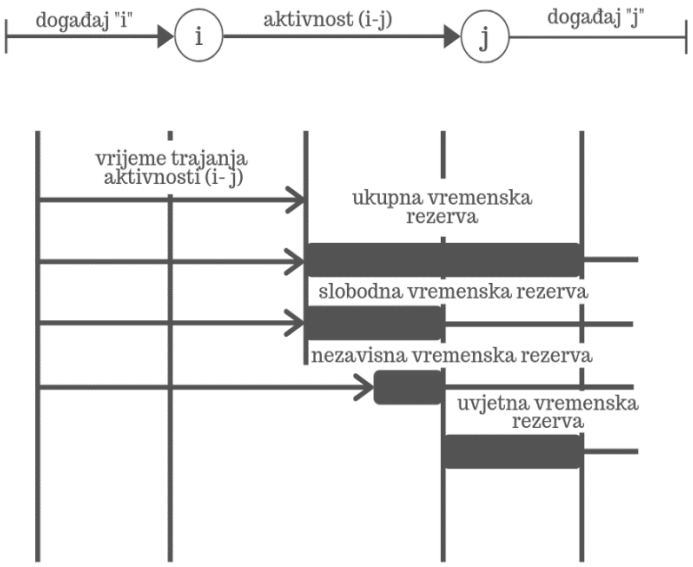
3.5.2. Metoda kritičnog puta

Metoda kritičnog puta *(eng. Critical Path Method – CPM)* izračunava teoretski najranije početne i završne datume, kao i zakašnjele datume početka i završetka, za sve aktivnosti bez obzira na bilo kakva ograničenja resursa, provođenjem analize prolaska naprijed i natrag kroz mrežu rasporeda. Regulirajući rani i kasni datumi početka i završetka nisu nužno vremenski raspored projekta; umjesto toga, oni ukazuju na vremenske periode unutar kojih se aktivnost može planirati s obzirom na trajanje aktivnosti, logičke odnose, vodstvo, zaostajanje i druga poznata ograničenja.

Dvije najvažnije stavke na koje se gleda pri izračunu kritičnog puta su troškovi i vrijeme. Neke aktivnosti mogu se izvršiti u najkraćem roku a prouzročiti velike troškove, kao i obrnuto – neke aktivnosti mogu malo koštati ali predugo trajati. Ideja je da se pronađe najoptimalniji put gdje je taj odnos prihvatljiv.

Kružnom oznakom definiraju se događaji dok se unutar kvadrata s lijeve strane piše najraniji završetak a s desne najkasniji. Na bilo kojoj mrežnoj stazi, fleksibilnost rasporeda mjeri se pozitivnom razlikom između ranih i kasnih datuma i naziva se "ukupno plutanje“. Vremenska rezerva je vremenski interval između najranijeg početka/kraja i najkasnijeg

početka/kraja neke aktivnosti. Ona prikazuje koliko se može izgubiti na sadašnjoj aktivnosti, a da to ne utječe na konačan datum projekta. Postoje četiri vrste vremenskih rezervi prikazane na slici 5:



Slika 5: Vrste vremenskih rezervi CPM-a (Brandenberger i Konrad, 1970)

Maksimalno dozvoljeno trajanje aktivnosti predstavlja razdoblje između najranijeg početka aktivnosti i najkasnijeg završetka aktivnosti. Ako je trajanje jedne aktivnosti jednako razlici najkasnijeg završetka i najranijeg početka te aktivnosti, onda se ova aktivnost naziva kritičnom. Kritični put je onaj koji se sastoji od kritičnih aktivnosti računajući od početnog do završnog događaja.

3.5.2.1. Prednosti i nedostaci Metode kritičnog puta

Metoda kritičnog puta dugo se koristi što daje očiti znak da ima mnogo prednosti, od koji su neke (O'Brien i Plotnick, 1999):

- Stvaranje ovisnosti između projektnih aktivnosti - ovo se izvodi konstrukcijom mrežnih dijagrama projekta ili dijagrama prioriteta.
- Organizira velike i složene projekte, što omogućava sustavniji pristup planiranju i rasporedu projekata, izvršavanju projekata i upravljanju rizikom.
- Omogućuje izračunavanje vremena svake aktivnosti. Time se određuje koliko neka aktivnost može kasniti bez utjecaja na plan projekta.
- Potiče voditelja projekta da skrati trajanje projekta optimiziranjem kritičnog puta i korištenjem tehnika kompresije ako je primjenjivo.

- Povećava vidljivost utjecaja izmjena rasporeda, koji su obično potrebni kada su propušteni glavni događaji ili kada je rizik od nedostatka glavnih prekretnica pretjeran.
- Omogućuje voditelju projekta da optimizira učinkovitost raspoređivanjem resursa na odgovarajući način, što će rezultirati smanjenjem ukupnih troškova.
- Pruža mogućnosti za odgovor na negativan rizik od prekomjernog rasporeda identificiranjem aktivnosti koje su najkritičnije.

Iako ima prednosti, ova metodologija broji i neke nedostatke od kojih su neki (O'Brien i Plotnick, 1999):

- Za velike i složene projekte izvest će se tisuće aktivnosti i odnosa ovisnosti. Bez softvera može biti teško za upravljanje. Ako se plan promijeni tijekom izvršavanja projekta, dijagram prioriteta morat će se izraditi nanovo .
- Za velike projekte s tisućama aktivnosti možda će biti teško ispisati dijagram mreže projekata i tada on postaje prekompleksan što članovima tima otežava razumijevanje plana.
- Nedostatak prilagodljivosti - CPM ne daje uvijek najbolje rezultate na projektima koji sadrže zadatke koje skupina tek treba izvršiti ili onima na kojima menadžeri tek trebaju prikupiti značajne podatke, što ga čini neprilagodljivim i ne može se primijeniti na svaki projekt.
- Nedostatak alokacije sredstava - iako CPM odlično izvršava svoju ulogu u procjeni vremena trajanja određene aktivnosti, jedan od glavnih nedostataka ove metode je da upraviteljima ne pruža sredstva za raspodjelu resursa potrebnih za ispunjavanje tih procjena.

3.5.3. Racionalni ujedinjeni proces (eng. *Rational Unified Process)*

Rational Unified Process (RUP) je objektno orijentirana metodologija za razvoj sustava i pogodna je za različite veličine timova koji sudjeluju u razvoju. RUP je strukturiran u dvije dimenzije (Jacobson, 2002):

- Vremenska dimenzija prikazuje kako se proces odvija u vremenu. Prikazuje dinamički aspekt procesa. Takav pogled na proces nam prikazuje faze i iteracije.
- Druga dimenzija predstavlja podjelu zadataka s obzirom na vrstu posla kojeg je potrebno obaviti. Prikazuje statički aspekt procesa. Prikazuje artefakte, aktivnosti i tijek rada.

Rational Unified Process je iterativni pristup vrlo prilagodljiv potrebama korisnika, primarno namijenjen projektima razvoja softverskih proizvoda. Procesi izvedeni iz RUP metodologije mogu biti lagani ili teški, ovisno o veličini i kompleksnosti projekata, što znači da su smjernice i okvir RUP metodologije prilagodljivi svim veličinama projekata. RUP je prilagodljiv i agilnim projektima, odnosno može se prilagoditi i oblikovati tako da ima karakteristike agilnih metodologija i podržava takve projekte. Nudi primjere najbolje prakse kroz upute i predloške, primjenjiv je na različite veličine timova, potiče produktivnost timova, omogućuje brz razvoj projekata, kontrolirane promjene i provjeru kvalitete, te smanjenje rizika (Charvat, 2003, str. 78).

RUP metodologija preporuča korištenje UML jezika za vizualno modeliranje, kako bi zahtjeve korisnika mogli razumjeti i ostali sudionici projekta, a isto tako su UML dijagrami preporučeni kao glavno sredstvo komunikacije između svih sudionika u razvoju (Rational, bez dat.)

RUP svoj razvoj bazira na objektno orijentiranom dizajnu koji u fokus stavlja procese, strukture te softverske i hardverske komponente potrebne za implementaciju. U objektno orijentiranom dizajnu nalaze se informacije o unutrarnjoj strukturi sustava, konfiguracijama i o tome kako bi taj sustav trebao biti realiziran (Davidović, 2016). U dizajnu su definirani nefunkcionalni zahtjevi, arhitektura sustava te konfiguracija sustava i rukovanje greškama. Ciljevi objektno orijentiranog dizajna su postizanje uporabljivosti, pouzdanosti, optimalnih performansi i mogućnosti ispravaka i nadogradnje.

3.5.3.1. Dimenzije RUP-a

RUP metodologija upravlja projektima kroz četiri glavne faze (Rational, bez dat.):

1. Inicijacija

Već dosad spomenuta i poznata faza pri pokretanju projekata je ona inicijalna, kojom svaki projekt započinje. Osnovni cilj je odrediti opseg i izvodljivost projekta, te pronaći rješenja za zahtjeve korisnika. U kontekstu RUP-a koji je iterativna metodologija, ova faza se izvodi kroz jednu ili dvije iteracije.

2. Elaboracija

Faza koja slijedi nakon što je ustanovljeno da je projekt isplativ i da će se provesti. Definira se arhitektura sustava i izrađuje se detaljan plan praćenja toka projekta. Faza se izvodi u nekoliko iteracija, odnosno dok ne ustanovimo stabilnu arhitekturu sustava.

3. Konstrukcija

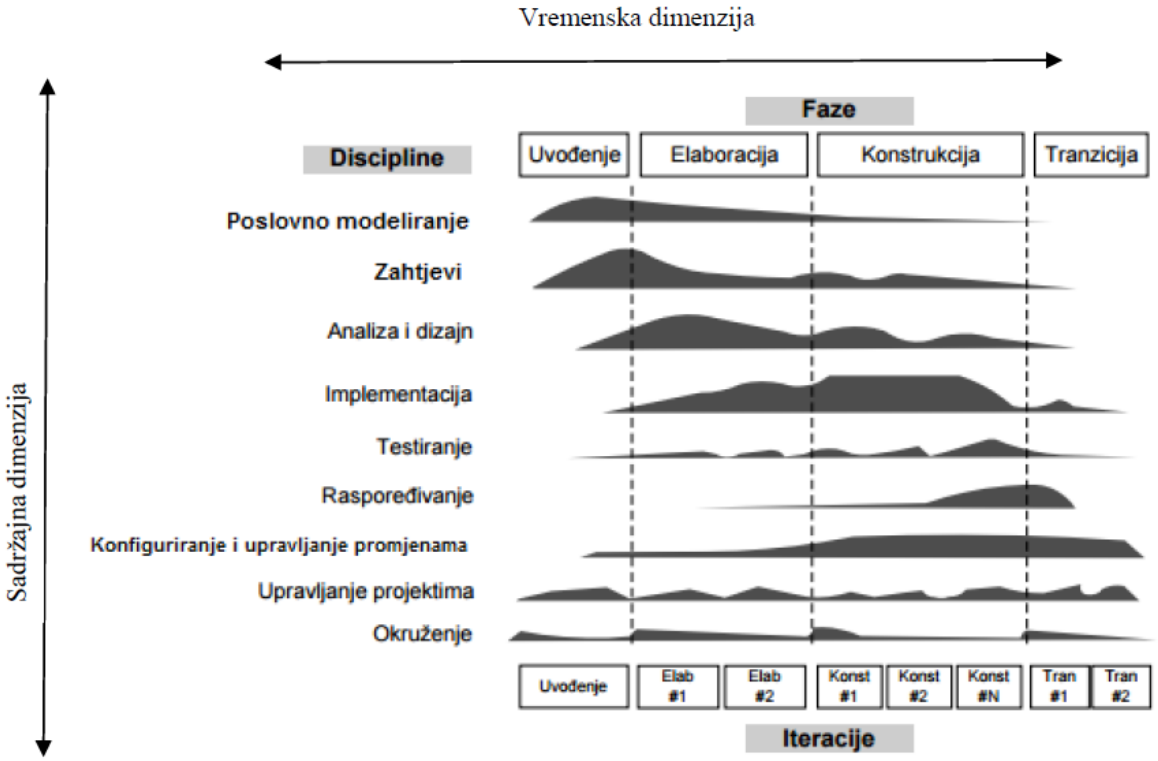
Faza konstrukcije predstavlja vremensku najdužu i opsegom najveću fazu. Tijekom faze glavni fokus je stavljen na detaljan dizajn, implementaciju i testiranje sustava. Sve

značajke i komponente projekta se objedinjuju i stvaraju sustav. Faza završava izradom „beta verzije“ sustava koja je spremna za testiranje u okolini korisnika. Cilj je isporučiti proizvod koji se može iskoristiti i vidjeti kako funkcionira kako bi korisnici mogli dati povratne informacije. U ovoj fazi se teži dostići te ciljeve uz što manje troškove.

4. Tranzicija

Faza tranzicije je posljednja faza. Cilj ove faze je omogućiti korištenje sustava od strane krajnjih korisnika, te završna testiranja. Na kraju ove faze ciljevi projekta bi trebali biti ispunjeni, a u nekim slučajevima završetak ove faze pokreće novi životni ciklus u kojem nastaje nova generacija izgrađenog sustava.

Na sljedećoj slici možemo u kojem omjeru se određena disciplina provodi po određenoj dimenziji RUP-a.



Slika 6: Dimenzije RUP-a (Izvor: Fertalj, 2016)

S obzirom da se faze provode iterativno, odnosno postupci u određenoj fazi se ponavljaju dok se postignu ciljevi, postavlja se pitanje spada li RUP u tradicionalne metode.

RUP posjeduje odlike prilagodljivosti i modularnosti što je karakteristika agilnih metoda ali u isto vrijeme posjeduje odlike podjele projekta na faze, ima strogo definirana pravila, puno dokumentacije te je veći naglasak na procese nego na ljude. Iako RUP ima iterativni pristup u pojedinoj fazi, svaka ta faza prati vodopadni model (čak i kada se ponovo provodi). RUP se

još uvijek svrstava u tradicionalne metode zbog većeg omjera odlika tradicionalnih metoda naspram agilnim.

2. Prednosti i nedostaci RUP-a

Kao i svaka metodologija, RUP ima svoje prednosti i nedostatke. Neke od prednosti RUP metodologije (Sousa, 2009):

- cjelovita metodologija s naglaskom na preciznu dokumentaciju i vizualne modele,
- uspješno rukovanje kompleksnim projektima gdje objektni dizajn osigurava fleksibilnost,
- velika pozornost je usmjerena na otklanjanje rizika povezanih s korisničkim zahtjevima,
- potrebno je manje vremena za integraciju pošto proces integracije prolazi tijekom cijelog razvojnog ciklusa softvera.

Iako se smatra da RUP metodologija ima više „dobrih“ strana, kao i kod svake metodologije određeni nedostaci postoje. Nedostaci RUP metodologije (Sousa, 2009):

- članovi tima moraju manje-više biti eksperti u području IT kako bi mogli sudjelovati u svim fazama projekta,
- razvojni proces je složen i ponekad neorganiziran,
- integracija procesa kod velikih projekata može uzrokovati zbunjenost i probleme tijekom faze testiranja.

4. Suvremeni ili iterativni pristup vođenju projektima

Zbog sve veće zahtjevnosti projekata i čestih promjena koje su se događale u okolini, tradicionalni pristupi u vođenju projekata više nisu bili efikasno i primjenjivo rješenje. Najveći problem u tradicionalnim pristupima bio je nedovoljne involviranosti klijenta u faze razvoja, testiranje tek na samom kraju projekta te nemogućnost vraćanja na prethodnu fazu jednom kada je ona završena. Tada je to zvučalo i bilo najlogičnije rješenje ali u današnjem svijetu čestih promjena i konstantnom potrebom za prilagodbom, takav način rada pokazao se neučinkovitim. Tradicionalne metode temelje se na predvidljivom pristupu što označava da bi se daleko unaprijed trebalo znati točno kako će se svaka faza odviti. Zbog raznih utjecaja i čestih promjena bilo je potrebno osmisliti metodologiju koja se prilagođava neočekivanim situacijama i nastalim izmjenama. Tako su osmišljene agilne metodologije koje se temelje na iterativnom pristupu (za razliku od tradicionalnih koje se temelje na faznom).

