Agilno planiranje i procjena u razvoju softvera

Valent, Antonio

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Antonio Valent

AGILNO PLANIRANJE I PROCJENA U RAZVOJU SOFTVERA

ZAVRŠNI RAD

Varaždin, 2023.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Antonio Valent

JMBAG: 0016141906

Studij: Poslovni sustavi

AGILNO PLANIRANJE I PROCJENA U RAZVOJU SOFTVERA

ZAVRŠNI RAD

Mentor:

Izv. prof. dr. sc. Zlatko Stapić

Varaždin, rujan 2023.

Antonio Valent

Izjava o izvornosti

Izjavljujem da je moj završni 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.

Autor potvrdio prihvaćanjem odredbi u sustavu FOI-radovi

_____

Sažetak

Agilne metode razvoja softvera odgovor su na bolju prilagodbu na stalne promjene u nestabilnom radnom okruženju, a za cilj imaju brže isporučivanje vrijednosti korisnicima. Uzevši u obzir izazove i nedostatke tradicionalnih metoda, prikazano je kako se agilnim metodama razvojni timovi mogu bolje prilagoditi vrlo promjenjivim zahtjevima industrije i nepredvidljivim situacijama koje ih mogu zadesiti te da budu fleksibilniji.

U uvodnom dijelu, prikazane su prednosti i nedostaci agilnih metoda i koncepata u poslovanju te isto tako usporediti ih s vodopadnim i ostalim pristupima koje je vrijeme, ali i tržište, pregazilo. Predstavljene su različite tehnike i alati koji se koriste prilikom planiranja agilnih projekata te kako se one primjenjuju na stvarne primjere u industriji. Primarni fokus je na agilnom planiranju, konceptima poput sprintova, popisa funkcionalnosti proizvoda i sprinta te prioritiziranja zadataka, koje ono podrazumijeva.

U drugom dijelu, usredotočenost je na agilnoj procjeni, točnije o istraživanju kako se procjenjuje trajanje i složenost razvojnih zadatak u agilnom okruženju. Također su predstavljene različite metoda procjene, poput planiranja pokera – a, timske procjene, tehnike usmjerenosti na vrijednost i ostale. Uz to, prikazano je kako se agilna procjena može kombinirati s planiranjem i kako ta integracija doprinosi uspješnom razvoju softvera. Osim toga, na praktičnom zadatku, prikazano je kako to izgleda u stvarnom poduzeću sa stvarnim zahtjevima korisnika.

Ključne riječi: agilni razvoj, planiranje projekata, razvoj programskih proizvoda

Sadržaj

1. Uvod

U današnje vrijeme, svijet razvoja softvera ubrzano se mijenja iz dana u dan, to stvara probleme u razvojnim tvrtkama koje koriste tradicionalne metode zato što se u tim situacijama pokazuju njihove slabosti i ograničenja. Upravo iz tog razloga, kao alternativa i odgovor na ove svakodnevne promjene pojavile su se agilne metode razvoja softvera koje dobivaju sve veću popularnost u industriji. Agilno planiranje i procjena pristupaju problemima tradicionalnog planiranja i procjene softverskog projekta na nov način. Umjesto strogih i detaljnih planova, agilne metode naglašavaju prilagodljivost i kontinuirano usklađivanje s promjenama koje se javljaju tijekom razvojnog procesa (Gulshan & Farooq, 2011.).

Cilj ovog rada je prikazati kako su agilne metode promijenile način na koji se softver razvija te način funkcioniranja pojedinaca kako u timu, tako i pojedinačno.

U teorijskom dijelu ovog rada bit će prikazane prednosti i mane agilnih i tradicionalnih metoda kao i njihove razlike. Biti će obrađene metode poput modela vodopada i V modela razvoja softvera. Nakon toga, detaljnije će biti obrađene najkorištenije agilne metode razvoja poput Scrum-a, Kanban-a, ekstremnog programiranja i kristalne obitelji metoda. Kao sastavni dio agilnog razvoja bit će prikazane i tehnike agilne procjene koja je nezaobilazni dio svakog agilnog projekta.

Poslije obrade metoda razvoja softvera, bit će prikazani alati za agilni razvoj i planiranje te usporedba njih samih kao i stupanj njihove korištenosti u organizacijama.

U praktičnom dijelu rada, bit će prikazana izrada zadatka dobivenog od kolegice Mirjane iz IBM iX poduzeća koji je ogledni primjerak zadatka u stvarnom svijetu. Zadatak podrazumijeva izradu Scrum projekta, tj. rudimentarnog webshop-a, u Jira alatu.

2. Metode i tehnike rada

Prilikom izrade ovoga rada korištene su informacije iz online baza, raznih relevantnih izvora, velik broj dostupnih znanstvenih i stručnih časopisa i članaka kao i knjiga vezanih za agilno planiranje te ostala literatura, poput diplomskih i završnih radova, usko vezana uz temu rada. Korištena je deskriptivna metoda istraživanja koja predstavlja skup znanstveno – istraživačkih postupaka kojima se opisuju pojave. Također se koristila i kvalitativna analiza literature.

Za pisanje ovog završnog rada korišten je program Microsoft Word, također i za izradu svih dijagrama i većine slika. Za uređivanja nekolicine slika koristio sam se alatom GIMP. Za referenciranje korišten je stil bibliografije APA (American Psychological Associtation).

Za izvođenje praktičnog dijela rada korišten je Jira softver kao programski alat za agilno upravljanje projektima. Pomoću Jire izraditi će se projektni plan za odabrani projekt razvoja programskog proizvoda. Za lakše prikupljanje i organizaciju ključnih referenci i literature korišten je alat Zotero. Niti jedan dio teksta nije generiran umjetnom inteligencijom, nego isključivo izvorima navedenima na popisu literature.

3. Tradicionalne metode razvoja softvera

Projektni menadžment je proces planiranja, komuniciranja, organiziranja i raspodijele resursa. Uključuje identificiranje i uravnoteženo korištenje vremena, cijene, kvalitete i resursa projekta. Agilne i tradicionalne metode su dva različita pristupa projektnom menadžmentu (Aslam, Farooq, 2011).

Pojavio se, kao formalna disciplina, u 1950-tima. Razvijen je kako bi unaprijedio dinamiku rada i produktivnost u industriji koja se bavi ekskavacijom prirodnih resursa, tradicionalne tehnike projektnog menadžmenta su najefektivnije u graditeljstvu, ekskavaciji, proizvodnji, itd. gdje se zahtjeva nit sekvencijalnih i repetitivnih postupaka kako bi proizvodnja bila konstantna (Tester, 2023).

Voditelj projekta (eng. Project Manager u daljnjem tekstu PM) nagleda projekt kako kod agilnih metoda tako i kod tradicionalnih. Oni donose odluke kako bi osigurali da projekt ide ispravnim putem i da dođe do zacrtanog cilja. Oba pristupa podrazumijevaju upravljanje rizikom, upravljanje kvalitetom te promjenama (Altvater, 2023.).

Softverske metodike poput metode vodopada, V-modela, RUP-a, spiralnog modela nazivamo tradicionalnim softverskim razvojnim metodologijama. Ove metodologije zahtijevaju definiranje i dokumentiranje stabilnog seta zahtjeva na početku projekta.