4.1. Iterativne faze upravljanja projektima

Iterativni pristup podrazumijeva razdiobu projekta na iteracije - razvoj manjih gotovih rješenja u kraćim vremenskim periodima. Nakon izrade samo jednog manjeg dijela projekta, on se testira, provjerava i stavlja u uporabu. Nakon što se ustanove moguće greške i potrebe za izmjenama, iteracija se ponavlja sve dok nije potvrđeno da se može prijeći na sljedeću. Svakom iteracijom projekt postaje bogatiji i napredniji za određenu funkcionalnost koja se prethodno razvijala. Svaka iteracija započinje planiranjem, zatim analizom zahtjeva, dizajnom, implementacijom, testiranjem i za kraj završava dokumentiranjem određene funkcionalnosti koja unapređuje cjelokupni sustav.

Slika 7: Procesi u iterativnom pristupu (Prema: Sharma i sur., 2012)

Kao što se može vidjeti na slici, agilni pristup sastoji se od velikog broja razvojnih ciklusa kod kojih se svaki puta radi novo planiranje. Konstantno se procjenjuju i testiraju manja

projektna rješenja te se temeljeno klijentskoj povratnoj informaciji, potrebne izmjene nadograđuju u sljedećem ciklusu.

4.2. Agilno upravljanje projektom

Zbog nezadovoljstva koje su voditelji IT projekata počeli osjećati korištenjem tradicionalnih metoda, 2001. godine skupina od sedamnaest stručnjaka iz područja softverskog inženjerstva odlučili su se sastati i pokušati pronaći rješenje. Rezultat njihovog rada je Agilni manifest kao inicijalni dokument i vodič za iterativno (agilno) vođenje projekata.

Agilni manifest je dokument koji opisuje temeljne postavke budućih agilnih metoda. Manifest možemo ukratko sažeti na četiri temeljne poruke (Beck i sur., 2001):

- Individualci i interakcija ispred procesa i alata
- Softver koji radi ispred iscrpne dokumentacije
- Saradnja s klijentom ispred pregovora o ugovoru
- Reagiranje na promjenu ispred praćenja plana

Prema tom istom manifestu, autori su naveli dvanaest osnovnih principa agilnih metoda (Beck i sur., 2001):

1. Najvažnije 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 upređu promjene da naručitelju stvore 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. Funkcionalni programski proizvod je osnovno mjerilo uspjeha.
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 samoorganizirajući timovi.
12. Tim u redovitim razmacima razmatra načine da postane učinkovitiji, zatim usklađuje i prilagođava svoje ponašanje.

Agilne metodologije temelje se na principima zadovoljavanja potreba klijenata kroz pravodobnu i kontinuiranu isporuku, prihvaćanje zahtjeva za promjenama u svim fazama izvođenja projekta, a promjene se usmjeravaju prema postizanju ciljeva i konkurentske prednosti. Funkcionalni proizvodi se isporučuju često kao manja gotova rješenja projekta, dok su poslovni i razvojni timovi u konstantnoj suradnji. Preporuča se i provodi se komunikacija licem u lice sa čestim ažuriranjima i sastancima. Prema (Parkash, 2011) promiče se održivi razvoj i održavanje tehničkog i dizajnerskog savršenstva. Ključna je jednostavnost i pojednostavljivanje posla koji se treba obaviti. Smatra se da najbolje arhitekture, zahtjevi i dizajn proizlaze od samoorganiziranih timova zato što su kreativni, inovativni i disciplinirani. Samostalni su, odnosno nitko njima ne upravlja - cijeli tim odgovoran je za razvoj projekta, a ne samo voditelj.

4.2.1. Prednosti i mane agilnog vođenja projekata

Agilnost se danas najviše primjenjuje u vođenju informatičkih projekata, odnosno projekata za razvoj softvera i uvođenja informacijskih sustava. S obzirom da to je industrija u kojoj su promjene i inovacije konstantne, logično je da će se većina informatičara složiti da agilni pristup ima više prednosti nego mana te da je definitivno efikasnije i bolje rješenje od tradicionalnog pristupa. Najveća prednost svakako se očituje u odnosu s klijentom (odnosno njegovoj involviranosti). Puno ranije (i češće) se isporučuju gotovi proizvodi koje klijent može testirati i dati svoju povratnu informaciju. Time te unaprjeđuje i odnos, odnosno rad u samom timu, jer znaju što trebaju raditi i znaju da to što rade je dobro. Od prednosti mogu se dakle istaknuti sljedeće stavke (prema: Sharma i sur., 2012):

- **Viša kvaliteta proizvoda**

Kao što je spomenuto, testiranje manjih gotovih dijelova proizvoda dio je proizvodnog procesa. Time se provjerava radi li proizvod ispravno i onako kako ga je klijent zamislio. Tako se na vrijeme mogu uvesti potrebne promjene i predvidjeti potencijalni problemi. Još neke odrednice kojima se postiže viša kvaliteta proizvoda su sljedeće:

- zahtjevi se definiraju i razrađuju na vrijeme kako bi poznavanje značajki proizvoda bilo što relevantnije
- kontinuirane integracije i svakodnevno testiranje uključuje se u razvojni proces, omogućujući razvojnom timu da rješava probleme dok su još svježi
- iskorištavaju se prednosti automatiziranih alata za testiranje
- izvođenje retrospektivnih pregleda, omogućujući timu da kontinuirano poboljšava procese i rad
- projekt se razvija u inkrementalnim, brzim ciklusima. To rezultira malim inkrementalnim izdanjima sa svakom nadogradnjom na prethodnoj funkcionalnosti. Svako izdanje temeljito je testirano kako bi se osigurala održavanje kvalitete projekta

- **Veće zadovoljstvo korisnika**

Vlasnik proizvoda uvijek je uključen, napredak razvoja ima visoku transparentnost i fleksibilnost što korisnika u konačnici čini zadovoljnijim. Zadovoljstvo se postiže na način:

- funkcionalnosti se redovito demonstriraju
- brže i češće isporučivanje proizvoda na tržište sa svakim izdanjem. Klijenti dobivaju rani pristup proizvodu tijekom životnog ciklusa
- klijenti su uključeni i angažirani tijekom cijelog razvojnog procesa projekta

- **Povećana kontrola projekta**

Zbog redovitih retrogradnih provjera i pregleda, kontrola projekta uvijek je na visokoj razini. To uključuje i transparentnost te pisanje popratne dokumentacije. Zbog toga što se u agilnom pristupu često koriste neki softverski alati kao pomoć, pregled i praćenje projekta dobiva višu razinu. Kontrola projekta se postiže zbog sljedećih stavaka:

- provođenje čestih sastanaka (nerijetko i na dnevnoj bazi) prije početka izvršavanja sljedećih zadataka
- provođenje retrogradnih provjera i pregleda
- korištenje alata za projektni menadžment
- povećana transparentnost i time mogućnost predviđanja potreba i potencijalnih problema

- **Smanjeni rizici**

Agilni razvoj pomaže uklanjanju rizika jer se problemi u razvoju uoče vrlo rano. S obzirom da se proizvod isporučuje iterativno i tester i korisnici testiraju manje dijelove

proizvoda, stoga je razvojni tim na vrijeme upozoren i može ispraviti svoje greške. Rizici su smanjeni zbog sljedećih postupaka:

- razvijanje proizvoda je u manjim segmentima što omogućava ranije primjećivanje rizika
- rano generiranje prihoda što osigurava smanjenje rizika nedovoljnih financijskih sredstava
- prilagodljivost i fleksibilnost smanjuju rizike neuspjeha u slučajevima promjene u okolini
- prilagođavanje potrebama klijenta kroz razvojni proces dovodi do smanjenja rizika sveukupnog neuspjeha i propadanja projekta

- **Veće zadovoljstvo članova tima**

Razvojni timovi u agilnim pristupima su većinom samostalni i nemaju voditelja projekta, već ga oni vode sami. Time se dobiva na opuštenosti i prilici da tim prilagodi tempo rada svojim mogućnostima i potrebama. Članovi tima postižu veće zadovoljstvo zbog sljedećih odrednica:

- samostalnost dovodi do veće kreativnosti, inovativnosti i mogućnosti da se razvojni stručnjaci izraze onako kako žele
- vlasnik proizvoda omogućava timu da ima sve što im je potrebno te da to imaju i na vrijeme
- smanjen je udio stresa zato što članovi tima nisu opterećeni rokovima koje je zadao netko drugi već onima koje su zadali sami
- veličinu i strukturu tima ponovo sami određuju te time ponovo dobivaju na većem zadovoljstvu a time i boljem radu
- zbog konstantnih testiranja i podjele projekta na više manjih dijelova, radnici na projektu ne moraju ispravljati ili ponavljati velike odrađene dijelove već one manje

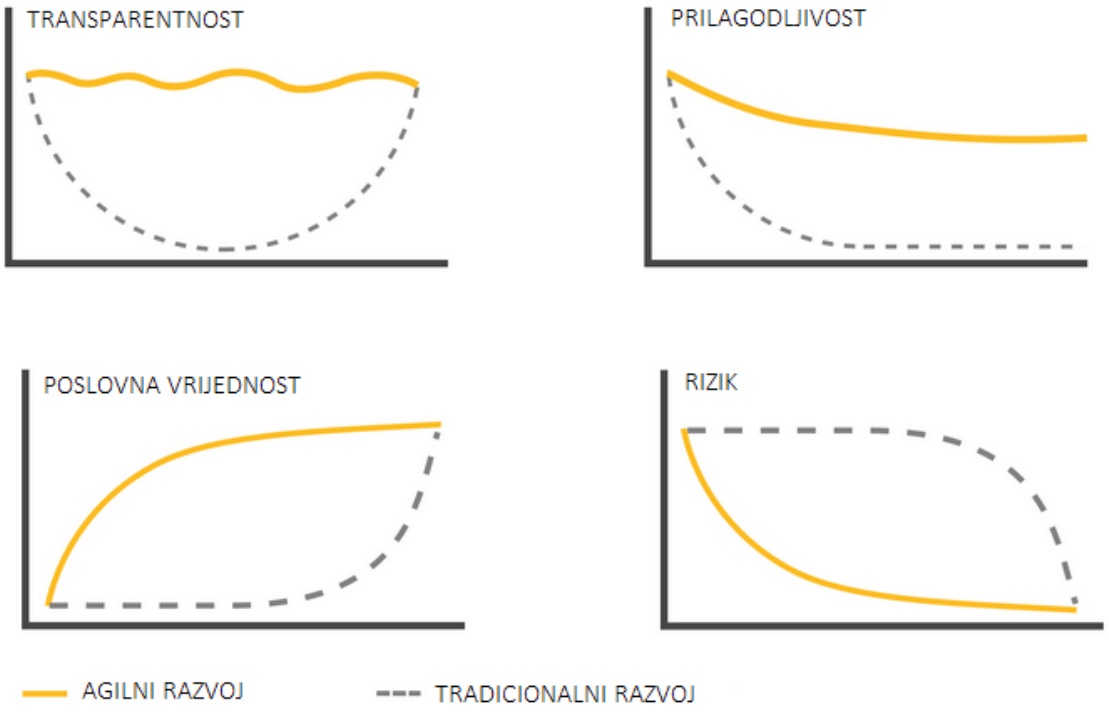
- **Poboljšana predvidljivost projekta**

Za projekt je važno da bude predvidljiv kako bi se mogli smanjiti rizici, odrediti tijek projekta te osigurati postizanje ciljeva. Predvidljivost projekta se postiže na sljedeće načine:

- zadržavanje duljine iteracije i nepromjenjivom alokacijom razvojnog tima tijekom cijelog projekta omogućuje projektnom timu da zna točan trošak za svaku iteraciju

- korištenje brzine pojedinačnog razvojnog tima omogućuje projektnom timu da predvidi vremenske rokove i proračune za izdanja, preostali zaostatak proizvoda ili bilo koju skupinu zahtjeva
- korištenje informacija s dnevnih sastanaka, dijagrama i popisa sa zadacima omogućuje projektnom timu da predvidi izvedbu za sljedeći set zadataka

Na sljedećoj slici prikazane su razlike u određenim performansama između agilnog i tradicionalnog pristupa. Većina navedenih prednosti mogu se vidjeti u sljedećim grafičkim prikazima.



Slika 8: Odnos parametara u tradicionalnom i agilnom razvoju (Izvor: Novoseltseva, 2016)

Transparentnost i poslovna vrijednost kod projekata odrađenih tradicionalnom metodom dobivaju rast tek pri samom kraju projekta. Kod agilnih metodologija transparentnost je konstantno prisutna a poslovna vrijednost vrlo brzo dobiva pozitivne rezultate. Već je poznato da tradicionalne metode nisu prilagodljive i strogo se drže onoga što je definirano na početku što je potpuna suprotnost kod agilnih metoda. Rizik kod tradicionalnih metoda tek pri kraju pada, dok se kod agilnih metoda nastoji smanjiti od samog početka pa tijekom cijelog razvoja projekta.

Iako agilni pristup ima jako puno prednosti, postoje neki aspekti ovih metoda koji nisu u svakom projektu pozitivni. Neki od nedostataka su sljedeći (prema: Sharma i sur., 2012):

- **Loš menadžment resursa**

Budući da se agilni pristup temelji na ideji da timovi neće znati kako će izgledati njihov krajnji rezultat (ili čak nekoliko ciklusa isporuke niz liniju) od prvog dana, teško je predvidjeti napore poput troškova, vremena i resursa potrebnih na početku projekta (a ovaj izazov postaje sve izraženiji kako projekti postaju sve veći i složeniji).

- **Skromna i neprecizna dokumentacija**

Dokumentacija se kreira tijekom projekta, a često i *„just in time“* za izgradnju rezultata, a ne na početku. Kao rezultat toga, projektni plan postaje manje detaljan, neprecizan, nekada i nejasan. Kada se uključuje nova osoba u proces nema jasnu dokumentaciju s kojom može odmah uskočiti u izradu.

- **Izrada cjelokupnog proizvoda u previše fragmenata**

Postupna isporuka može pomoći bržem plasiranju proizvoda na tržište, ali je i velik nedostatak Agilne metodologije. Kada timovi na svakoj komponenti u različitim ciklusima, kompletan izlaz često postaje vrlo fragmentiran, a ne jedna kohezivna jedinica.

- **Zatvaranje projekta nije strogo definirano**

Činjenica da agilni pristup na početku zahtjeva minimalno planiranje dovodi do čestih izmjena i skretanja s puta a time se odužuje i vrijeme izrade. Projekti tada nemaju definiran konačan kraj, jer nikada ne postoji jasna vizija kako izgleda konačni proizvod.

- **Zahtjevno mjerenje rezultata**

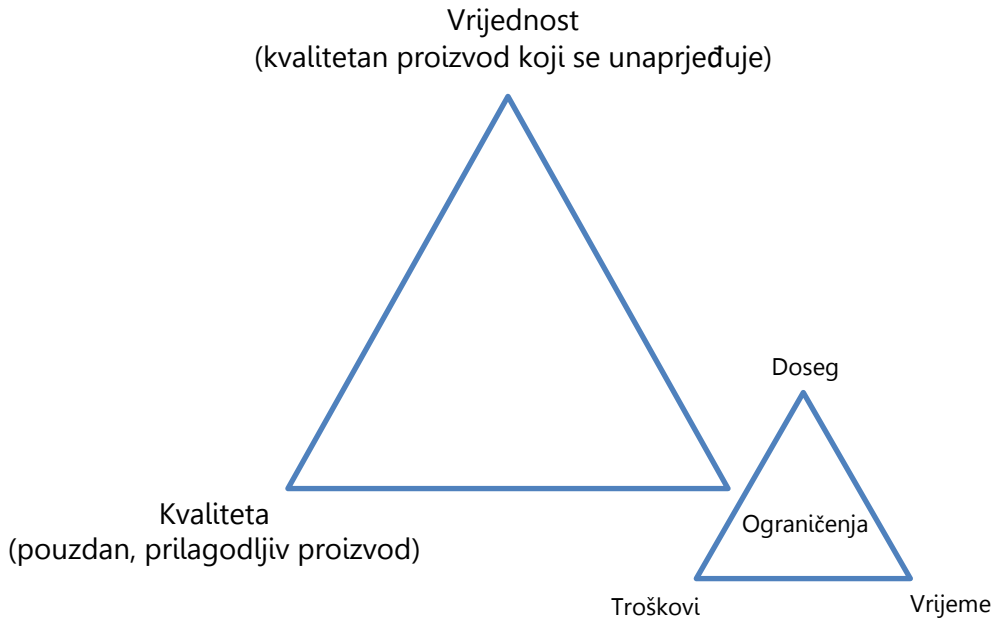
Budući da se agilni proizvod isporučuje u inkrementima, napredak praćenja zahtjeva pregled kroz više ciklusa. A priroda praćenja paralelno s radom znači da se ne može postaviti mnogo ključnih pokazatelja poslovanja na početku projekta.

4.3. Parametri agilnog projekta

Parametri tradicionalnog projekta su dinamički ovisni doseg, trošak i vrijeme, koji moraju biti u ravnoteži kako bi se postigla kvaliteta. Američki softverski inženjer i stručnjak u agilnim metodologijama, Jim Highsmith, primijetio je kako se fokusom na ove parametre u agilnom okruženju ne može postići zadovoljstvo klijenta (Talented Tester Support, 2018). Strogo praćenje rasporeda, raspoređivanje troškova i izvršavanje projekta unutar dosega otežava mogućnost prilagodbe, koja je glavna značajka suvremenih projekata i ključ postizanja

zadovoljstva korisnika. Timovi su konstantno u dilemi trebaju li strogo pratiti tijek i plan razvoja projekta ili se prilagoditi klijentu i nerijetko mijenjati taj plan.

Na sljedećoj slici možemo vidjeti redizajniran, odnosno „preuređen“ trokut parametara projekata kako bi on trebao izgledati za agilni razvoj.



Slika 9: Agilni trokut ograničenja (Prema: Highsmith i Cockburn, 2001)

Primjećujemo da je novi trokut ograničenja proširenje tradicionalnog, odnosno parametri tradicionalnog projekta postaju samo jedan parametar agilnog dok su preostala dva vrijednost i kvaliteta. Vrijednost se želi stvoriti za korisnika u smislu proizvoda koji se može distribuirati i čija uporaba donosi koristi za organizaciju. Kvaliteta se postiže kontinuiranom isporukom instanci proizvoda te prilagodba i usavršavanje prema povratnim informacijama klijenta. Ograničenja označavaju parametre tradicionalnih projekata gdje doseg, troškovi i vrijeme trebaju biti u ravnoteži.

Prema (Highsmith i Cockburn, 2001), ograničenja jesu važni parametri, ali ne i ciljevi projekta. Ciljevi trebaju biti kvaliteta i vrijednost a ograničenja su ta koja se trebaju prilagođavati kako bi se postiglo zadovoljstvo klijenata. Pri razvoju proizvoda fokus bi trebao biti na isporuci kvalitetnog cjelovitog rješenja, a ne na zadovoljavanju prethodno definiranih zahtjeva u opsegu projekta.

Iako je vrijednost i kvalitetu teško izmjeriti u usporedbi s troškovima i rasporedom, ipak se više treba usmjeriti na mjerenje vrijednosti isporučenog proizvoda, a ne na izračunavanje manje važnih segmenata u ograničenju. Prema (Highsmith i Cockburn, 2001), bolje je imati nejasne mjere stvarno važnih stvari nego precizne mjere manje važnih stvari.

4.4. Agilne metodologije

Metodologija je općenito znanost o metodama i njihovoj primjeni a metoda je planski postupak za postignuće nekog cilja na teoretskom ili praktičnom području (Strahonja, 2015).

Agilne metodologije razvile su se iz dosad opisanog agilnog načina rada. Omogućuju brzu isporuku i fleksibilnost te garantiraju kvalitetu proizvoda i zadovoljstvo klijenta. Agilne metodologije vođene su načelom “pojedinac i interakcija ispred procesa i alata” što znači da se rad usmjerava mogućnostima i potrebama dionika a ne na strogo izvršenje procesa. Tijekom godina razvila se nekolicina agilnih metodologija, čiji je predvodnik i gotovo sinonim za agilan način rada - SCRUM metoda.

4.4.1. Scrum metodologija

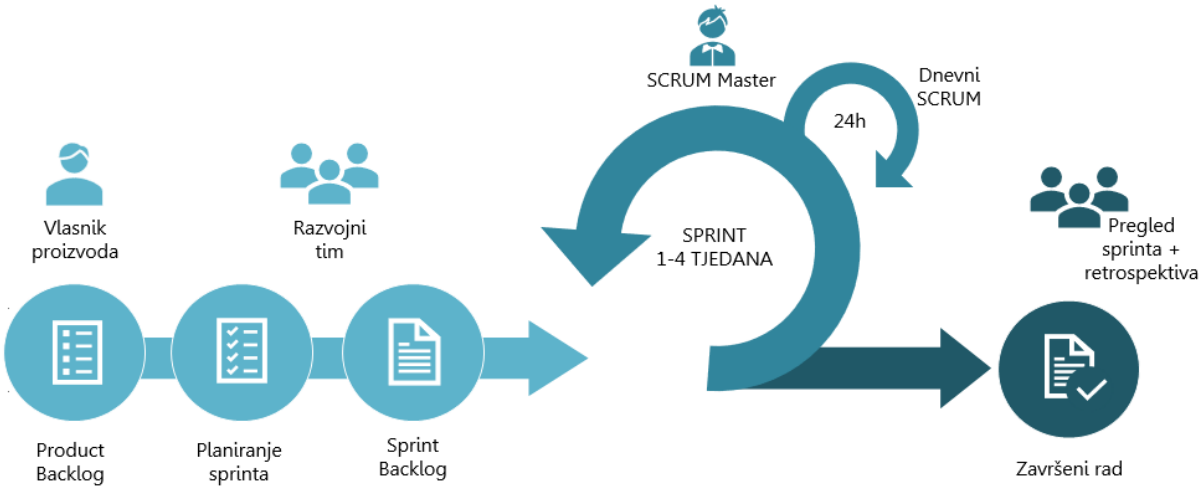
Kao što je opisano u povijesnom razvoju vođenja projekata, Scrum-ovi korijeni sežu još od 1986. godine ali metoda svoju uporabu službeno dobiva od 1995. godine kada je predstavljen rad “SCRUM proces u razvoju softvera”. Kao što i ime službenog rada kaže, ova metoda većinom se koristi u razvoju softverskih proizvoda. Iako se ne mora nužno koristiti za razvoj takve vrste projekata, *IT* je područje u kojem je ova metoda zaista postala popularna te je se dan danas sve više koristi.

Scrum je ime zapravo dobio prema izrazu *Scrum* u ragbiju. Taj se izraz koristi za situaciju nakon prekida kad se protivnički timovi zbijaju na hrpe i bore za posjed lopte. Svakim je prekidom (*Scrum-om*) tim sve bliže cilju i zauzima nove pozicije. Na putu prema cilju, tim ne pokušava i ne može predvidjeti sve situacije koje se mogu dogoditi u igri, već se stalno prilagođava trenutnoj situaciji na terenu, a glavni im je cilj progurati loptu što bliže protivničkoj liniji i postići zgoditak (Kardum, 2010).

Članovi tima se tijekom Scrum procesa mogu baviti složenim prilagodbama klijentima, dok istovremeno isporučuju proizvode najviše vrijednosti. Tri najosnovnije karakteristike SCRUM-a su da je lagan, jednostavan za razumjeti ali težak za savladati.

4.4.1.1. Scrum proces

Scrum proces se sastoji od skupa radnji koje se ponavljaju svako onoliko koliko si taj tim zada prema svojim potrebama i mogućnostima. Proces se sastoji od tima, vlasnika proizvoda (eng. *Product Owner*), Scrum mastera te događaja i artefakata. Proces je određen pravilima kojih se svi trebaju držati. Na sljedećoj slici je prikazan Scrum proces.



Slika 10: SCRUM proces (Prema: Kukhnavets, 2019)

Svaki proces započinje idejom za proizvod, što bi se moglo usporediti s inicijacijom projekta u tradicionalnom pristupu vođenja projekata. Ideja se prenosi na *Product Owner*-a koji sastavlja *Product Backlog* odnosno popis svih elemenata koje taj proizvod treba sadržavati kako bi u konačnici predstavljao funkcionalnu i smislenu cjelinu da zadovolji sve potrebe naručitelja i korisnika. Iz samog planiranja sprinta nastaje *Sprint Backlog* koji predstavlja zadatke koji su prema prioritetima izabrani i trebaju biti izvršeni u sljedećem razvojnom ciklusu odnosno sprintu. Nakon izrade *Backlog*-a slijedi planiranje Sprinta*. Scrum Sprint* predstavlja jedan ciklus izrade proizvoda koji završava jednim manjim ali konačnim rješenjem cjelokupnog projekta. Svaki dan tijekom sprinta se vode dnevni sastanci (*Daily Scrum).* To su kratki sastanci od 15 minuta u kojima se raspravlja o poslu kojeg su obavljali od zadnjeg dnevnog Scrum-a i o poslu kojeg imaju namjere napraviti do sljedećeg. Po završetku sprinta očekuje se da su svi zadaci izvršeni i da su funkcionalnosti *Sprint Backloga* razvijene, testirane, dokumentirane i spremne za isporuku. Nakon toga se slijedi osvrt i retrospektiva obavljenog posla. S*print Retrospective* predstavlja događaj gdje se tim fokusira na sam ciklus koji je iza njih te analiziraju cijeli vremenski period te pokušavaju pronaći odgovore postavljajući si pitanja:

- Što smo dobro napravili?
- Što smo loše napravili?
- Kako smo mogli bolje?
- Što možemo promijeniti?

Takvim pristupom omogućuju da sljedeća iteracija sprinta bude kvalitetnija i produktivnija te se oni kao tim unapređuju i prilagođavaju okolini.

U Scrum metodologiji često se koristi Scrum ploča. To je fizička ili digitalna tablica koja timovima služi da elementi *Sprin backlog-a* budu vizualizirani. Ploča je podijeljena na stupce

unutar kojih se postavljaju zadaci. Nazive stupaca timovi mogu odrediti svojim potrebama ali to najčešće budu ili faze projekta ili faze procesa (za napraviti, u tijeku i gotovo). Sa ovakvim pregledom zadataka, projekt dobiva na transparentnosti i jasnoći te potiče interakciju i diskusiju. Dionici projekta mogu dati svoj *input* što se tiče radnih zadataka te je svakome uvijek jasno što trenutno radi te u kojem je stadiju projekt.

4.4.1.2. Prednosti i nedostaci SCRUM metodologije

SCRUM metoda najpoznatija je i najrasprostranjenija agilna metoda. Stoga je logično zaključiti da sadrži većinu pozitivnih aspekata agilnog vođenja projekata. Najviše se ističe pozitivna strana fleksibilnosti i prilagodbe klijentu. Scrum metoda zamišljena je da bude fleksibilna tijekom cijelog životnog vijeka projekta. To pruža nadzorne mehanizme za planiranje izdanja, a zatim i za upravljanje varijablama projekta kako napreduje. Ovo omogućava organizacijama da u bilo kojem trenutku izmjene projekt i njegove rezultate time stvarajući najbolje izdanje. Prema (Ionel, bez dat.) velika prednost ove metode je zadovoljstvo timova. Developeri su slobodni i time se mogu fokusirati na kvalitetnu izradu proizvoda po tempu koji si sami određuju. Osim toga mali timovi imaju bolju radnu dinamiku i razmjena znanja je na visokoj razini. Ovo su (prema Adell, 2013) prednosti SCRUM metode:

- Zbog jednostavnog uočavanja i otklanjanja grešaka, programiranje napreduje brže i kvalitetnije je
- Razvoj projekta je svima jasan i vidljiv zbog čestih i redovitih (retrospektivnih) sastanaka, čime se postiže i olakšana kontrola projekta
- Zbog primjene povratne informacije klijenata u svakom novom sprintu, postiže se kvalitetniji proizvod i veće zadovoljstvo dionika.
- Zbog što su sprintovi kratki i informacije se primaju redovito, lako se nositi s uvođenjem promjena.
- Svakodnevni sastanci stvaraju transparentnost među članovima tima gdje si pomažu i dijele znanja što dovodi do produktivnosti i zadovoljstva.
- Kvalitetniji je proizvod lakše isporučiti u zakazanom roku.
- SCRUM se može raditi s bilo kojom tehnologijom/programskim jezikom, a odrednice metode mogu se primijeniti i u drugim industrijama
- Troškovi u pogledu procesa i upravljanja su minimalni, što dovodi do bržeg i jeftinijeg rezultata.

Iako je velika involviranost klijenta i njegova konstantna suradnja s razvojnim timom većinom pozitivan aspekt Scrum metode, ponekad ono otežava razvoj projekta. Nedostatak

ovog načina rada je u tome što česte i konstantne promjene te česti sastanci produljuju proces razvoja (Highsmith i Cockburn, 2001). Nerijetko se događa da ni sam klijent ne zna točno verbalizirati svoje zahtjeve te se tada broj iteracija povećava, zadovoljstvo pada a rok predaje se odgađa. Osim toga, postoje klijenti koji su ili udaljeni ili nisu navikli na taj način rada i tada prilagodba postaje zahtjevna. Još jedna stavka koja je većinom pozitivna ali u određenim situacijama ima negativan aspekt je veličina tima. Ova metoda idealna je za manje organizacije i timove, i iako se može primijeniti i na većim timovima nije ju lako za provesti. (Prema Adell, 2013) nedostaci Scrum metode su slijedeći:

- Doseg projekta je nedefiniran, odnosno nema konačnog datuma isporuke. Zbog konstantnih iteracija trajanje projekta može se značajno odužiti.
- Ako zadatak nije dobro definiran, procjena troškova i vremena projekta neće biti točna. U tom slučaju, zadatak se može odužiti na nekoliko sprintova.
- Ako članovi tima nisu predani, projekt se nikada neće dovršiti ili neće uspjeti.
- Dobar je većinom za male, brze projekte jer dobro funkcionira samo s malim timom.
- Ova metodologija treba samo iskusne članove tima. Ako tim čine ljudi koji su neiskusni, projekt se ne može dovršiti na vrijeme.
- SCRUM dobro funkcionira za upravljanje projektima kada Scrum Master vjeruje timu kojim upravlja. Ako prakticiraju previše strogu kontrolu nad članovima tima, to može biti izuzetno frustrirajuće, što dovodi do demoralizacije i neuspjeha projekta.
- S obzirom na veličinu timova, gubitkom samo jednog člana dovodi se do većih zastoja u razvoju projekta.

4.4.2. Ekstremno programiranje

Ekstremno programiranje (eng. e*Xtreme Programming, XP)* jedna je od popularnijih agilnih metoda. Njezin začetnik je Kent Beck koji je 1996. godine razvio ovu metodu za situacije kada su zadaci nejasni i često se mijenjaju, sa orijentacijom na male timove i sveukupno zadovoljstvo dionika projekta.