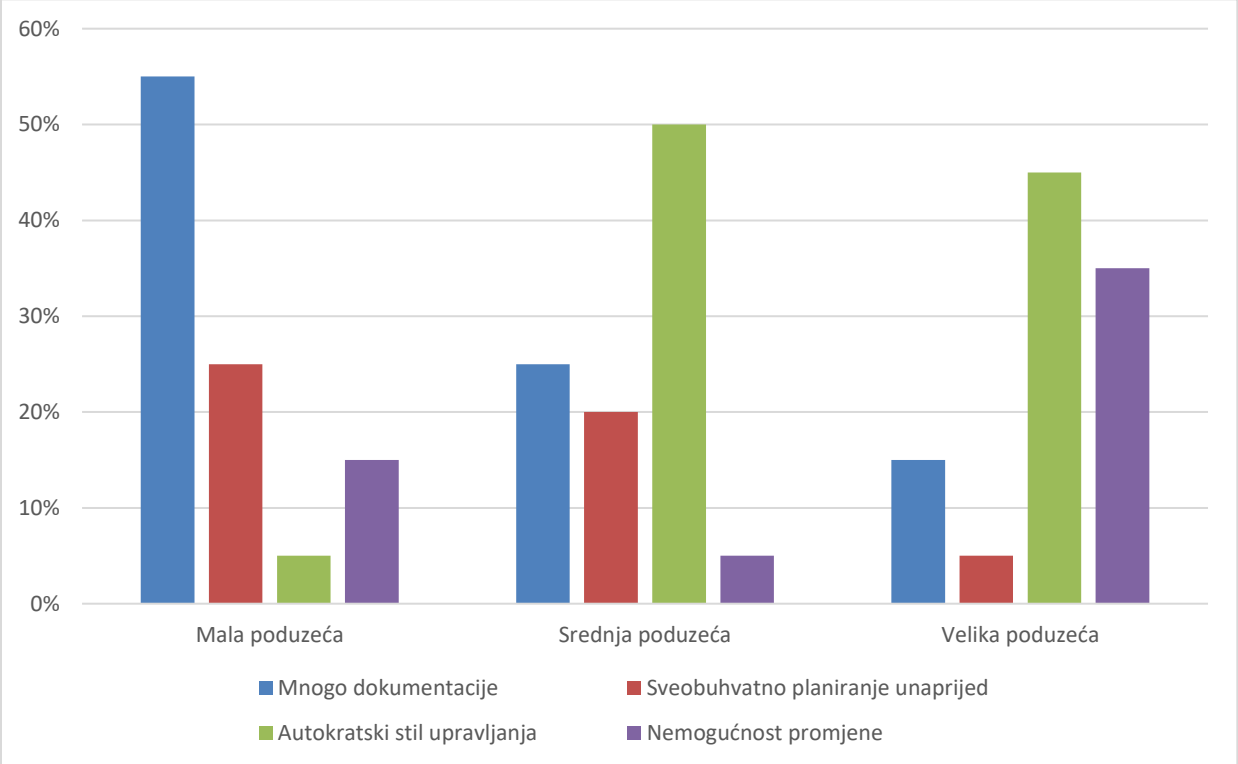
Prilikom definiranja projekta, prvi korak je postavljanje zahtjeva projekta i određivanje trajanja implementacije različitih faza razvoja. Tijekom ovog procesa, potrebno je unaprijed prepoznati moguće probleme koji bi se mogli pojaviti tijekom provedbe projekta. Nakon toga, izrađuju se rani dijagrami i modeli kako bi se prikazali potencijalni problemi s kojima se projekt može suočiti, te se stvara plan djelovanja (eng. roadmap) koja vodi developere kroz njihov rad na projektu. Kada tim odobri plan i arhitekturu, kreće se u fazu programiranja koja traje sve dok se ne postignu određeni ciljevi projekta. Kako se projekt približava kraju, a developeri završavaju svoj dio, pokreće se testiranje proizvoda i ciklus povratnih informacija, koji uključuje i klijenta. Klijent sudjeluje u testiranju proizvoda te pruža povratne informacije. Proizvod se isporučuje nakon što klijent izrazi zadovoljstvo s njime (Yu Beng, Wooi Khong, Wai Yip, Soo Fun, 2012.).

Nažalost, stagnacija tradicionalnih metoda projektnog menadžmenta počela je pojavljivati u polju softverskog dizajna. U 2015. godini softverska statistička tvrtka Standish Group provela je istraživanje vezano za uspješnost i neuspješnost 10 000 projekata u SAD – u. Rezultati istraživanja prema Layton, Ostermiller, Kynaston pokazali su sljedeće:

- Od svih projekata koji su koristili tradicionalne metode, njih 29% završilo je neuspjehom odmah u početnoj fazi. Ti projekti su otkazani prije nego što su uopće bili završeni i nisu rezultirali nikakvim izlaznim produktom.

- Kod 60% projekata koji su koristili tradicionalne metode, primijećeni su izazovi uzrokovani raznim faktorima kao što su očekivana i stvarna cijena projekta, trajanje i kvaliteta. Prosječna razlika između očekivanih i stvarnih rezultata za ove faktore bila je više od 100%.
- Samo 11% projekata je uspjelo ostvariti cilj. Ovaj mali postotak predstavlja projekte koji su bili završeni na vrijeme i unutar budžeta te su uspješno isporučeni.

Na grafu 1. prikazani su rezultati istraživanja (Islam, Fwrworn, 2020.) u kojem je sudjelovalo 21 poduzeće iz područja informacijske tehnologije, telekomunikacija, inženjerstva i ostalih.



Dijagram 1: Nedostaci tradicionalnih metoda u ovisnosti veličine poduzeća (prema Islam, Ferworn, str. 34., 2020.)

U idućim poglavljima detaljno su opisane tradicionalne metode.

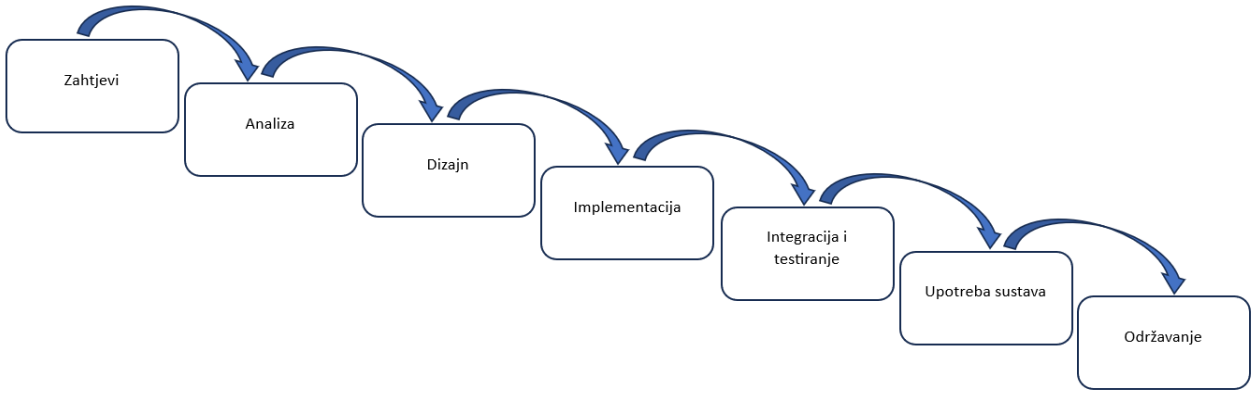
3.1. Vodopadni model razvoja softvera

Od vremena kada je prvo objavljen od Winstona W. Royce – a 1970. godine, model vodopada korišten je i priznat u mnogo veliki tvrtki u polju razvoja softvera (McCormick, 2012.).