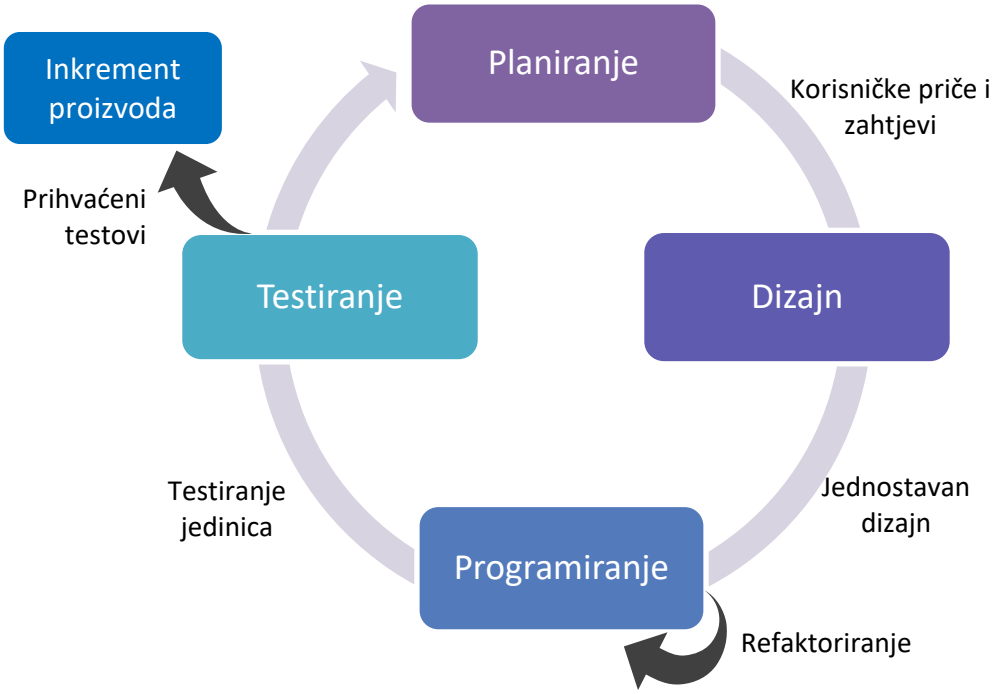
U primarni plan postavljeno je zadovoljstvo klijenta, sa redovitom komunikacijom, pravodobnom suradnjom i isporukom programskog proizvoda onda kada to klijent treba. Razvojni tim reagira na promjenu zahtjeva čak i u kasnim fazama razvojnog ciklusa. Naglasak je na rad u timu gdje su klijent i razvojni programeri partneri a okruženje je jednostavno što razvojnom timu donosi produktivnost i zadovoljstvo (Topić, 2012).

Prema (Jeffries i sur., 2001) Ekstremno programiranje temelji se na pet osnovnih vrijednosti:

- Komunikacija
- Jednostavnost
- Povratna informacija
- Hrabrost
- Poštovanje

4.4.2.1. Procesi Ekstremnog programiranja

Kao pripadnica agilnih metoda, i XP isporučuje proizvod u niz manjih dijelova pri čemu je svaki u potpunosti integriran i funkcionalan dio krajnjeg proizvoda. Četiri glavna procesa razvoja sustava XP-om su planiranje, dizajn, programiranje i testiranje. Na slijedećoj slici možemo vidjeti odnos tih procesa.



Slika 11: Procesi razvoja sustava u XP-u (Prema: Software Engineering Fundamentals, 2014)

Pri planiranju kod XP-a, umjesto klasične dokumentacije za definiranje zahtjeva koriste se korisničke priče. U njima korisnici opisuju u par rečenica što bi sustav trebao raditi iz njihove perspektive (bez tehnički zahtjevnih termina). Prema (Tadić, 2005) njihova svrha je planiranje razvoja i vremena dostave inkrementa proizvoda. Aktivnosti se u agilnom duhu ne moraju isplanirati na početku projekta, već prije početka određene iteracije. Pri planiranju se procjenjuje trajanje ispunjenja određene korisničke priče u pojedinoj iteraciji.

Cilj dizajniranja je jednostavnost sustava pri čemu se ne dodaju nove funkcionalnosti prije predviđenog roka kako bi se projekt što brže izvršio. Prilikom dizajniranja koriste se interno definirane metafore za konzistentno definiranje klasa i objekata (Info Novitas, bez dat.)

Tijekom procesa programiranja unaprijed se kreiraju testovi kako bi se znalo koje će funkcionalnosti konkretno razviti te radi uštede vremena. Jedna od poznatih odlika XP-a je programiranje u paru, pri čemu jedna osoba programira dok ju druga kontrolira. Prema (Info Novitas, bez dat.) programski kod se često integrira te je vlasništvo nad cjelokupnim programskim rješenjem zajedničko kako bi svatko mogao raditi izmjene i popravke na bilo kojem dijelu kôda.

Testovi se kreiraju prije početka implementacije u vlastitom razvojnom okruženju kako bi mogli biti automatizirani. Kreiraju se prema definiranim korisničkim pričama i služe kako bi se provjerila ispravnost kreiranih funkcionalnosti.

4.4.2.2. Prednosti i nedostaci Ekstremnog programiranja

Glavna prednost ove metodologije i razlog zašto ju puno razvojnih timova koristi je jednostavnost. Ne koristi se previše dokumentacije i ne troši se puno vremena na planiranje. Proizvod se razvija u manjim inkrementima s jednostavnijim kôdom kojeg svatko može mijenjati i ispravljati. Programiranje u paru donosi mogućnost ranog uočavanja grešaka i dijeljenje znanja a time i veće zadovoljstvo razvojnog tima. Projekti koji se ovom metodologijom provode su najčešće manji i jednostavniji s fokusom na brzoj isporuci a ne kompleksnom dizajnu. Kao i kod SCRUM-a česta testiranja i povratna informacija od korisnika donosi u svakoj iteraciji kvalitetniji i bolji proizvod a time se postiže zadovoljstvo klijenata. Prema (Jeffries i sur., 2001) prednosti su slijedeće:

- **Robusnost:** Fokusiranje na jednostavnost velika je prednost ove metode. Ovaj pristup stvara softver koji radi brže, s manjom količinom nedostataka. Otkrivanje grešaka zajamčeno je kada se tijekom faze razvoja rade redovna testiranja.

- **Otpornost:** Ova vrsta programiranja prilagođava promjene zahtjeva dobivanjem i pohranjivanjem korisničkih priča od samog početka, kao i stalnim povratnim informacijama tijekom iteracija.

- **Ušteda troškova:** XP pomaže u skraćivanju neproduktivnih aktivnosti kako bi se smanjili frustracija i troškovi. Izbjegava se nepotrebna papirologija kako bi se programerima omogućilo da se usredotoče na kodiranje. Činjenica da se promjene vrše na temelju povratnih informacija klijenata tijekom razvojnih faza održava ukupne troškove niskim.

- **Zadovoljstvo zaposlenika:** Programeri imaju svoju slobodu za integraciju i raspored vremena. U timu se dijele znanja kao i cjelokupni kôd gdje svi zajedno rade na postizanju što boljeg proizvoda.

Najočitiji nedostatak ekstremnog programiranja bilo bi nemogućnost provedbe ove metode na velike timove i projekte. Uz to se veže i nedovoljno dokumentiranje i praćenje prethodnog i budućeg rada. U većim sustavima bi zbog nedostatka dokumentacije moglo doći do konfuzije i neorganiziranosti. Osim toga, velika usmjerenost na kôd a minimalna na dizajn nije u svakom slučaju idealna s obzirom da je dizajn vrlo važan pri izradi softverskih rješenja. Prema (Jeffries i sur., 2001) nedostaci su slijedeći:

- **XP se oslanja na vrlo mnogo faktora**: ovo je u osnovi minimalistički proces. Njegov nedostatak energičnosti nadoknađuje se brojnim postupcima.

- **Visok rizik od neuspjeha**: Ukoliko nešto pođe po zlu ova metoda nema puno alternativnih rješenja niti plan kako se suočiti s neočekivanim situacijama.

- **Kodeks**: Ova vrsta programske metodologije je usmjerena na kod, a ne na dizajn. Ovo se može pokazati vrlo napornim kada su uključeni veći softverski projekti.

- **Dugotrajnost i opsežnost:** Probno kodiranje, testiranje i velik broj ponovljenih iteracija dovodi do dugotrajnosti projekta. To se uglavnom događa zbog nedostataka koji nisu dobro dokumentirani i kodova koji nisu dovoljno strukturirani.

4.4.3. Kanban metodologija

Poznato je da od japanskog automobilskog proizvođača Toyote proizlaze nekolicina metoda i primjera dobrih organizacijskih praksi. Jedna od metoda nastalih 1940. godine u toj tvornici je i Kanban. Ideja je bila naručivati i dobavljati resurse za proizvodnju u točnim količinama koje su potrebne i ne skupljati zalihe. Signal za kretanje proizvodnih dijelova ili potrebu za istima obavljao se prosljeđivanjem *kanban* fizičke kartice te je tako ova metoda dobila ime. Iako se od tada do danas značajno razvila i unaprijedila, *„just-in-time“* ideja ostaje srž ove metode (Radigan, bez dat.)

Kanban uporište ima u *Lean,* odnosno linijskoj, „vitkoj“ proizvodnji. Glavni cilj *Lean* načina rada je smanjenje trošenja vremena, ljudskog napora i sveukupnih troškova u proizvodnoj okolini. Linijsko Kanban razmišljanje u fokusu ima uspješno odrađivanje radnih zadataka, bez naglaska na to tko će ih izvršiti. Ljudi rade timski i zajedno, te zbog različite brzine izvršavanja zadataka i različite razine znanja, potrebna je redovita sinkronizacija (Ikonen, 2011).

Osnovna načela Kanbana (prema Hiranabe, 2008) koja kontinuirano prati tim su:

• Vizualna kontrola (Kanban ploča)

• Ograničenje posla u procesu (eng. *Limit Work in Process – LWIP*)

• Povlačenje (eng. *pull*) kartica (s ograničenjem *WIP*)

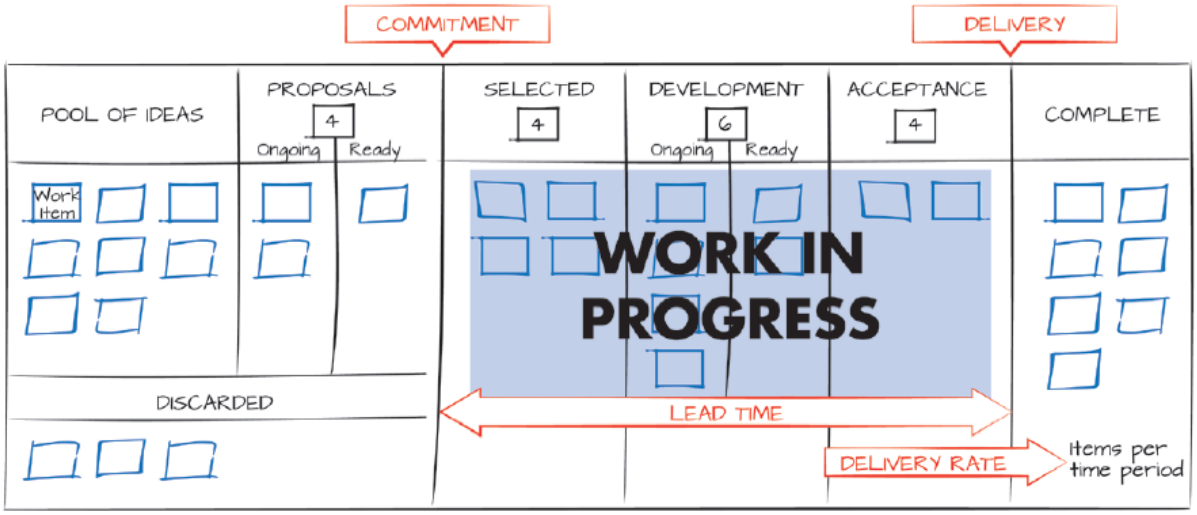
• Povećanje propusnosti

• Fiksna Kanban zaliha (eng. *backlog*)

• Kvaliteta ugrađena u proces

4.4.3.1. Kanban tijek rada

Tijek rada u Kanban metodi prikazuje se uz pomoč Kanban ploče. Radni elementi prolaze kroz faze slijedno prikazujući ih s lijeva na desno. Ukoliko postoji potreba za ispravcima i doradama, radni zadatak se može vratiti u prethodnu fazu. je Primjer Kanban ploče nalazi se na slijedećoj slici.



Slika 12: Kanban ploča (Izvor: Anderson i Carmichel, 2016)

Prema (Anderson i Carmicher, 2016) ploča se sastoji od pet komponenata:

- **Vizualni signali** – koriste se kartice na kojima se zapisuju zadaci. Za agilne timove, jedna kartica može predstavljati jednu korisničku priču. Ovakvim vizualnim prikazom dionicima je uvijek jasno na čemu trenutno projektni tim radi.
- **Kolumne** – svaka predstavlja specifičnu aktivnost koje zajedno čine tijek rada. Kartice se kreću po kolumnama sve dok aktivnosti nisu do kraja izvršene. Tokovi rada najčešće su označeni naslovima „za napraviti“, „trenutno se izvršava“ i „završeno“, ali naravno da može biti puno kompleksnije od toga.

- **Ograničenje posla u procesu** – predstavlja maksimalan broj kartica koje mogu biti u jednoj kolumni za definirano vrijeme.
- **Točka preuzimanja odgovornosti** – dio ploče na kojoj se nalazi *backlog.* Korisnici i članovi tima stavljaju ideje za projekt koje timovi mogu preuzeti kada su spremni.
- **Točka isporuke** – kraj Kanban tijeka rada kada je proizvod ili usluga (rezultat projekta) u rukama klijenta. Cilj je prebaciti kartice s točke preuzimanja odgovornosti do točke isporuke u najbržem mogućem vremenu.

4.4.3.2. Razlika između Kanban i SCRUM ploče

U obje metode je praksa koristiti ploču preko koje se prikazuju radni zadaci i faze u kojima se nalaze. Iako je razlika između ove dvije ploče suptilna, ona postoji. Prema većini interpretacija SCRUM koristi Kanban ploču, samo sa vlastitim procesima, artefaktima i ulogama. Prema (Radigan, bez dat.) razlike su slijedeće:

- *Scrum sprint-ovi* imaju početni i završni datum, dok je Kanban proces u tijeku
- Uloge u SCRUM-u su jasno definirane dok u Kanbanu nisu. Oba tima su samoorganizirajuća
- Kanban ploča se koristi koz cijeli životni ciklus projekta, dok se kod SCRUM-a nakon svakog *sprint-a* briše i obnavlja
- Scrum ploča ima definiran broj zadataka i strogi datum za njihov završetak
- Kabnan je više fleksibilan u pogledu zadataka i vremena. Zadaci se mogu ponovo prioritetizirati, dodijeliti i ažurirati ukoliko je potrebno

4.4.3.3. Prednosti i nedostaci Kanban metodologije

Kanban je u odnosu na ostale agilne metodologije ona koja se najviše koristi u industrijama izvan *IT-a.* Zbog svoje jednostavnosti, općenitosti i dobrih rezultata, mnoga velika i uspješna poduzeća prisvojila su ovaj način rada. Kanban sadrži pozitivne aspekte agilnih principa kao što je fleksibilnost, transparentnost te orijentiranost na zadovoljstvo svih dionika. Prema (Ganev, bez dat.) prednosti Kanban metodologije su slijedeće:

- **Jednostavnost:** Kanban je vrlo jednostavna metodologija za shvaćanje te je zbog svoje prilagodljivosti prigodan za primjenu u mnogo različitih industrija i veličina poduzeća.

- **Smanjenje troškova** zbog reduciranog otpada i zaliha koje nisu potrebne. Najveća ušteda je vremenska zato što se osim naručivanja resursa i zadaci definiraju *just in time,* odnosno neposredno prije njihovog izvršavanja a ne tjednima unaprijed.

- **Kontinuirano usavršavanje:** Vizualni sustav vođenja projekata omogućava lakše pregledavanje procesa i uočavanje prostora za poboljšanje.

- **Povećan broj *outputa:*** Zbog limitiranja posla u procesu, timovi su motivirani raditi zajedno kako bi se trenutno zadani posao priveo kraju. Time se uklanjaju distrakcije mogućeg *multitasking-a* i timovi su fokusirani odraditi zadatke u što kraćem vremenu.

- **Kvalitetniji proizvod:** Kanbanova usredotočenost na neprestano usavršavanje i agilno reagiranje na probleme često znače da projekti promatrani do završetka imaju manje pogrešaka i zahtijevaju manje ponavljanja zadataka. Postavlja kontrolu kvalitete u proces upravljanja projektima, dajući preciznije rezultate.

Nedostaci Kanban metodologije su slijedeći (Ganev, bez dat.):

- **Kriva interpretacija zadataka:** Postoji mogućnost krive interpretacije informacije na Kanban ploči što dovodi do grešaka i smanjuje kvalitetu proizvodnog procesa. Iako se manje gubi vremena na sastanke, ne raspravljajući o zadacima timovi se dovode do situacije krivog shvaćanja zadataka i stadija projekta.

- **Nedostatak vremenskih okvira:** Iako neimanje rokova donosi slobodu i manje stresa članovima tima, može prouzročiti kašnjenja i dulje trajanje projekta od očekivanog.