Vodopadni model razvoja softvera, također poznat i kao linearni ciklus, je proces koji izgrađen oko planiranog posla i često je korišten kod projekata koji imaju jasno definirane zahtjeve. Jasno zacrtava korake i faze koje moraju biti poduzeti tijekom kreiranja informacijskog sustava. Ove faze često prate specifičan poredak s pregledom posla koji je iza nas na kraju svake faze (Eternal Sunshine Of The Mind, 2013.).

Ključna stvar kod ovog modela koja se smatra izuzetno bitnom jest da svaki korak mora biti završen i ne može se nadograđivati više ili izmjenjivati, zato što kako i samo ime kaže, ovaj model funkcionira kao voda koja teče nizvodno, a vode ne može teći prema gore, tj. Penjati se ili u kontekstu projektnog menadžmenta, nije moguće vraćati se u prijašnju fazu projekta (Aslam, Farooq, 2011.).

Promatrajući sliku 1., možemo vidjeti da model vodopada ima 7 ključnih faza kroz koje svaki



Slika 1: Model vodopada (prema Education-Wiki, 2023)

Na prvom mjestu prema Education-Wiki (2023) nalazi se kreiranje zahtjeva, ovdje je bitno da se zna i jasno razumije što je potrebno dizajnirati. Predstavlja materijale koji se koriste kod izrade proizvoda. Zatim dolazi analiza koja se dijeli na prikupljanje i analizu zahtjeva gdje se podaci prikupljaju od kupca i daju na obradu te na specifikaciju zahtjeva koja predstavlja kreiranje SRS (specifikacija softverskog zahtjeva) dokumenta koji se kasnije koristi za rješavanje nastalih sporova i nesuglasica. U fazi dizajna provjerava se i proučava specifikacija zahtjeva iz prijašnje faze te se u suštini izrađuje čitava arhitektura projekta razvoja softvera. Slijedi implementacija koja uključuje programiranje softvera na temelju specifikacija dizajna koji smo prethodno definirali. Također, u ovoj se fazi svaka od komponenata testira kako bi se provjerila ispravnost njihova rada. Tako prelazimo u sljedeću fazu koja podrazumijeva integraciju i testiranja. Nakon zasebnih testiranja komponenti, okolina za testiranje je na promatranju kako bi se utvrdile neke dodatne greške u dizajnu, kodu ili nepoštivanja protokola. Ključno je u ovoj fazi temeljito ispitivanje i testiranje, a ono se dijeli na:

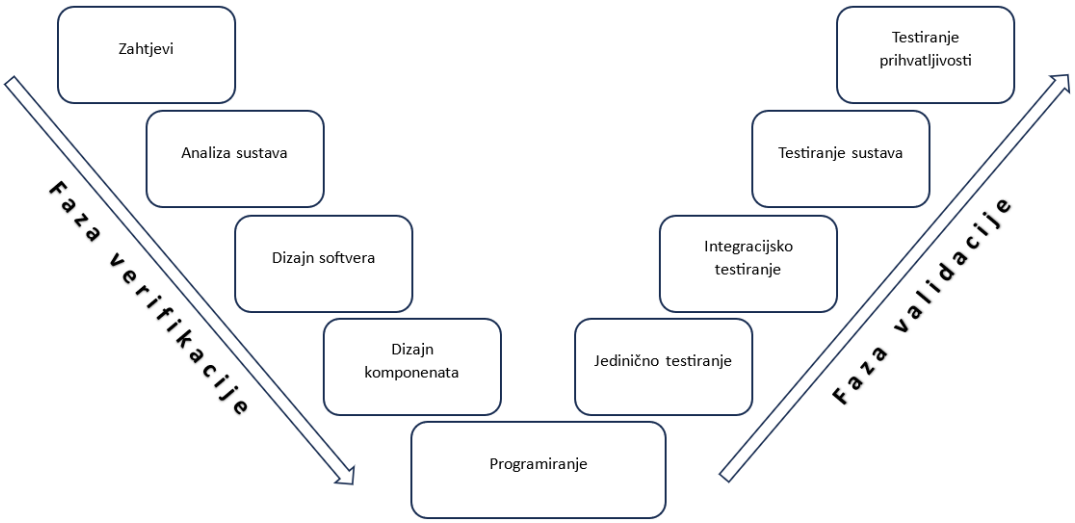
- Alpha testiranje – testiranje razvojnog tima
- Beta testiranje – testiranje kupaca i korisnika

- Provjera prihvatljivosti – izvodi se nakon što smo dobili povratne informacije od kupaca te se donosi finalna odluka o prihvaćanju ili odbijanju softvera.

Kada sva testiranja završe, proizvod koji smo kreirali lansira se na tržište. Ovo uključuje instalaciju, migraciju i podršku čitavog sustava te nas čeka održavanje samog softvera. TE finalno, dolazi faza koja se smatra najvažnijom u modelu vodopada, a to i pokazuju brojke tako što je sav trud koji je uložen u softver u fazama dizajna i razvoja proizvoda samo 60% truda koji je potrebno uložiti u ovoj fazi. Za cilj ima poboljšanje performansi, a podrazumijeva promjene na softveru koje su otkrivene aktivnim korištenjem aplikacije te uključuju sitne greške ili neke zahtjeve postavljene od strane kupaca. Bitno je da se tijekom ove faze korisnicima pruža redovita podrška i pravovremeno održavanje i ispravci softvera. Kao takvo, postoji korektivno održavanje, savršeno i prilagodljivo održavanje (Education-Wiki, 2023).

3.2. V model razvoja softvera

Ovaj model je varijacija modela vodopada koji stavlja veći naglasak na samo testiranje komponenata, ali i softvera u cijelosti. Kao što je vidljivo na slici 2., razvojni proces počinje s lijeve strane, silazno, gdje se definiraju zahtjevi, provodi se analiza sustava, dizajn softvera i komponenata te se u samoj bazi modela kreće s programiranjem prethodno definiranih komponenti sustava. U uzlaznoj putanji modela, započinje testiranja komponenata, integracijsko testiranje kao i testiranje sustava te na kraju samo testiranje prihvatljivosti čiji je proces sličan kao i kod modela vodopada. Kod ovog modela dokumentacija za testiranje se kreira usporedno sa definiranjem zahtjeva i dizajniranjem komponenata te se na taj način pokušava postići visoka kvaliteta testiranja i dobra rezultat na testu efektivnosti (Dennis, Wixom, Roth, str. 53., 2012.).



Slika 2: V-model (prema Oppermann, 2023.)