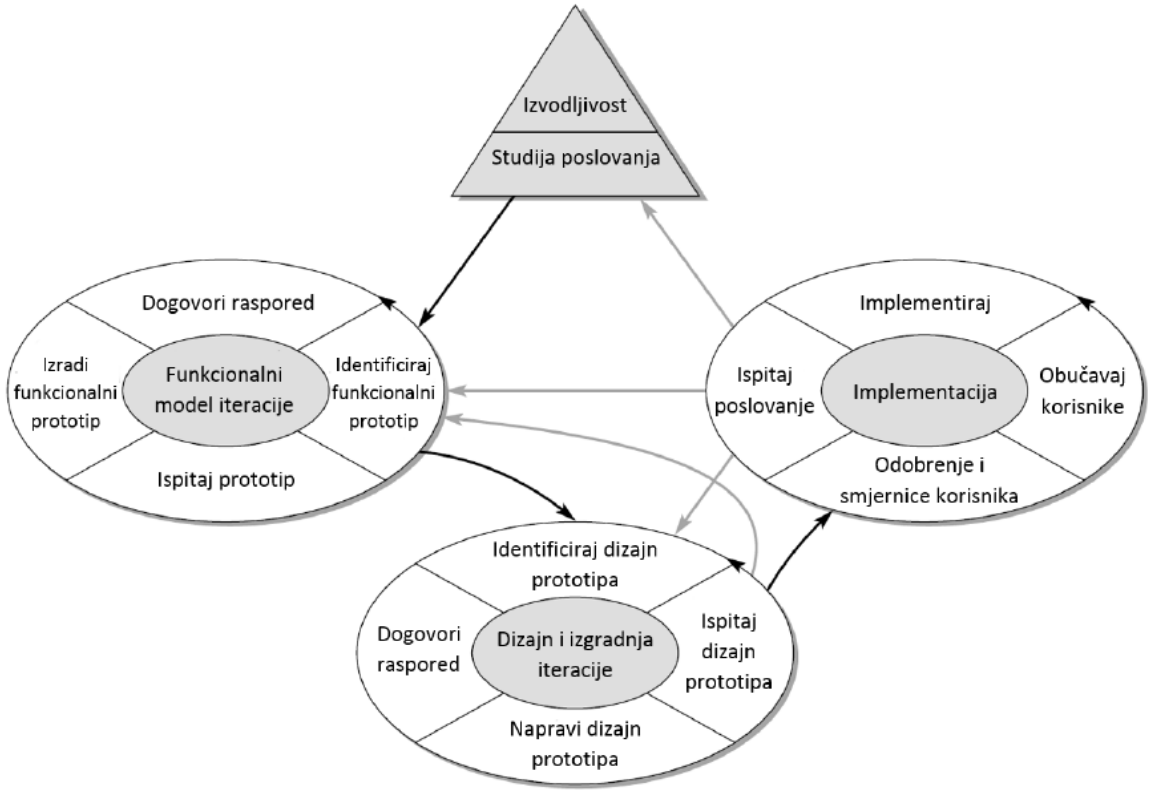
- **Krivo korištenje ploče dovodi do grešaka:** Za vrijeme razvoja projekta, postoji mogućnost da timovi naprave ploču previše kompleksnom što je upravo suprotno od glavne ideje Kanbana te projekt zbog toga može patiti. Isto tako, ukoliko se ploča ne ažurira sukladno fazama projekta, mogu nastati problemi u procesu razvoja.

4.4.4. Metoda dinamičkog razvoja sustava

Metoda dinamičkog razvoja sustava (eng. *Dynamic System Development Method – DSDM)* razvila se sredinom 90-ih godina prošlog stoljeća u Ujedinjenom Kraljevstvu kao razvojni okvir za *Rapid Application Development (RAD).* Glavni razlog izrade RAD metode bilo je nezadovoljstvo vodopadnim modelom i potreba za agilnijom metodo. Zbog toga što se RAD razvijao na nestrukturiran način i nisu postojale definicije oko njegovih procesa, pokrenut je DSDM konzorcij za održavanje i definiranje pravila ovih metoda (Farkaš, 2015).

Glavna filozofija DSDM-a temelji se na usklađivanju projekta sa jasno definiranim strateškim ciljevima te rane isporuke sa realnim koristima u poslovanju. Filozofiju slijedi ideja da su vrijeme i resursi prioritet a tek onda se određene funkcionalnosti mogu ispravljati ukoliko je to potrebno.

DSDM se izvodi kroz tri glavne faze: pretprojektna faza, faza životnog ciklusa projekta i postprojektna faza. Faza životnog ciklusa dijeli se na pet stupnjeva prikazanih na slijedećoj slici.



Slika 13: Faze DSDM procesa (Izvor: Farkaš, 2015)

Pet faza životnog ciklusa projekta su studija izvodljivosti, studija poslovanja, funkcionalni model i iteracije, implementacija te dizan i izgradnja iteracije. Prve dvije faze su slijedne i izvršavaju se samo jednom dok su preostale iterativne i inkrementalne. U DSDM metodologiji iteracije se gledaju kao „vremenske kutije“ (eng. *timebox*) koje traju prethodno definirani vremenski period, što može biti od nekoliko dana do nekoliko tjedana.

Studija izvodljivosti faza je u kojoj se s obzirom na tip projekta, dionike i organizacijske potrebe određuje treba li se projekt provoditi uz pomoć DSDM metodologije. Osim toga se bavi i rizicima projekta te tehničkim mogućnostima njegova izvršavanja.

Studija poslovanja je faza u kojoj se analiziraju osnovne karakteristike poslovanja te korištena tehnologija. Najčešće se organiziraju radionice na kojima prisustvuju razni stručnjaci kako bi se analizirao sustav te se dogovorili razvojni prioriteti.

Funkcionalni model iteracije je prva iterativna faza nakon prethodnih slijednih. U svakoj se iteraciji planiraju zadaci i pristup te se na kraju vrši analiza radi provjere ispravnosti. *Output* ove faze je prototip, a stečena iskustva tijekom njegova razvoja služe za poboljšanje modela analize.

Dizajn i izgradnja iteracije je faza u kojoj se uglavnom sustav izrađuje. Rezultat je testirani sustav koji ispunjava neke dogovorene funkcionalnosti. Prototip testiraju korisnici te se daljnji razvoj temelji na njihovoj povratnoj informaciji.

Faza implementacije je vrijeme u kojem se sustav prenosi iz razvojnog okruženja u stvarno poslovno okruženje. Provodi se edukacija korisnika i daljnja nadogradnja ukoliko je potrebna.

4.4.4.1. Prednosti i nedostaci DSDM metodologije

Prednosti DSDM metodologije ne razlikuju se uvelike od prednosti ostalih agilnih metodologija. Visoka uključenost korisnika u proces razvoja, brze isporuke inkrementa proizvoda i transparentnost samo su neke od prednosti. Prema (Stapleton, 1997) prednosti su slijedeće:

- Pruža proces neovisan o tehnici izvođenja
- Fleksibilan je u pogledu evolucije zahtjeva
- Strogo pridržavanje vremena i proračuna omogućuje izvršavanje projekta na vrijeme unutar zadanih troškovnih okvira
- Uključuje sve dionike u razvojni proces
- Naglasak na testiranju je tako jak da se očekuje najmanje jedan ispitivač od svakog projektnog tima
- Dizajniran iz temelja poslovnih ljudi, pa se prepoznaje poslovna vrijednost za koju se očekuje da će biti najveći prioritet u isporuci
- Ima specifičan pristup u određivanju koliko je svaki zahtjev važan za iteraciju

- Postavlja očekivanja dionika od početka projekta da neće svi zahtjevi učiniti konačnim rezultatom.

Najveći nedostatak DSDM metodologije je da je skup za implementaciju zato što podrazumijeva edukaciju developera i korisnika za njezino uspješno korištenje. Osim toga, ova metodologija je relativno nova te je stoga nedovoljno korištena i razrađena, a time je teža za razumijevanje. Prema (Stapleton, 1997) nedostaci su slijedeći:

- Usmjerenost na RAD može dovesti do smanjenja robusnosti koda
- Zahtijeva potpunu predanost DSDM procesu
- Zahtijeva značajno sudjelovanje korisnika
- Zahtijeva stručan razvojni tim kako u poslovnom tako i tehničkom području
- Definira nekoliko artefakata i proizvoda rada za svaku fazu projekta; teža dokumentacija.
- Konzorcij kontrolira pristup materijalu i mogu se naplaćivati naknade samo za pristup referentnom materijalu.

4.4.5. Razvoj vođen funkcionalnostima

Razvoj vođen funkcionalnostima (eng. *Future Driven Development)* nastao je 1997. godine od strane Jeffa De Luce i Petera Coada. Metoda je nastala kada su osnivači (zajedno s drugim kolegama) shvatili da s tradicionalnim metodama koje su koristili neće stići napraviti veliki softverski projekt na kojem su radili. Tada su zajedničkim snagama osmislili metodu FDD koja je posjedovala „*taman dovoljno procesa da bi osigurao skalabilnost i ponovljivost te istovremeno poticao kreativnost i inovativnost“*. Prvo tiskano izdanje bilo je objavljeno u knjizi *Java Modeling in Color with UML*, a autor je bio Peter Coad (Goyal, 2008).

FDD sadrži karakteristike zajedničke svim agilnim metodama od kojih se ističu važnost timskog rada, fleksibilnost, iterativnost, stvaranje inkrementalnog proizvoda i druge. Metoda nalaže korištenje UML dijagrama kojima se prikazuju klase sustava te njihov odnos u sustavu. Fokus je na dizajnu i implementaciji, iteracije traju do dva tjedna, dok je glavni cilj postizanje kvalitete aktivnosti pri razvoju programskog proizvoda.

FDD ne pokriva cijeli proces razvoja softvera, već je naglasak samo na dizajnu i implementaciji. Sastoji se od pet sekvencijalnih procesa i pruža metode, tehnike i smjernice za uspješnu i kvalitetnu provedbu projekta. Kako samo ime i nalaže, funkcionalnosti (svojstva,

osobine) su primarni aspekt zahtjeva i *input* u planiranju. Metodologija uključuje definiciju uloga i ciljeva koji se trebaju postići do određenog vremenskog roka.

Slika 14: FDD proces razvoja (Prema: Roggio, 2007)

Na slici su prikazane faze u procesnom razvoju prema FDD metodologiji. Prve tri se izvode slijedno dok su dizajn i implementacija iterativne. U nastavku slijedi objašnjenje faza.

Razvijanje cjelokupnog modela prva je faza koja se provodi tijekom razvijanja projekta. Poslovni eksperti (klijenti, korisnici ili poslovni analitičari) su svjesni opsega, konteksta i zahtjeva sustava. U ovoj fazi se razvija kako i što sustav treba raditi sa ciljem razumijevanja osnovnih zahtjeva sustava. Eksperti predstavljaju tzv. prohod (engl. *walkthrough*) gdje se članovi tima informiraju o višim razinama sustava.

Izrada liste funkcionalnosti slijedeći je korak u razvojnom procesu. Razvojni tim prema *inputima* iz razvoja modela kreiraju listu funkcionalnosti koje projekt mora sadržavati. Nakon napravljenog popisa, on se prezentira korisnicima koji daju svoje povratne informacije te na kraju i odobravaju prelazak na slijedeću fazu.

Planiranje prema funkcionalnostima uključuje kreiranje detaljnog plana razvoja gdje su skupovi funkcionalnosti predstavljeni i dodijeljeni developerima. Svaki programer postaje vlasnik klasa i njihov zadatak je razvijati dodijeljene klase.

Dizajn i implementacija prema funkcionalnostima je glavni dio FDD procesa. Aktivnosti koje se izvršavaju u ovim fazama su modeliranje, dizajniranje, programiranje i testiranje kreiranih klasa. Dizajn i implementacija su iterativne procedure koje mogu trajati od nekoliko dana do maksimalno dva tjedna. Mogu postojati istodobno više različitih razvojnih timova koji razvijaju sustav u ovim fazama.

4.4.5.1. Prednosti i nedostaci FDD metodologije

S obzirom da je FDD nastao za vrijeme izrade velikog projekta, logično je zaključiti da je prigodan i koristan pri izradi opsežnih i zahtjevnih projekata. Proces je poprilično jednostavan za razumijevanje i primjenu. Kao i kod drugih agilnih metoda prilagodljivost i kvalitetan proizvod samo su neke od prednosti. Prema (Wikidot, 2011) prednosti su slijedeće:

- Vodi prema velikim projektima sa konstantnim zadržavanjem uspjeha.
- Zbog jednostavnosti procesa olakšano je uključivanje novih osoba na projekt.
- Razvoj vođen funkcionalnostima temelji se na skupu najboljih priznatih praksi. Redovite izgradnje omogućavaju da je sustav uvijek ažuran i spreman za prezentaciju klijentu. Tako se rano mogu uočiti i ispraviti greške, poboljšati sustav i sl.
- Vidljivost rezultata omogućen je čestim i preciznim izvještajima o napretku na svim razinama (unutar i izvan projekta). Temeljeno isporučenim funkcionalnostima, menadžeri lakše upravljaju projektom.
- Zbog iterativnog dizajna i implementacije, smanjen je rizik od neuspjeha
- Zbog faze razvijanja modela dobiva se na boljem i jasnijem razumijevanju zahtjeva i cjelokupnog sustava koji se gradi.
- Trošak projekta se temelji prema funkcionalnostima što omogućuje veću točnost procjene budžeta.

Nedostaci FDD metodologije uključuju neprimjenjivost na manje projekte, nedovoljno dokumentiranje i nedovoljno objašnjeni iterativni proces u odnosu na ostale agilne metodologije. Visoka je ovisnost o glavnom programeru koji djeluje kao koordinator, mentor i glavni dizajner. Zbog podjele klasa svakom zasebnom programeru nema zajedničkog vlasništva koda čime se gubi na transparentnosti i mogućnosti zajedničkog rada i ispravaka. Ova metodologija se temelji na modelu sustava, što može izazvati probleme ukoliko se primjenjuje na neki stari sustav koji ga nema definiranog.

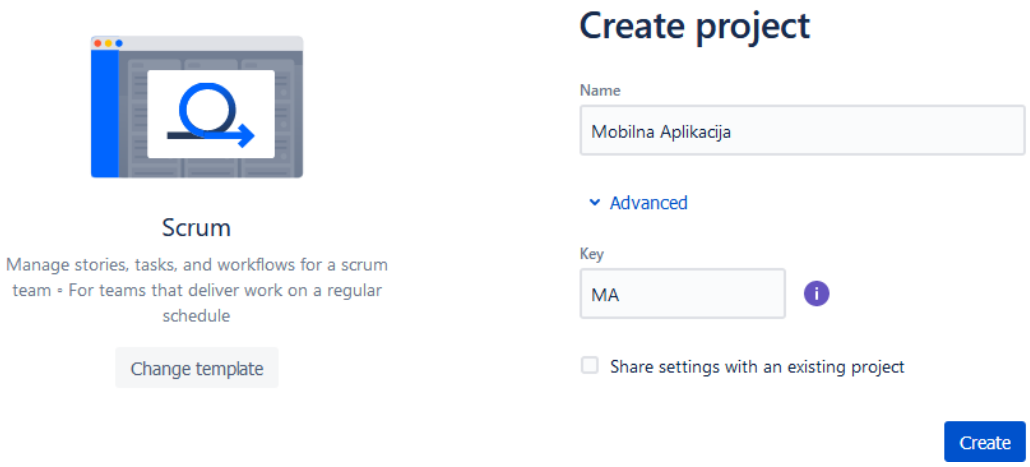
5. Primjer upravljanja razvojem softvera primjenom SCRUM metodologije korištenjem alata JIRA

SCRUM trenutno slovi kao najpoznatija a time i najčešće korištena agilna metodologija. Zbog jednostavnosti i širokog spektra prednosti pri razvoju softverskih rješenja, sve više poduzeća odlučuje se primijeniti ovu metodologiju. Na tržištu danas postoji veliki broj alata koji su podrška pri upravljanju softverskim projektom gdje se po popularnosti i pozitivnim recenzijama najviše ističe Jira. U ovom poglavlju bit će predstavljen prvi *sprint* Scrum procesa za izradu jednostavne mobilne aplikacije u navedenom alatu. Aplikacija služi osobama koje putuju i rade na udaljeno, kako bi mogle pronaći mjesta za rad s dostupnim internetom.