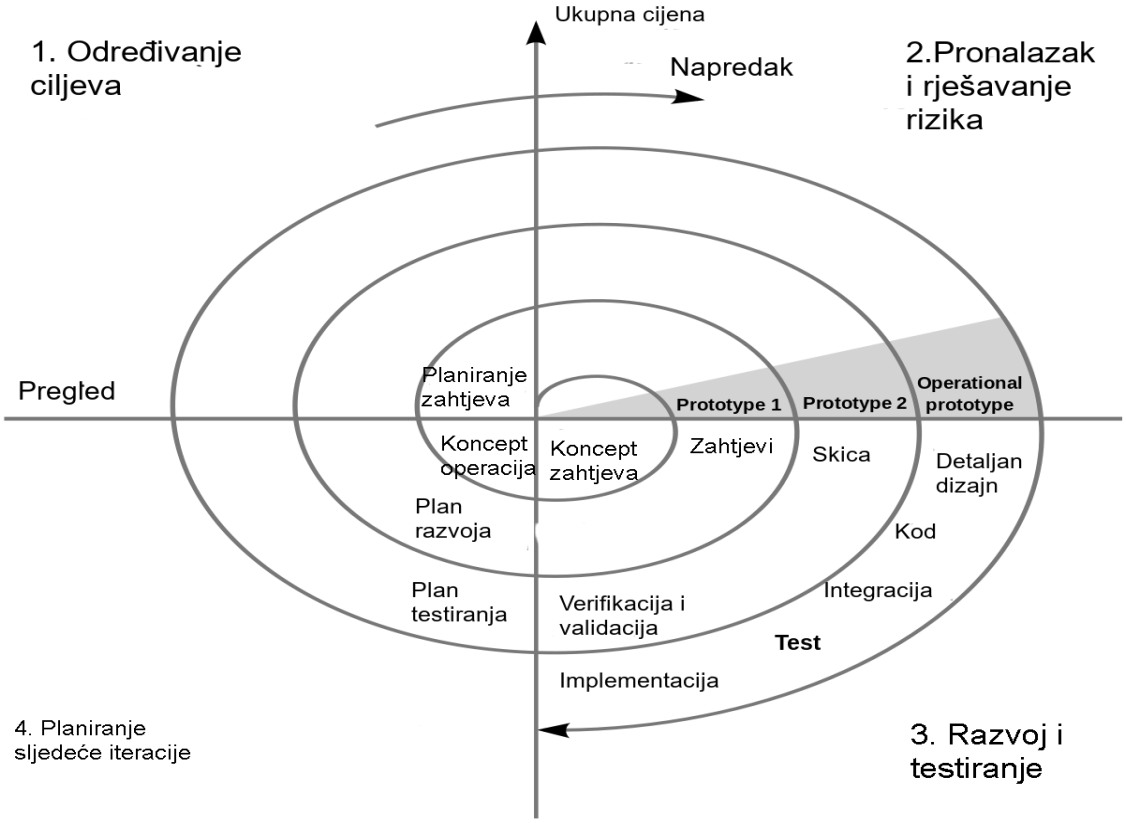
V-model je jednostavan i nastoji pružiti veću kvalitetu proizvoda i sustava time što uvodi ranije testiranje komponenata, samim time testeri imaju uvid u projekt puno ranije nego kod modela vodopada, ali još uvijek pati od krutosti i manjka prilagodljivosti koju je naslijedio od njega. Sukladno tome, ne smatra se veoma prikladnim za korištenje u ubrzanim i dinamičnim radnim okolinama kojih danas ima sve više kao u softverskom razvoju tako i u ostalim područjima poput marketinga, IT razvoja i slično (Dennis, Wixom, Roth, str. 54., 2012.).

3.3. Spiralni model razvoja softvera

Spiralni model baziran je na modelu vodopada i iterativnom modelu, a kao revolucionaran dizajn predstavio ga je javnosti Barry Boehm davne 1986. godine. Kod ovog modela, svaki prototip dodaje neke nove značajke prijašnjima, a testiranja svake faze se provodi nakon što je jedan dio odrađen. One nisu vremenski određene već ovise o vrsti projekta i njegovoj veličini, a započinju ciljevima dizajna te završavaju nakon što je klijent pregledao i odobrio kreirani proizvod (Education-Wiki, 2023).Projekt koji koristi ovaj model, neprestano prolazi kroz iteracije koje se nazivaju spirale odakle i naziv za model. Iznimno je pogodan za dizajniranje sustava koji nisu kruti po svojoj prirodi i otvoreni su za eksperimentiranja. Model se sastoji od 4 faze prema (Eternal Sunshine of The Mind, 2013) i one su vidljive na slici 3:

- Planiranje
- Analiza rizika
- Programiranje
- Evaluacija

Projekt funkcionira na način da prolazi kroz ove 4 faze, slično kao što i model vodopada prolazi kroz svoje. Dobar primjer bi bio Windows operacijski sustav koji nakon što je bio pušten u javnost, prošao je kroz fazu evaluacije i povratnih informacija korisnika te je nakon par godina izašla nova verzija.



Slika 3: Spiralni model (prema Eternal Sunshine of the Mind, 2013)

Glavni nedostatak tradicionalnih metoda koji je presudio i doveo do razvitka agilnih metoda 2001. godine bio je to što je dizajn morao biti u potpunosti definiran prije početka samog programiranja čime su promjene bile ograničene. U slučaju da bi i došlo do promjena bile bi iznimno skupe i neisplative za nastavak projekta. Uz to, prateći prethodno kreiranu dokumentaciju, timovi nisu imali potrebe za pretjeranom komunikacijom, no budući da su ti projekti trajali poduže vremena postojala je mogućnost da će se prvotni zahtjevi klijenata ili kupaca promijeniti što će rezultirati njihovim nezadovoljstvom nakon što će proizvod biti pušten na tržište (Dennis, Wixom, Roth, str. 52., 2012.). Tome su na kraj stalne agilne metode koje su spriječile da se to dešava neprestanom komunikacijom timova i klijenata, ali i međusobnom komunikacijom. Više o agilnim metodama i koje postoje nalazi se u poglavljima koja slijede.

4. Agilne metode razvoja softvera

Agilni menadžment predstavlja pristup projektnom menadžmentu koji stavlja naglasak na rano isporučivanje poslovne vrijednosti, kontinuirani napredak proizvoda i procesa tijekom projekta, te angažiranost razvojnog tima. Osnovna svrha agilnog menadžmenta je osigurati isporuku dobro testiranih proizvoda koji su usklađeni s potrebama klijenta (Layton, Ostermiller, Kynaston, 2017).