5.1. Pokretanje projekta

Jira je alat za projektni menadžment napravljen od strane tvrtke Atlassian. Alat je namijenjen projektnim timovima koji na jednom mjestu mogu pratiti projektne aktivnosti, greške i probleme, izvještavati i postavljati dokumentaciju, analizirati, integrirati kôd i dr. Atlassian ima još nekoliko alata koji služe kao podrška u razvoju projekata koji se mogu integrirati s Jirom – to su primjerice BitBucket za integraciju s kôdom i Confluence za dokumentaciju. Sama Jira dijeli se na nekoliko različitih vrsta kao što su Jira Agile za agilne projekte i Jira Capture koja služi primarno testerima za praćenje grešaka i *bug-ova.* Za primjer ovog projekta korištena je Jira Agile s obzirom da Scrum pripada agilnim metodama.

Pri samoj izradi projekta alat nudi predloške s obzirom na njegovu vrstu. Mogu se odabrati Kanban, Scrum, Bug tracking, Task tracking, itd. te se u ovom slučaju odabire predložak za Scrum metodologiju. Nakon odabira predloška otvara se ekran za definiranje imena projekta te kako će izgledati njegova skraćenica u samom alatu. Za primjer predstavljenog projekta odabrano je jednostavno ime „Mobilna aplikacija“ dok bi se u većim sustavima trebalo koristiti ime samog projekta za lakše raspoznavanje. Na sljedećoj slici možemo vidjeti izgled ekrana za kreiranje projekta. Primjećujemo da je korisničko sučelje alata iznimno jednostavno i navigacija je intuitivna.



Slika 15: Odabir predloška i kreiranje projekta

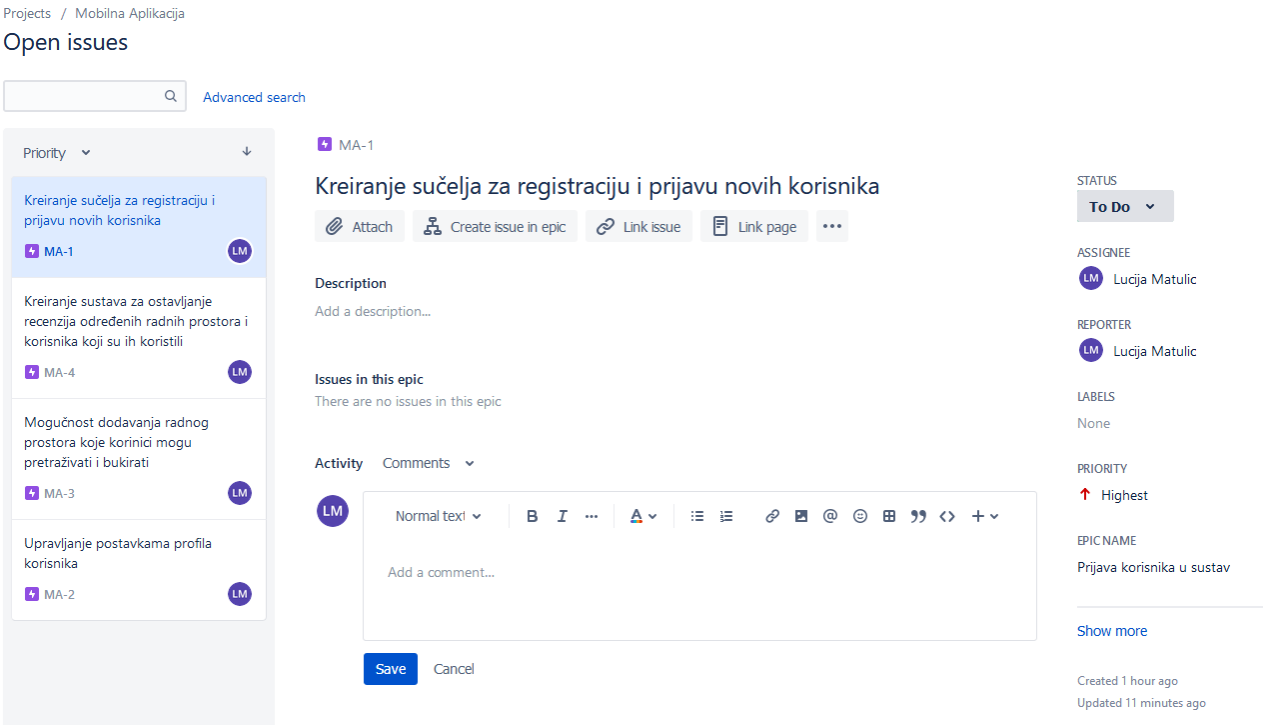
Nakon što smo kreirali projekt otvara se odabrani predložak za Scrum proces u alatu. Pri samoj inicijaciji projekta u Scrum procesu definiraju se opći ciljevi, određuje se projektni tim, definira se voditelj projekta (*Scrum Master*) te se identificiraju dionici na projektu. Glavni cilj pri inicijaciji projekta u Scrum procesu preko alata Jire je definicija *Epic-a* i prioretiziranog *Product Backlog-a* (lista zadataka projekta). Zadaci na projektu se u alatu definiraju kao *„issue“* i postavljaju se na listu zadataka projekta. Oni mogu biti kategorizirani kao:

- **Ep** *(eng. Epic)* – generalni *Use Case* projekta koji je skupina više korisničkih priča (eng. *User Stories)*
- **Priča** *(eng. Story)* – Funkcionalni zahtjev koji je izražen iz perspektive korisnika, poznat pod imenom korisnička priča. Svaka priča se razlaže na manje zadatke – *Sub-task-ove.*
- **Podzadatak** (eng. *Sub-task)* – Zadaci koji moraju biti izvršeni kako bi bila zadovoljena jedna korisnička priča.
- **Zadatak** *(eng. Task)* – Zadatak koji nije direktno vezan za korisničku priču ali je nužan kako bi se ostali zadaci mogli provesti. To je najčešće kreiranje Git repozitorija za pohranu kôda i slično
- **Greška** *(eng. Bug)* – Problem koji ometa funkcionalnost proizvoda ili usluge. On se definira u reviziji

Sljedeća slika predstavlja grafički prikaz odnosa epova, korisničkih priča i podzadataka. Jedan ep sadrži više priča dok priče sadrže podzadatke koji su programerske prirode i služe kako bi se korisničke priče ostvarile.

Slika 16: Odnos zadataka u Scrum procesu u alatu Jira (Prema: Rehkopf, bez dat.)

Dakle kao prvi korak u kreiranju projekta, potrebno je definirati epove. To su veći funkcionalni dijelovi koji čine čitav projekt. Oni ne moraju biti izvršeni u jednom *sprintu,* odnosno jedan *Epic* može biti dovršen kroz nekoliko *sprintova.* Za izradu ove aplikacije definirani su sljedeći epovi: Prijava korisnika u sustav, Korisnički profil, Radni prostori i Sustav recenzija. Svaki *Epic* ima ime, opis, osobu kojoj je dodijeljena te prioritet. Mogu se dodavati komentari, razni prilozi te se može kreirati novi *issue* unutar *Epic-a* (tako dodajemo *Story* pojedinom *Epic-u).* Na sljedećoj slici možemo vidjeti ekran sa otvorenim zadacima (eng. *Open issues)* u kojem se trenutno nalaze definirani *Epic-i.*



Slika 17: Definirani *Epic* funkcionalnosti projekta

Navedene glavne funkcionalnosti određuju vlasnik proizvoda (*Product Owner)* zajedno sa klijentima koji su naručitelji aplikacije. Nakon što su sve glavne odrednice definirane, vrijeme je za sastanak s razvojnim timom.

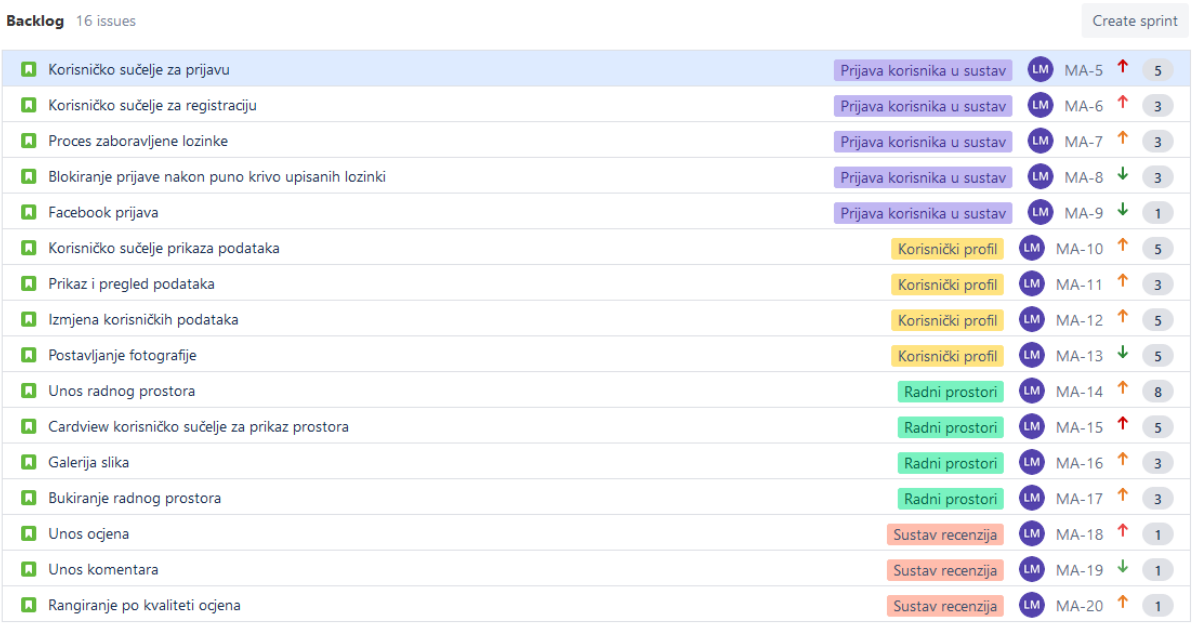
5.2. Prvi sastanak (eng. *Kick-off meeting)*

Prvi sastanak nakon što je projekt inicijaliziran i njegove glavne odrednice su determinirane je *Kick-off* sastanak sa razvojnim timom. Na ovom sastanku razvojni tim se upoznaje s projektom, definiraju se glavni rokovi te se razgovara o ostalim osnovnim stavkama projekta. Na sastanku su prisutni svi sudionici Scrum procesa: klijent (naručitelj projekta), voditelj projekta (*Scrum Master),* vlasnik proizvoda (*Product Owner)* i razvojni tim. Dužnost vlasnika proizvoda na ovom sastanku je kvalitetno i sažeto predstaviti projekt i što se njime želi postići. Razvojni tim postavlja eventualna pitanja naručitelju i vlasniku proizvoda kako bi imali što bolje razumijevanje projekta. Dogovaraju se izmjene među glavnim funkcionalnostima ukoliko se vidi potreba, definiraju se pravila za *sprtinove*, termini dnevnih sastanaka, alati za komunikaciju s naručiteljem i ostale glavne stavke. Ovaj sastanak održava se samo jednom kao uvod u projekt te često ne traje dugo.

5.3. Kreiranje liste zadataka projekta (eng. *Product Backlog)*

Nakon što su se svi sudionici Scrum procesa upoznali s projektom dolazi trenutak za definiciju liste zadataka. U nekim projektima se očekuje od vlasnika proizvoda da pred razvojni tim dovede gotovu listu s definiranim prioritetima, dok se u nekim projektima ova lista kreira zajedno s razvojnim timom. S obzirom da je velika odlika Scrum metodologije da tim sam određuje što će raditi, kako i do kada; proces definiranja liste zadataka najčešće se radi zajedno. Trenutno se u alatu nalaze samo definirani epovi u listi zadataka i potrebno je dodati korisničke priče, njihove podzadatke za uspješno izvršenje te zadatke za programere koje nisu direktno vezane za korisničke priče.

Na sljedećoj slici prikazan je definiran *Product Backlog.* Zelena oznaka s lijeve stane naziva priče upravo označava da se radi o *User Story-ju.* S desne srane imena priče piše kojem epu pripada, koja osoba je zadužena za zadatak, kratka oznaka zadatka, prioritet (mogućnosti su *Lowest, Low, Medium, High, Highest),* te je na kraju „broj bodova“ zadatka.



Slika 18: Lista zadataka projekta

U slučaju epa prijave korisnika u sustav, priče su podijeljene na izradu korisničkog sučelja za prijavu i registraciju kao definiranje izrade obrazaca putem kojih korisnik može ispuniti navedene radnje. Slijedi proces zaboravljene lozinke kako bi se kreirao slijed aktivnosti u aplikaciji u slučaju da korisnik zaboravi lozinku te blokiranje računa ukoliko lozinka previše puta bude krivo upisana. Zadnja priča u ovom epu je integracija Facebook prijave što je danas česta praksa pri izradi sučelja za prijavu, kako bi se korisniku proces olakšao. Nakon toga su definirane priče za ep Korisnički profil koje se odnose na izradu sučelja za prikaz podataka, omogućavanje izmjene istih te postavljanje fotografije. To su sve specifikacije tipične za mogućnost upravljanja kreiranim profilom. Sljedeće priče vezane su za korisnike koji u aplikaciju žele postavljati radne prostore koje druge osobe mogu rezervirati. Priče u ovom epu se odnose na izradu sučelja za unos i pregled prostora, kreiranje galerije slika te mogućnost rezervacije određenog prostora. Zadnji ep vezan je za sustav recenzija gdje su priče podijeljene na unos ocjena, komentara te rangiranje poslovnih prostora prema ocjenama. Tako će korisnici moći pretraživati radne prostore prema visini njihove kvalitete i rezervirati one koje im najviše odgovaraju.

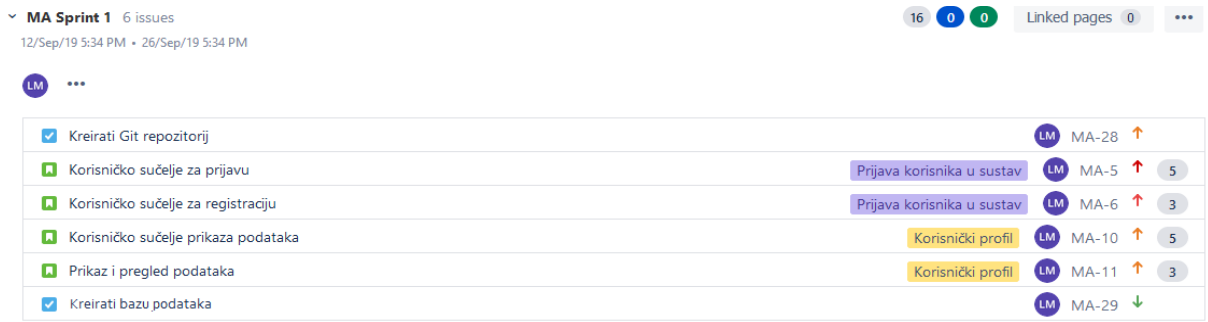
Bodovi zadatka (*Story points)* određuju koliko je taj zadatak težak. Brojevi se određuju onako kako se tim dogovori no najčešće se koriste brojevi Fibonaccijevog niza. Postoje razne metode za određivanje težine zadatka no najčešća je da za svaki zadatak posebno svaki član tima sebi (putem neke od mobilnih aplikacija koje za to postoje ili jednostavno u bilješku, na papri i sl.) napiše broj koji smatra da predstavlja težinu zadatka. Nakon što svi osobno napišu broj, odbroji se tri sekunde i svi istovremeno pokažu koju težinu su odredili. Ako se dogodila

situacija gdje je netko za zadatak npr. postavio težinu 1 (najlakše) a netko 8 (teško) tada se o tome raspravlja – zašto određeni član misli jedno a drugi drugo. Na kraju se nakon zajedničkog dogovora postavljaju bodovi koji označavaju konačnu težinu zadatka. Težina je važna za odrediti zato što time lakše definiramo koliko je vremena potrebno da se odradi i koliko je članova potrebno za izvršenje.

5.4. Planiranje *sprinta*

Nakon što je definiran *Product Backlog* može započeti planiranje sprinta. *Product Backlog* se definira samo na početku ali se može dorađivati i uređivati što se često naziva *grooming*. *Sprint* se planira svaki puta prije nego što započne odnosno nakon što prethodni završi. Na sastanku su prisutni svi članovi Scrum tima te on najčešće traje jedan do dva sata, ovisno o kompleksnosti specifikacija koje se moraju dovršiti u tom *sprintu*. Ishod svakog *sprinta* mora biti funkcionalan inkrement proizvoda koji se može testirati i koristiti. Prethodno su se definirale priče a sada se definiraju podzadaci koji se trebaju izvršiti da bi se te priče ostvarile. Kao što se moglo i na pričama, tako se i na zadacima određuju njihove težine, tko ih izvršava, koliko procjenjuju da će trajati, i sl. Tim može definirati koje će priče i zadatke izvršavati po određenom sprintu, ne moraju ići slijedno ili pripadati istom epu. Njihov redoslijed, prioritet i vrijeme izvršavanja određuju sami prema svojim mogućnostima i procjenama (osim ako vlasnik proizvoda ne nalaže drugačije). Budući da je teško predvidjeti sve zadatke koji trebaju biti riješeni tijekom sprinta, razvojni tim ima pravo tijekom sprinta nadopunjavati zadatke ovisno o potrebi.

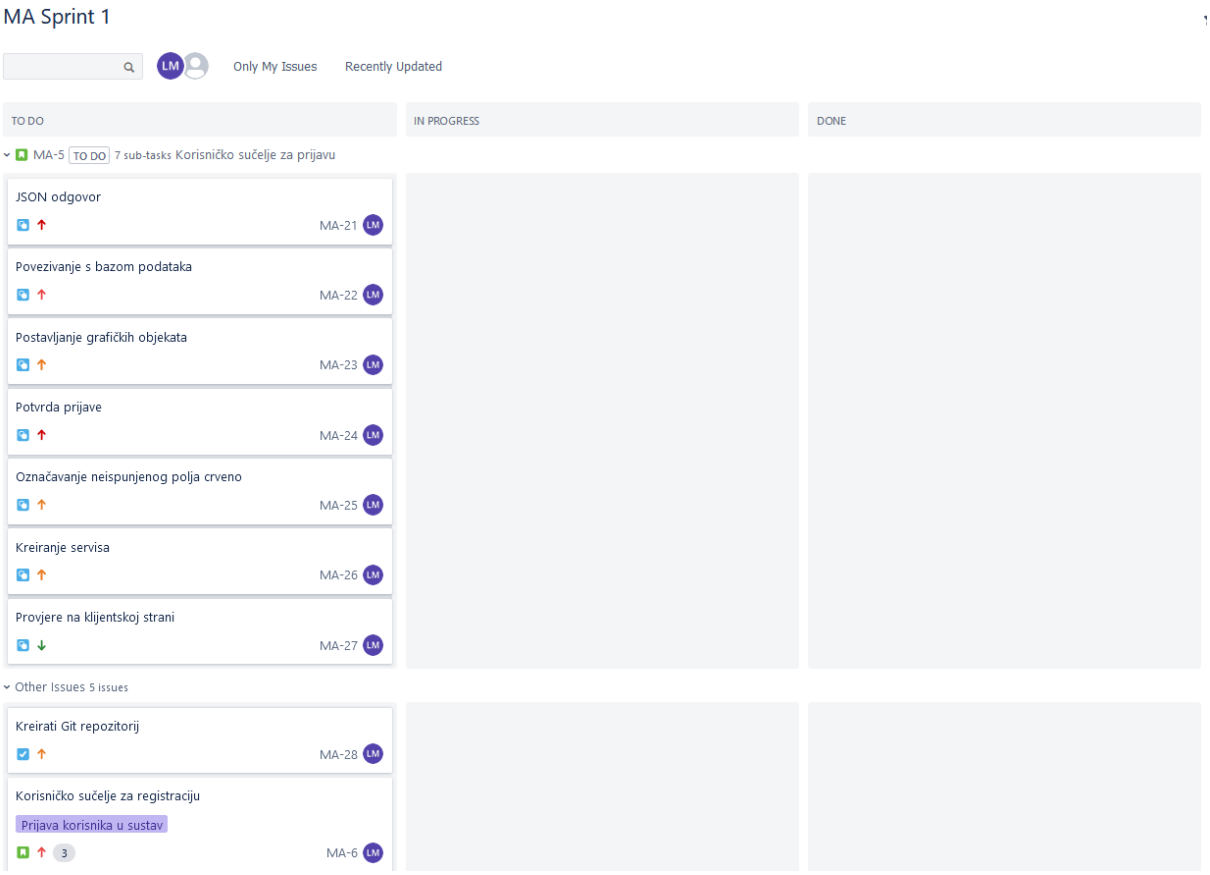
Sastanak planiranja sprinta završava kada su definirane korisničke priče i njihovi podzadaci koji će se izvršavati za taj sprint. Ukoliko se prijeđe dogovoreno vrijeme trajanja sastanka, nedogovoreni zadaci prebacuju se za slijedeći sprint. Nakon što je sve dogovoreno, u alatu se postavljaju priče i podzadaci te se pokreće sprint. Na sljedećoj slici prikazane su korisničke priče i zadaci određeni za prvi sprint u ovom projektu.



Slika 19: Prvi *Sprint Backlog*

Plave oznake predstavljaju zadatke (*Task)* koji nije vezan za izvršenje priče. To su često vezani za kreiranje repozitorija, nabava licenci i slično. Unutar samih priča nalaze se podzadaci koje je potrebno izvršiti da bi se ta priča ostvarila. Priče i zadaci odabrani za ovaj sprint odabrani su prema potrebama klijenta koji u prvom inkrementu želi vidjeti prijavu korisnika te mogućnost pregleda korisničkih podataka. Iz tog razloga su preuzete priče iz dva različita epa te dva zadatka nevezana strogo za same priče već za omogućavanje njihova izvršenja. To su kreiranje Git repozitorija i baze podataka kako bi se kod i podaci spremali na odgovarajuća mjesta.

Prilikom pokretanja *sprinta*, u Jiri se oni postavljaju u kategoriju *Active Sprints* gdje su priče sa podzadacima i zadaci postavljeni na virtualnu Scrum ploču. Prikaz možemo vidjeti na sljedećoj slici gdje se ističe priča „Korisničko sučelje za prijavu“ sa svojim podzadacima.



Slika 20: Scrum ploča nakon planiranja *sprinta*

Ploča je podijeljena na tri dijela: *TO DO, IN PROGRESS* i *DONE.* S obzirom da je sada tek završilo planiranje sprinta, svi zadaci nalaze se u koloni *TO DO.* Podzadatke možemo određivati po prioritetu, dodijeliti odgovorne članove, odrediti vrijeme trajanja te dodavati *fix versions* – verzije kojima smo ispravili neku grešku u zadatku.

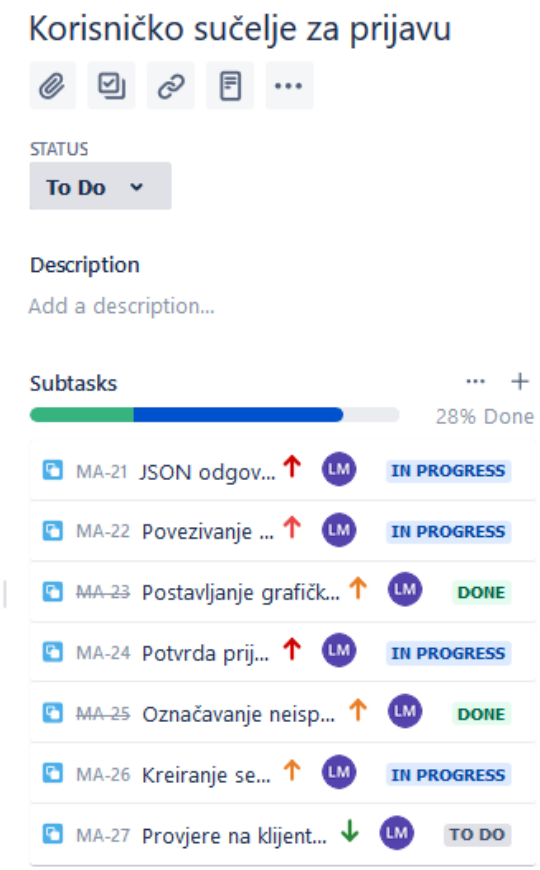
5.5. Izvršenje *sprinta*

Sprint je vremenski determiniran period u kojem se zabilježeni zadaci trebaju izvršiti i nakon čijeg završetka dobivamo inkrement uporabljivog proizvoda. Tijekom trajanja *sprinta* svaki član tima u svakom trenutku zna što trenutno radi ili treba napraviti. Budući da su zadaci poredani po prioritetima prvo se izvršavaju oni koji će na kraju sprinta tvoriti inkrement proizvoda koji se može testirati. Svi ostali zadaci koji su manjeg prioriteta i ne utječu uvelike na dobivanje uporabljivog inkrementa, ukoliko nisu izvršeni prebacuju se za sljedeći *sprint*. Postavljanjem vremenskog okvira za izvršenje zadataka olakšava voditelju projekata upravljanje vremenom i opsegom projekta.

Uloga vlasnika proizvoda za vrijeme *sprinta* je da cijelo vrijeme dostupan, prisutan i uključen u aktivnosti koje se izvršavaju. Odgovara na pitanja o tome kako stvari trebaju funkcionirati i izgledati, kao i praviti eventualne kompromise kada je to potrebno. Ima ulogu motivacije i komunikacije s članovima tima. On je taj koji najbolje poznaje ciljeve projekta te stoga mora uvijek usmjeravati članove ka njihovim ostvarenjima. Za vrijeme *sprinta* novi zadaci i izmjene se ne bi trebale raditi, već tek nakon što *sprint* završi.

Uloga *Scrum Mastera* tijekom *sprinta* je da maksimizira produktivnost te da riješi sve probleme između *inputa* koje daje vlasnik proizvoda i razvojnog tima. *Scum Master* ima veliku ulogu pri obavljanju dnevnih Scrum sastanaka (eng. *Daily Scrum Meetings).* Provode se svaki dan, pri samom početku rada te traju kratko, oko petnaestak minuta. Na dnevnim sastancima svaki član odgovara na pitanja što je do sada napravio, što sada kani raditi te ima li problema. Ovim sastancima se postiže transparentnost – svi prisutni imaju uvid u širu sliku stadija projekta te znaju koliko su bliže ciljevima. Ovaj sastanak služi kako bi si članovi međusobno pomogli i dobili uvid u međusobni napredak.

Sljedeća slika prikazuje izgled *Story-ja* izrade korisničkog sučelja za prijavu i njegovih podzadataka za vrijeme *sprinta.*



Slika 21: Prikaz korisničke priče za vrijeme sprinta

Primjećujemo da su neki zadaci još uvijek u koloni *IN PROGRESS*, odnosno trenutno se odvijaju, neki su gotovi odnosno nalaze se u koloni *DONE* te oni koji se još nisu počeli raditi.

S obzirom na broj zadataka koji se nalaze u pojedinoj fazi, možemo vidjeti grafički prikaz razvoja priče. Gotovi zadaci označeni su zeleno, oni koji se trenutno izvršavaju su plavi a nezapočeti su sive boje. Pokraj piše i postotak izvršenih zadataka za tu priču. U nastavku opisa selekcionirane priče može se pratiti vrijeme. Tijekom planiranja postavljena je estimacija trajanja izvršenja zadataka za određenu priču. Tijekom same provedbe *sprinta* i izvršenja zadataka prati se stvarno vrijeme koje je bilo potrebno za izvršenje. U pregledniku sa strane možemo vidjeti koliko vremena nam je preostalo ili koliko smo vremena viška utrošili na same zadatke.

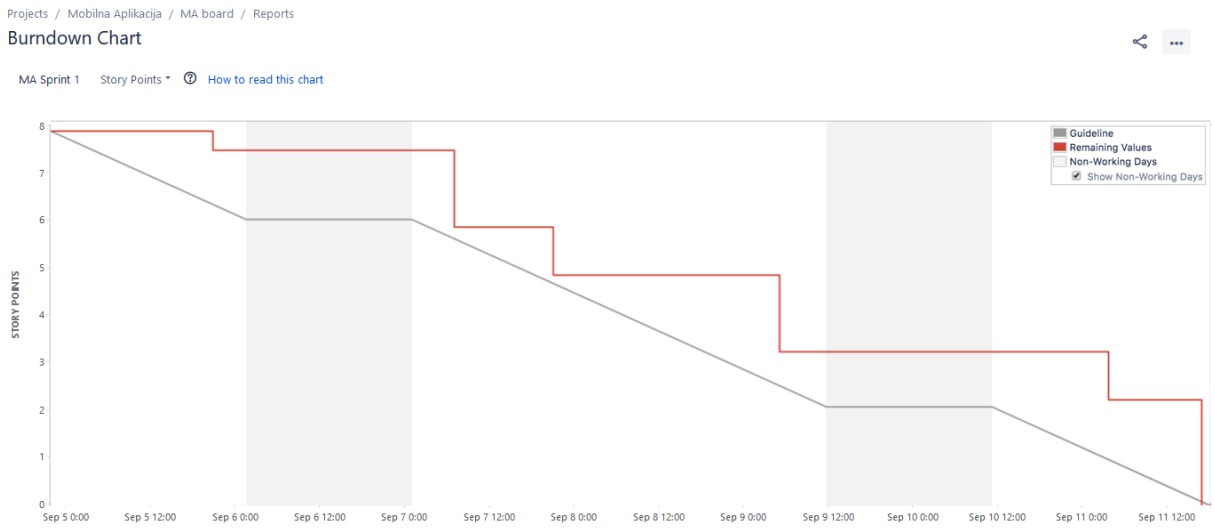
5.6. Pregled *sprinta*

Pregled *sprinta* aktivnost koja se provodi neposredno prije samog kraja. Tijekom pregleda provjeravaju se dosad napravljene funkcionalnosti proizvoda, odnosno inkrementa. Cilj je raspraviti što je napravljeno i dobiti povratne informacije od ostalih dionika.

Osobe koje sudjeluju na pregledima su razvojni tim, vlasnik projekta te interesne strane koje vlasnik projekta poziva. Prema (Schwaber i Sutherland, 2013) pregled *sprinta* uključuje sljedeće aktivnosti:

- Vlasnik projekta prezentira koje su stavke s *Product Backloga* završene i koje nisu. Na osnovi dosadašnjeg napretka može već predodrediti datume kada se mogu očekivati implementirane nove funkcionalnosti.
- Razvojni tim diskutira o iskustvima tijekom *sprinta* – probleme i zastoje na koje su naišli te kako su ih riješili. Osim toga prezentiraju dosad napravljeno i odgovaraju na pitanja vezana uz kreirani inkrement.
- Svi sudionici komentiraju što se slijedeće treba napraviti i te informacije služe kao osnova za sljedeći *sprint planning.*
- Ocjenjuju se promjene na tržištu te zadovoljava li proizvod trenutne potrebe. Zbog raznih vanjskih utjecaja i okolnosti, definira se što bi se sljedeće trebalo napraviti da proizvod ostvari konkurentsku prednost.
- Pregledavaju se vremenski okvir, budžet, prilike i tržište za narednu planiranu verziju proizvoda.

Nakon pregleda *sprinta,* voditelj tima i vlasnik proizvoda imaju uvid u izvještaje *sprinta.* U Jiri se oni mogu pronaći i generirati na vrlo jednostavan način u kategoriji *Reports.* Nakon svakog *sprinta* onaj najčešće korišteni je dijagram napretka (eng. *Burndown chart).* On pokazuje koliko brzo razvojni tim dovršava zadatke vezane za korisničke priče. Na sljedećoj slici je prikazan dijagram napretka za prvi dovršeni *sprint.*



Slika 22: Dijagram napretka nakon prvog *sprinta*

Siva linija označava idealan prikaz napretka odnosno idealno potrošeno vrijeme, dok je crvena linija stvarni napredak odnosno stvarno potrošeno vrijeme za izvršavanje zadataka. Grafikon jasno daje do znanja da su članovi tima uspjeli privesti kraju navedeni *sprint* te da je količina truda (eng. *estimate of effort*) svedena na nulu što znači da su svi zadaci uspješno dovršeni do kraja *sprinta*.