Agilnost je sposobnost stvaranja promjena i reagiranja na iste. To je zapravo način nošenja s neizvjesnim i turbulentnim okruženjem te u konačnici i uspjeha u tome (Agile Alliance, 2023)

2001. godine, okupila se skupina softverskih i projektnih stručnjaka kako bi raspravljali o zajedničkim uspješnim projektima. Ova skupina nedugo zatim je stvorila dokument nazvat „Agilni manifest“, koji je sadržavao izjavu o vrijednostima uspješnog softverskog razvoja kako slijedi:

„Otkrivamo bolje načine razvoja softvera i pomaganju drugima da učine isto. Uz zajedničke napore došli smo do sljedećih vrijednosti:

Pojedinci i interakcije prije procesa i alata

Ispravan softver prije sveobuhvatne dokumentacije

Suradnja kupaca prije pregovora o ugovoru

Odgovor na promjenu prije držanja plana“

Ti isti stručnjaci također su osmislili "Principles behind the Agile Manifesto", skup od 12 načela koja podržavaju vrijednosti agilnog manifesta. Agilan pristup u kontekstu razvoja proizvoda fokusira se na ljude, komunikaciju, proizvod i fleksibilnost kao ključne elemente projektnog upravljanja. Unatoč različitim agilnim metodikama (npr. Crystal), okvirima (npr. Scrum), tehnikama (npr. Zahtjevi za korisničke priče) i alatima (npr. Relativno procjenjivanje), svi dijele zajedničku karakteristiku – pridržavanje pravila agilnog manifesta i 12 agilnih načela (Layton, Ostermiller, Kynaston, 2017)

4.1. Osnovni principi

Agilni pristupi temelje se na empirijskoj metodi kontrole – procesu donošenja odluka na temelju stvarnosti koje se promatraju u projektu. U kontekstu metodologija razvoja softvera, empirijski pristup može biti učinkovit, kako u razvoju novih proizvoda, tako i u projektima poboljšanja i nadogradnje. Učestalim provjerama odrađenog posla, iz prve ruke je moguće napraviti poboljšanja ili ispravke projekta, ako je to potrebno.

Kako bi se omogućile česte provjere odrađenog posla i mogućnost donošenja promjena u posljednjem trenutku, agilni projekti rade u iteracijama tj. Manjim segmentima cjelokupnog projekta. Agilni projekt uključuje istu vrstu rada kao i u tradicionalnom projektu: kreiraju se zahtjevi i dizajni, razvija se proizvod, dokumentira se, ako ima potrebe za tim, integrira se s drugim proizvodima. Zatim slijedi testiranje proizvoda, popravljanje problema te ga se stavlja u upotrebu. Međutim, umjesto da istovremeno dovršite ove značajke proizvoda, kao što je slučaj s tradicionalnim projektom, upotrebom agilnih metoda, projekt se dijeli na iteracije koje se nazivaju „*sprints“* (Layton, Ostermiller, Kynaston, 2017)

Principi na koje se mogu podijeliti agilne metodike su prema (Kušek, 2010) sljedeći:

- Zadovoljstvo kupaca
- Promjena zahtjeva
- Kratak period isporučivanja
- Suradnja tima i naručitelja
- Komunikacija licem u lice
- Osnovna mjera za praćenje napretka je funkcionalni softver
- Usmjerenost na tehničku izvrsnost i dobar dizajn
- Jednostavnost
- Periodičko ispitivanje dobrih i loših odluka

Agilne metode razvoja na prvo mjesto stavljaju poslovni povrat ulaganja (ROI) te su dobro poznate po naglasku na komunikaciju i uključenost kupaca. Za svaku isporučenu iteraciju, razvojni tim i kupci će održati sastanak na kojem će članovi tima komunicirati o promjenama koje su donesene i sažeti svoj rad obavljen u završenoj iteraciji. Nakon što razvojni tim iznese novosti na projektu, kupci će pružiti povratne informacije o isporučenom softveru kako bi usavršili trenutne značajke ili uključili dodatne značajke u sustav (Beng, Woo Khong, Wai Yip, Soo Fun, 2012)

S obzirom na činjenicu da se zahtjevi kupaca prihvaćaju iterativno kao kontekst, agilni razvoj omogućuje isporuku krajnjeg proizvoda koji bolje zadovoljava potrebe kupaca. Kroz svaki kratki ciklus iteracije, završeni moduli se pružaju kupcima radi pregleda. Ovi moduli nisu integrirani kao cjelovit sustav, što znači da bilo kakve promjene ili dodatne značajke ne bi značajno povećale troškove razvoja. Koristeći ovu fleksibilnost, programeri su uvijek spremni uključiti sve značajke koje korisnici žele, a integracija sustava će se dogoditi samo kad korisnici nemaju dodatnih zahtjeva. Stoga, ovaj pristup ima potencijal u velikoj mjeri zadovoljiti kupce pružanjem cjelovitog sustava koji sadrži sve željene funkcije (Beng, Woo Khong, Wai Yip, Soo Fun, 2012). U nastavku će biti objašnjene prednosti i nedostaci agilnih metoda te prikazane grafički u odnosu na veličinu organizacije.

4.2. Prednosti i nedostaci agilnog pristupa

Prednosti agilne metodologije su direktno povezane s bržim, lakšim i angažiranim načinom razmišljanja. Proces, ukratko, isporučuje ono što kupac želi, kada to želi. Mnogo je manje „praznog hoda“ i izgubljenog vremena utrošenog za razvoj u pogrešnom smjeru, a cijeli sustav brže reagira na promjene.

Najvažnije prednosti prema kojima agilni pristup doprinosi projektu su (Altvater, 2023):

- Brzina – Jedna od najznačajnijih prednosti agilne metodologije je njezina brzina. Skraćeni životni ciklus razvoja softvera rezultira manje vremena između faze trošenja resursa i faze generiranja prihoda. To se reflektira na povećanu profitabilnost poslovanja.
- Povećano zadovoljstvo kupaca – uz agilni pristup, kupci ne čekaju mjesecima ili godinama kako bi dobili finalni proizvod koji možda ne zadovoljava njihove stvarne potrebe. Umjesto toga, dobivaju iteracije nečega što je vrlo blisko onome što zapravo žele, i brzo. Sustav se brzo prilagođava kako bi se poboljšalo rješenje prema potrebama kupca, uzimajući u obzir promjene u okruženju ili zahtjevima. Time se osigurava da kupci dobiju rješenje koje točno odgovara njihovim potrebama u najkraćem mogućem roku.
- Cijeni zaposlenike – zaposlenici čije su ideje cijenjene, daleko su produktivniji od onih kojima se naređuje da slijede skup pravila. Agilna metodologija poštuje zaposlenike tako što im daje cilj, a zatim vjeruje da će ga lako postići. Budući da su oni ti koji imaju ruke na kontrolama i oni koji vide prepreke koje se svakodnevno pojavljuju, zaposlenici su u najboljoj poziciji da odgovore na izazove i ispune ciljeve koji su pred njima.
- Eliminira dupli posao – uključivanjem kupca u više faza od samo faza zahtjeva i isporuke, projekt ostaje usklađen s potrebama kupca na svakom koraku. To znači manje povlačenja unazad i manje potraćenog vremena između vremena kada obavljamo posao i vremena kada kupac predlaže izmjene.