7. Retrospektiva *sprinta*

Prema (Schwaber i Sutherland, 2017) retrospektiva je prilika za članove tima da se kontroliraju i donesu plan unaprijeđena koji će se provesti tijekom sljedećeg *sprinta.*

U pregledu *sprinta* fokus je bio na proizvodu i dosad napravljenim funkcionalnostima dok je u retrospektivi fokus na Scrum procesu i što se može napraviti da bi se poboljšao. Tim raspravlja o svemu što utječe na način na koji se proizvod razvija, što uključuje sve procese, prakse, pravila, vrstu i način komunikacije, okolne uvjete, artefakte i alate. Ova aktivnost je vrlo važna jer omogućuje timu da napreduje i razvija se a da tako i što bolje radi i razvija proizvod. Kvalitetan rad donosi kvalitetniji proizvod, što je vrlo značajna odlika agilnih metodologija. Dok se kod nekih vrsta metoda retrospektiva izvršava tek na samom kraju čitavog projekta, u Scrum metodici odvija se nakon svakog *sprinta.* I za ovu aktivnost postoje razni načini i metode provedbe, među kojima je najpoznatija morska zvijezda u kojoj su definirani pet glavnih elemenata:

- **Prestati** (engl. *Stop*) – Aktivnosti koje ne donose vrijednosti timu ili naručitelju te ih treba prestati raditi.
- **Manje** (engl. *Less*) – Aktivnosti za koje je potreban veći napor, a ne donose nikakvu korist klijentu ili timu.
- **Nastaviti** (engl. *Keep*) – Dobre aktivnosti ili prakse koje tim želi nastaviti raditi. Takve aktivnosti se već primjenjuju i potrebno ih je nastaviti raditi.
- **Više** (engl. *More*) – Aktivnosti na koje se tim treba više fokusirati ili obavljati češće.
- **Početi** (engl. *Start*) – Aktivnosti ili ideje koje tim nije do sada radio, a smatra da će donijeti korist timu i/ili klijentu.

Nakon retrospektive provodi se ponovo cijeli proces od dotjerivanja *Product Backloga*, planiranja novog *sprinta* do kreiranja svaki puta za jedan inkrement bogatijeg proizvoda. Na kraju Scrum procesa dobiva se kvalitetan finaliziran proizvod.

6. Zaključak

U današnjem vremenu kompleksnih i visoko obuhvatnih projekata, kvalitetno vođenje i upravljanje projektnim zadacima postaje iznimno važno. Kroz vrijeme znanstvenici i stručnjaci iz projektnog menadžmenta dolazili su do novih saznanja i rješenja kako što bolje iskoristiti tehnike i alate da bi se projektni ciljevi ostvarili na vrijeme i na zadovoljstvo svih dionika. Razvijene su razne metodologije kojima se definiraju procesi, faze, pravila i prakse kako pristupiti kvalitetnom vođenju projekata koji donose novi, kvalitetan proizvod ili uslugu.

Tradicionalnim metodama smatraju se sve one koje imaju uporište u vodopadnom modelu. To su metode u kojima se projekt razlaže na slijedne faze, gdje je fokus na procesima i obavljanju ciljeva u zadanom vremenu i budžetu. Najveća prednost ovih metodologija je da su unaprijed poznati svi zadaci i faze kroz koje će projekt proći te stoga nema puno rizika jer se zna točno što će se odvijati. Problemi na koje su naišli voditelji projekata novijeg doba jest da su ove metode dosta strogo definirane, nisu fleksibilne i mnogo se vremena troši na iscrpna planiranja i pisanje dokumentacije. Zbog naglih promjena u okolini, sve većih i složenijih projekata, osmišljene su suvremene – agilne metode. Ime su dobile upravo zbog svoje prilagodljivosti i fleksibilnosti, a u odnosu na tradicionalne metode projekti su razloženi na iterativne a ne slijedne faze. Njihova prednost očita je u tome da na prvo mjesto stavljaju čovjeka i njegove mogućnosti i potrebe a ne strogo praćenje plana. Timovi su samostalni i oslobođeni raditi prema svojem tempu, naručitelj je uključen u proces i češće dobiva gotove dijelove proizvoda. Time se uvelike dobiva na kvaliteti proizvoda zato što se za vrijeme njegove izrade mogu uvesti promjene i ispraviti greške. Agilne metodologije najčešće se koriste pri izradi softverskih rješenja zato što takvim vrstama projekata najviše odgovaraju. Njihova pojava ne označava da bi se tradicionalne metode trebale u potpunosti prestati koristiti, već da svako poduzeće posebno odabere onu metodu koja njihovom poslovanju najviše odgovara. Postoje poduzeća koja kombiniraju najbolje prakse iz više različitih metoda pa tako i istovremeno iz tradicionalnih i suvremenih. PMBOK vodič se pretežito odnosi na tradicionalni način vođenja projekata ali se smjernice napisane u knjizi mogu prilagoditi suvremenim metodama.

Na tržištu danas postoji veći broj alata koji služe kao potpora pri praćenju projekata. Jira Atlassian pokazala se iznimno intuitivnom i jednostavnom za korištenje sa uvidom svih aspekata Scrum metodologije. Lako je zaključiti zašto je jedan on najkorištenijih alata u svrhu praćenja projekata za izradu softverskih rješenja.

Popis literature

[1] Adell, L. (2013). *Benefits & Pitfalls of using Scrum software development methodology.* Preuzeto 30.8.2019. s http://blog.belatrixsf.com/wp-content/uploads/kalins-pdf/singles/benefits-pitfalls-of-using-Scrum-software-development-methodology.pdf.

[2] Alredainy, A., Beierer, J., Christoffels, M., Riechert, M., i Falconer, T. (2010). *Improving Advertisement Efficiency with Information Systems Design* (Projektno izvješće). Norderstedt: GRIN Verlag.

[3] Anderson, D., Carmichael, A. (2016). *Essential Kanban Condensed.* Seattle, Washington: Lean Kanban University Press.

[4] Babić, D. (2018). *Rješavanje projektnog zadatka* (Priručnik). Zagreb: Sveučilište u Zagrebu, Fakultet prometnih znanosti.

[5] Balaji, S. (2012). *Waterfall vs V-model vs Agile: A comparative study on SDLC*. Preuzeto 12.8. 2019. s https://www.jitbm.com/Volume2No1/waterfall.pdf

[6] Beck, K., Beedle, M., Bennekum, A. V., Cockburn, A., Cunningham, W., Fowler, M., … Sutherland, J. (2001). *The Agile Manifesto*. Salt Lake City.

[7] Brandenberger, J., i Konrad, R. (1970). *Tehnika mrežnog planiranja.* Zagreb: Tehnička knjiga.

[8] Buble, M. (2010). *Projektni Menagement.* Dugopolje: Minerva - Visoka poslovna škola.

[9] Charvat, J. (2003). *Project Management Methodologies: Selecting, Implementing, and Supporting Methodologies and Processes for Projects.* New Jersey: Wiley & Sons, Inc.

[10] Davidović, V. (2016). *Objektno orijentirane tehnologije II* (Nastavni materijal). Rijeka: Veleučilište u Rijeci.

[11] Dujanić, M. (2010). *Projektni menadžment* (Udžbenik Sveučilišta u Rijeci). Rijeka: Tiskara futura.

[12] Farkaš, A. (2015) *Suvremeni trendovi razvoja informacijskih sustava* (Diplomski rad). Pula: Sveučilište Juraja Dobrile u Puli, Fakultet ekonomije i turizma.

[13] Fertalj, K., Car, Ž., i Nižetić Kosović, I. (2016). *Upravljanje projektima* (Skripta). Zagreb: Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva.

[14] Ganev, P. (bez dat.). *Advantages and disadvantages of using Scrum, Kanban and Scruban for Software Development.* Sofia: University of Management.

[15] Goyal, S. (2008). *Major Seminar On Feature Driven Development*. Technical University Munich. Preuzeto 3.9.2019. s http://csis.pace.edu/~marchese/CS616/Agile/FDD/fdd.pdf

[16] Hiranabe, K.(2008). *Kanban applied to software development: From agile to lean* [Blog post]. Preuzeto 2.9.2019. s http://www.infoq.com/articles/hiranabe-lean-agile-kanban.

[17] Haughey, D. (2014). *A Brief History of Project Management* [Blog post], Preuzeto 28.8.2019. s https://www.projectsmart.co.uk/brief-history-of-project-management.php

[18] Highsmith, J., Cockburn, A. (2001). *The Business of Innovation*. Los Angeles: IEEE Computer.

[19] Ikonen, M. (2011). *Lean Thinking in Software Development: Impacts of Kanban on Projects.* (Doktorska disertacija). Helsinki: Department of Computer Science, University of Helsinki.

[20] Ivković, B., Popović, Ž. (2005). *Upravljanje projektima u građevinarstvu*. Beograd: Građevinska knjiga a.d.

[21] Info Novitas. (bez dat.). *Ekstremno programiranje (XP).* Preuzeto 1.9.2019. s https://www.info-novitas.hr/o-nama/metodologije-rada/ekstremno-programiranje-xp/

[22] Ionel, N. (bez dat.). *Critical analysys of the SCRUM Project Management Methodology.* Preuzeto 30.8.2019. s http://steconomiceuoradea.ro/anale/volume/2008/v4-management-marketing/077.pdf.

[23] Jacobson, S. (2002). *The Rational Objectory Process: A UML-based Software Engineering Process* [Blog post]. Preuzeto 10.8.2019. s http://www.iscn.at/select_newspaper/object/rational.html

[24] Jeffries, R., Anderson, A., Hendrickson, C. (2001). *Extreme Programming Installed.* Boston: Addison Wesley

[25] Kardum, I. (2010). *Priprema, pozor, SCRUM!* [Blog post]. Preuzeto 01.09.2019. s http://www.bug.hr/mreza/tekst/SCRUM-agilni-proces-razvoja-softvera/95217.aspx

[26] Krezner, H. (2013). *A Systems Approach to Planning, Scheduling, and Controlling.* New Yersey: Wiley.

[27] Kukhnavets, P. (2019). *How to run Scrum efficiently in 2019? Quick guide for beginners* [Blog post]. Preuzeto 2.9.2019. s https://habr.com/en/company/hygger/blog/455022/

[28] Manger, R. (2016). *Softversko inženjerstvo.* Zagreb: Element.

[29] Novoseltseva, E. (2016). *Benefits of Agile project management* [Blog post]. Preuzeto 20.8.2019. s https://apiumtech.com/blog/agile-project-management-benefits/

[30] O'Brien, J., & Plotnick, F. L. (1999). *CPM In Construction Management.* Mcgraw - Hill Professional.

[31] Omazid, M., Baljkas S. (2005). *Projektni menadžment*. Zagreb: Sinergija nakladništvo d.o.o.

[32] Palmquist, S., Lapham, M., i Miller, S. (2013). *Parallel Worlds: Agile and waterfall differences and similarities.* Preuzeto 26.8.2019. s https://www.researchgate.net/publication/266054729_Parallel_Worlds_Agile_and_Waterfall_Differences_and_Similarities

[33] Parkash, S. (2011). *Agile Development using Scrum.* Preuzeto 27.8.2019. s http://ecomcanada.wordpress.com/2011/12/06/agile-development-using-Scrum/

[34] Pavlić, M. (2011). *Informacijski sustavi.* Zagreb: Školska knjiga.

[35] Pawar, R., Mahajan, K. (2017) *Benefits and Issues in Managing Project by PRINCE2 Methodology.* International Journal of Advanced Research in Computer Science and Software Engineering. Preuzeto 20.7.2019. s http://ijarcsse.com/Before_August_2017/docs/papers/Volume_7/3_March2017/V7I3-0134.pdf.

[36] Phillips, J. (2004). *PMP Project Management Professional Study Guide*, McGraw-Hill.

[37] Project Management Institute (2000). *A Guide to the Project Management Body of Knowledge.* Newton Square, Pennsylvania.

[38] Project Management Institute (2008). *Vodič kroz znanje o upravljanju projektima - četvrto izdanje.* Zagreb: MATE.

[39] Projektura (2018). *Metodologija upravljanja projektima - Planiranje projekta (doseg).* Preuzeto 22.8.2019. s http://www.projektura.org/blog/documents/2018/08/ebook-upravljanje-projektima-planiranje-projekta-doseg.pdf

[40] Radigan, D. (bez dat.). *Kanban - How the kanban methodology applies to software development* [Blog post]. Preuzeto 2.9.2019. s https://www.atlassian.com/agile/kanban

[41] Rational (bez dat.). *Rational Unified Process: Best Practices for Software Development Teams.* Preuzeto 12.8.2019. s https://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf

[42] Rehkopf, M. (bez dat.). *Epics, Stories, Themes, and Initiatives* [Blog post]. Atlassian Agile Coach. Preuzeto 11.9.2019. s https://www.atlassian.com/agile/project-management/epics-stories-themes

[43] Roggio, Robert. (2007). *Process-Driven Software Development: An Approach for the Capstone Sequence.* Jacksonville, Florida: University of North Florida, School of Computing.

[44] Royce, W. (1970). *Managing the development od large software systems.* Clifton: IEEE WESCON.

[45] Shwaber, K., i Sutherland, J. (2017), Scrum vodič. Preuzeto 11.9.2019. s https://tododoingdone.com/vodic-kroz-scrum/

[46] Sharma, S., Sarkar, D., i Gupta, D. (2012). *Agile Processes and Methodologies: A Conceptual Study.* Preuzeto 30.8.2019. s https://www.researchgate.net/publication/267706023_Agile_Processes_and_Methodologies_A_Conceptual_Study

[47] Sousa, S. (2009). *The advantages and disadvantages of RUP software development*, Preuzeto 2.8.2019. s http://www.my-project-management-expert.com/the-advantages-and-disadvantages-of-rup-software-development.html

[48] Stapleton, J. (1997). *Dsdm: The Method in Practice.* Boston: Addison-Wesley Longman Publishing Co.

[49] Strahonja, V. (2015). *Metodologija informacijskih sustava.* Projektiranje informacijskih sustava [Moodle]. Sveučilište u Zagrebu, Fakultet organizacije i informatike, Varaždin

[50] Stojcetović, B., Lazarević, D. Prlinčević, B., Stajčić, D., Miletić, S. (2014). *Project Management: Cost, Time and Quality.* Preuzeto 30.8.2019. s https://www.researchgate.net/publication/305462896_Project_managament_cost_time_and_quality

[51] Špundak, M. (bez dat.). *Upravljanje projektima – definicije i metodologije* (Članak). Zagreb: Fakultet elektrotehnike i strojarstva. Preuzeto 10.08.2019. s https://www.fer.unizg.hr/_download/repository/kvalifikacijski_clanak.pdf

[52] Talented Tester Support (2018). *Difference between a Waterfall Iron Triangle & Agile Triangle?* [Blog post]. Preuzeto 30.8.2019. s https://talentedtester.com/difference-between-waterfall-iron-triangle-agile-triangle/

[53] Topić, A. (2012). *Utjecaj Kanban agilne metodologije na razvoj programskog proizvoda* (Kvalifikacijski doktorski ispit). Split: Fakultet elektrotehnike, strojarstva i brodogradnje, Sveučilište u Splitu.

[54] Trainer, A. (2012). *PRINCE2 Process Diagrams* [Blog post]. Preuzeto 15.8.2019. s https://www.siliconbeachtraining.co.uk/blog/prince2-process-diagrams

[55] Tridibesh, S. (2003). *Scrum body of knowledge, SCRUMstudy.* Phoenix, Arizona.

[56] Turley, F. (bez dat.): *What is PRINCE2?* [Blog post]. PRINCE2 wiki. Preuzeto 3.8.2019. s https://prince2.wiki/extras/what-is-prince2/

[57] Westland, J. (2018). *History of Project Managemen* [Blog post]. Preuzeto 11.8.2019. s https://www.projectmanager.com/blog/history-project-management

[58] Zjakić, Z. (2013). *Agilni projektni menadžment* (Diplomski rad). Varaždin: Fakultet organizacije i informatike, Sveučilište u Zagrebu.

Popis slika

Popis tablica