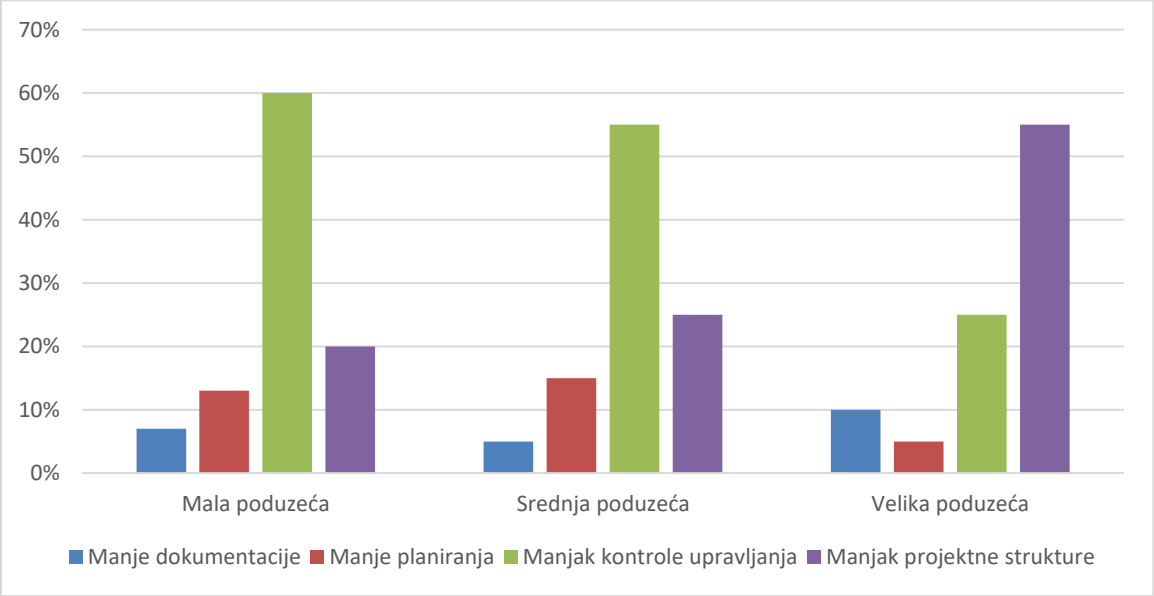
Unatoč mnogim prednostima agilnih metoda u odnosu na tradicionalne, njihova primjena može izazvati neke poteškoće. Jedna od tih poteškoća je značajno smanjenje količine dokumentacije u agilnom pristupu, gdje se naglašava da sam kod treba služiti kao dokumentacija. To može potaknuti programere koji koriste agilne metode da stavljaju više komentara u kod kao način objašnjavanja. No, ova praksa može predstavljati izazov za početnike ili nove članove tima, jer mogu imati teškoća u razumijevanju projekta ili koda na kojem rade. Nedostatak dokumentacije može dovesti do toga da moraju često postavljati pitanja iskusnijim programerima kako bi razjasnili stvari. To može rezultirati kašnjenjem u

isporuci iteracija projekta, što nadalje može neželjeno povećati troškove razvoja (McCormick, 2012).

Jedan od načina da se prevlada ovaj problem jest traženje ravnoteže između smanjenja dokumentacije i osiguranja dovoljno informacija za nove članove tima. Moguće rješenje je da se kod nadopuni komentarima samo tamo gdje je to stvarno potrebno kako bi se pojasnila složenija logika ili rješavanje određenih problema. Također, timovi mogu redovito provoditi sastanke ili radionice s novim članovima kako bi im pružili bolje razumijevanje projekta i koda (Milošević, 2016).

Izazova ima mnogo, a onaj koji potencijalno snažno može narušiti percepciju superiornosti agilnih metoda je činjenica da programeri često moraju sudjelovati u redovitim sastancima na tjednoj bazi. Ti sastanci, koji uključuju prezentacije svojih modula drugim članovima tima i klijentima, često mogu biti dosadni i iscrpljujući. Razlog za to je što će vjerojatno biti potrebne promjene u modulima zbog čestih promjena zahtjeva tijekom iteracija projekta. Ovakva dinamika može dovesti do gubitka vremena i energije, umjesto fokusiranja na stvarni razvoj softvera. Programeri se mogu osjećati frustrirano zbog čestih prezentacija i promjena u radu, što može smanjiti njihovu produktivnost i motivaciju (Beng, Woo Khong, Wai Yip, Soo Fun, 2012).

Na grafu 2. prikazani su rezultati istraživanja (Islam, Fwrworn, 2020) u kojem je sudjelovalo 21 poduzeće iz područja informacijske tehnologije, telekomunikacija, inženjerstva i ostalih.

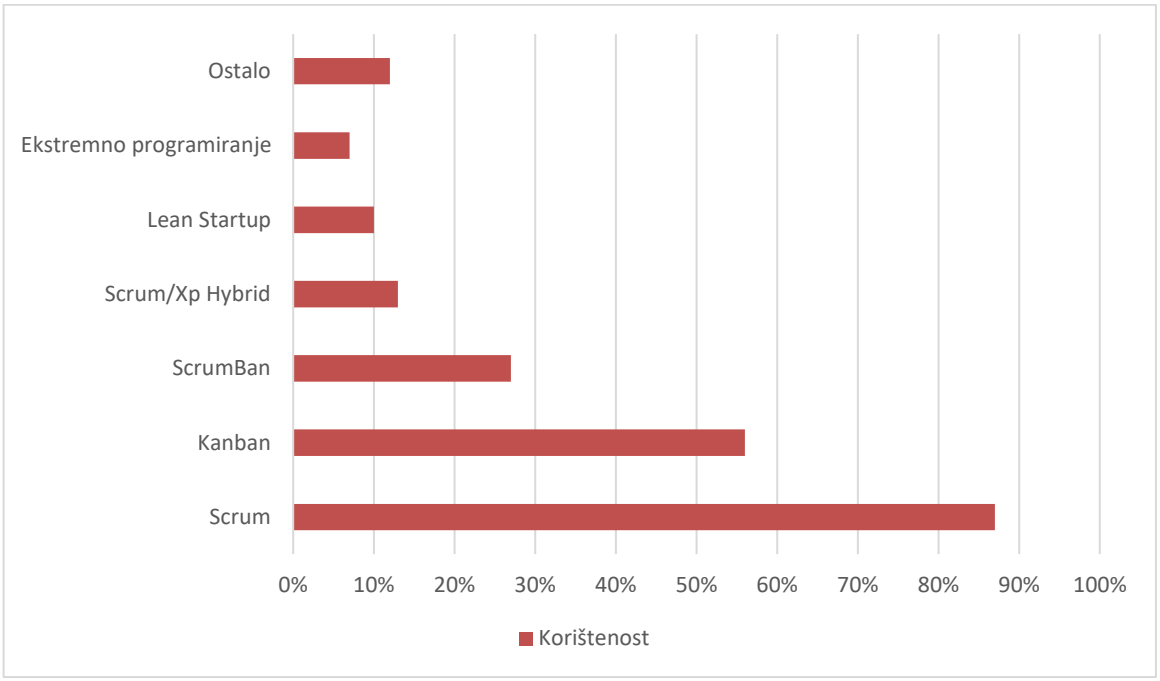


Dijagram 2: Nedostaci agilnih metoda u ovisnosti veličine poduzeća (prema Islam, Ferworn, str. 34., 2020)

Međutim, važno je napomenuti da su redoviti sastanci i komunikacija s drugim članovima tima i klijentima ključni elementi agilnih metoda. Oni omogućuju razmjenu

informacija, razumijevanje promjena u zahtjevima i otkrivanje problema na vrijeme. Timovi mogu pokušati smanjiti teret sastanaka koristeći tehnike poput "daily stand-up" sastanaka koji su kratki, usmjereni na ključne informacije i održavaju se svakodnevno. Iako mogu postojati izazovi u vezi s redovitim sastancima, prednosti agilnih metoda uključuju brzu prilagodbu promjenama, fleksibilnost i mogućnost rada na prioritetima kako bi se ostvarili ciljevi projekta. Važno je da timovi pronađu ravnotežu između potrebnih sastanaka i stvarnog rada kako bi ostvarili uspješan razvoj softvera (Beng, Woo Khong, Wai Yip, Soo Fun, 2012).

Neke od njih su ranije navedene, no na dijagramu 3. prikazane su najkorištenije i najpopularnije metode, a to su: Scrum, Kanban, Scrumban, Scrum/XP Hybrid, Lean Startup, Ekstremno programiranje.



Dijagram 3: Usporedba agilnih metoda (prema State of Agile, 2022)

U idućem poglavlju, slijedi detaljna usporedba agilnih i tradicionalnih metoda na različitim područjima djelovanja.

5. Usporedba agilnih i tradicionalnih metoda

Jedna od glavnih razlika između agilnog razvoja i konvencionalnih razvojnih metoda je sposobnost agilnih metoda da brzo i ekonomično isporuče rezultate čak i na složenim projektima s nejasno definiranim zahtjevima. Agilne metode ističu važnost timskog rada, suradnje s klijentima i prilagodljivosti na promjene, dok konvencionalne metode više naglašavaju formalne ugovore, planiranje, striktno definirane procese, dokumentaciju i korištenje alata (Yu Beng, Wooi Khong, Wai Yip, Soo Fun, 2012).

Agilni menadžment se temelji na ponavljajućem i fleksibilnom pristupu s fokusom na klijentsko zadovoljstvo i funkcionalan softver. Nasuprot tome, tradicionalne metode su čvršće i manje podložne promjenama nakon što se zahtjevi jednom definiraju. Svaka metoda ima svoje prednosti i nedostatke. Konačan izbor između njih ovisi o specifičnim potrebama i zahtjevima projekta, timskom iskustvu te raspoloživim resursima. U nekim slučajevima, projektni timovi odluče primijeniti hibridni pristup koji kombinira elemente oba pristupa kako bi postigli ravnotežu između fleksibilnosti i predvidljivosti (Narasimman, 2023).

U tablici 1. detaljnije je prikazana usporedba agilnih i tradicionalnih metoda na temelju niza parametara.

Parametar	**Agilne metode**	**Tradicionalne metode**
Težina uvođenja promjena	Laka	Teška
Pristup razvoju	Prilagodljiv	Predvidljiv
Razvoj	Orijentiran na kupca	Orijentiran na proces
Veličina projekta	Mali ili srednji	Velik
Stil upravljanja	Vodstvo i suradnja	Naređivanje i kontrola
Dokumentacija	Štura	Opširna
Tip organizacije	Mali ili srednji prihodi	Visoki prihodi
Broj zaposlenika	Manji broj ljudi	Veći broj ljudi
Budžet	Mali	Veliki
Broj timova	Jedan tim	Više timova
Veličina tima	Mala	Srednja ili velika

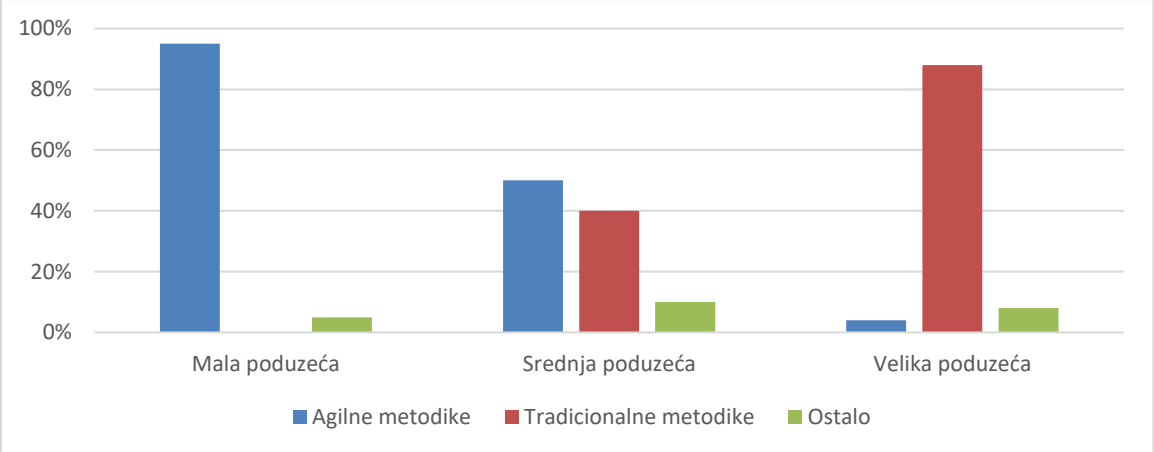
Tablica 1: Usporedba agilnih i tradicionalnih metoda (prema Shaikh, Khan, Farah Siddique i Quershi, 2021)

Učinkovitost različitih metoda određuje se prema kvaliteti konačnog proizvoda, ukupnom broju grešaka u proizvodu te vremenu potrebnom za njegovu izradu. Prema istraživanju, agilni pristupi su se pokazali učinkovitijima od modela vodopada, jer su sposobni

prilagoditi se zahtjevima stvarnog svijeta. Za razliku od tradicionalnih metoda, agilni pristupi su fleksibilniji i mogu se lakše nositi s promjenama koje se pojavljuju tijekom projekta, što rezultira kraćim vremenom rada (McCormick, 2012).

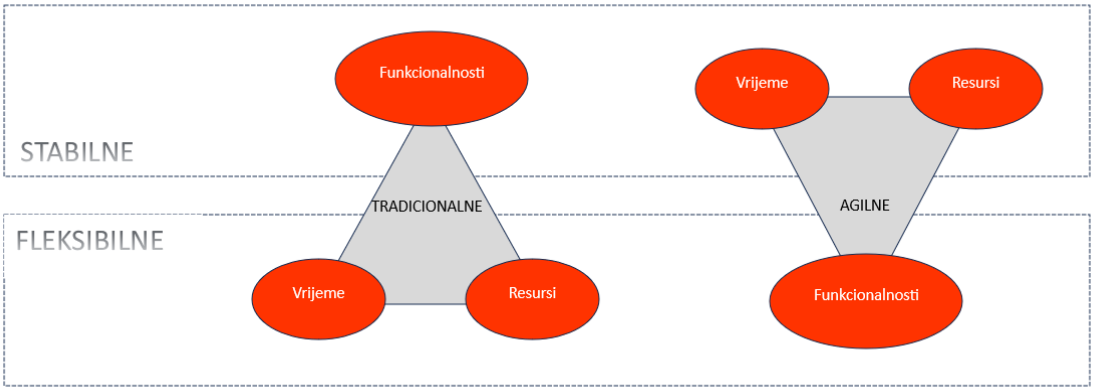
Agilne metode također potiču timski rad, interakciju i brze iteracije kako bi se postigla bolja transparentnost i suradnja unutar razvojnog tima. To je dovelo do promjene u „mindset"-u te samim time proizvelo veću uspješnost projekata, smanjenje rizika i veće zadovoljstvo korisnika (Sustirnere, 2018). Tradicionalne vodopadne metode razvoja softvera karakterizirane su dugim i detaljnim planiranjem unaprijed, vrlo često propadnu zbog promjenjivosti zahtjeva, tehnološkog napretka kao i dinamike tržišta (Eternal Sunshine Of The Mind, 2013). Agilne se metode s druge strane više okreću iterativnom i inkrementalnom pristupu s kraćim ciklusima planiranja, tj. „sprintova". Poznavajući osnove agilnog planiranja i procjene, razvojni timovi mogu povećati svoju učinkovitost, optimizirati iskoristivost resursa te uz pomoć neprestane komunikacije s klijentom postići željeni rezultat. Nažalost, u trenucima kada klijenti nemaju vremena ili ne žele utrošiti svoje vrijeme s ciljem poboljšanja softvera, to može biti teško izvedivo te potencijalno može dodatno oduljiti vrijeme razvoja ili ga prekinuti (Markovinović, str. 4., 2018).

Međutim, valja napomenuti da niti agilni pristupi nisu bez mana. Ipak, agilni pristupi imaju širu primjenu u usporedbi s tradicionalnim kada su u pitanju manja ili srednja poduzeća, ali stvari se uvelike mijenjaju kada se radi o velikim organizacijama i to je vidljivo na dijagramu 4. No, bez obzira na odabranu metodu, ključni čimbenik uspješnosti projekta leži u sposobnostima i vještinama tima koji na njemu radi, njihovoj sposobnosti međusobne komunikacije te njihovoj posvećenosti cilju projekta (McCormick, 2012).



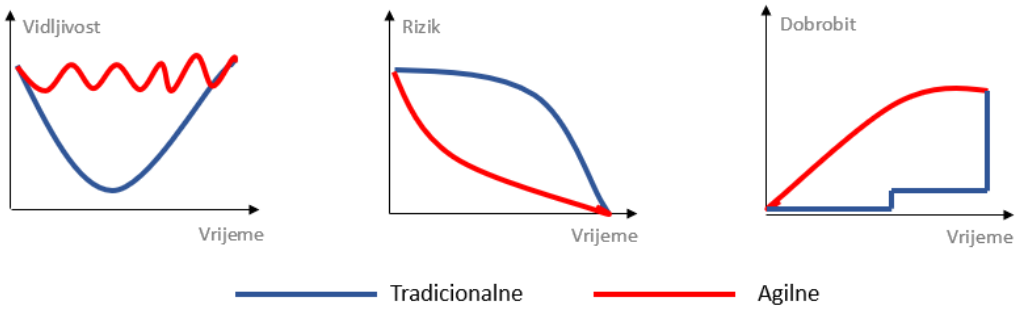
Dijagram 4: Odabir metode ovisno o veličini poduzeća (prema Islam, Ferworn, str. 35., 2020.)

Kao što smo već rekli kod opisivanja načina rada tradicionalnih metoda, softverske funkcionalnosti se definiraju u početnim fazama projekta te se vrijeme i resursi koji su potrebni za izradu, testiranja i provođenje tih komponenti u svijet korisnika, mijenjaju ovisno o njihovoj zahtjevnosti, mogli bi smo reći da su te promjenjive, a funkcionalnosti da su fiksne varijable. Kod agilnih metoda imamo suprotan slučaj, vrijeme i resursi su fiksni, primjerice unutar jednog sprinta od 2 tjedna, a izradimo onoliko funkcionalnosti koliko stignemo u tom vremenskom periodu te ih to čini promjenjivom varijablom u karikaturi prikazanoj na slici 4.



Slika 4: Različitost pristupa tradicionalnih i agilnih metodika (prema Islam, Ferworn, 2020.)

Agilne i tradicionalne metode uvelike se razlikuju, kao što je već spomenuto, kada u pitanje dolazi vidljivost projekta, rizik kojem se potrebno izložiti te dobrobitima koji nude kroz vrijeme izrade projekta. Na slici 5 možemo vidjeti grafičku usporedbu na temelju tih stavki.



Slika 5: Usporedba tradicionalnih i agilnih metodika (prema Strasser, 2022)

U nastavku rada obradit ću najpopularnije i najkorištenije metode spomenute na dijagramu 3.

6. Popularne metodike razvoja softvera

Prilikom stvaranja nekog softvera razvojni inženjeri moraju se koristiti agilni metodama kako bi lakše stvorili vrijednost. Prije početka samog razvoja, pred njima je bitna odluka odabira agilne metode kojom će tu vrijednost postići. U poglavljima koja slijede, opisane su najpopularnije metodike razvoja softvera današnjice.

6.1. Scrum

Kada govorimo o agilnim metodama, daleko najpopularniji je **Scrum** za što i postoji dobar razlog. Jednostavan je okvir (eng. *Framework*) koji pomaže ljudima, timovima i organizacijama da stvore vrijednosti kroz prilagodljiva rješenja. Funkcionira na principu teorije empirizma koja tvrdi da znanje proizlazi iz iskustva i da se odluke i prilagodbe donose na temelju opažanja (Agile Alliance, 2023).

Scrum podupire nesigurnost, kreativnost i rizik. Postavlja strukturu oko procesa učenja, omogućujući timovima da procjene ono što su stvorili, kako su to stvorili. Upija način na koji timovi rade te im daje alate za samoorganizaciju i efikasno poboljšanje brzine i kvalitete rada (Sutherland, 2014).

Kod izrade projekta koristeći Scrum tehnologiju, susrećemo se s raznim novim terminima. Definirane su specifične uloge ljudi na projektu, artefakti i događaji. Sukladno tome, imamo tri bitne uloge koje se provlače kroz cijeli projekt, a to su:

- Vlasnik proizvoda (eng. product owner) – to je osoba koja predstavlja tvrtku za koju se razvija softver te prenosi poslovne zahtjeve razvojnom timu (Layton, Ostermiller, & Kynaston, 2017).
- Scrum tim – imaju glavnu zadaću kod stvaranja softvera. Posvećeni su projektu, a svaki član tima ima set vještina koje ih osposobljavaju za to da mogu raditi više različitih poslova na projektu (Layton, Ostermiller, & Kynaston, 2017).
- Scrum master – najvažnija uloga Scrum mastera je da zaštiti tim od distrakcija, ukloni potencijalne nejasnoće te da neprestano unaprjeđuje radnu atmosferu (Layton, Ostermiller, & Kynaston, 2017). Ova uloga nema nužno neki autoritet, no često ljudi koji je utjelovljuju imaju zadaću voditi tim zbog tako utjecajne pozicije (Scrum Alliance, 2023).

Jedna od najbitnijih značajki Scruma, žila kucavica svakog projekta kreiranog ovom tehnologijom, jesu kratke iteracije (eng. sprint). Sprint je vremenski okvir od mjesec dana ili manje tijekom kojeg tim proizvodi inkremente finalnog proizvoda (Scrum Alliance, 2023). Novi

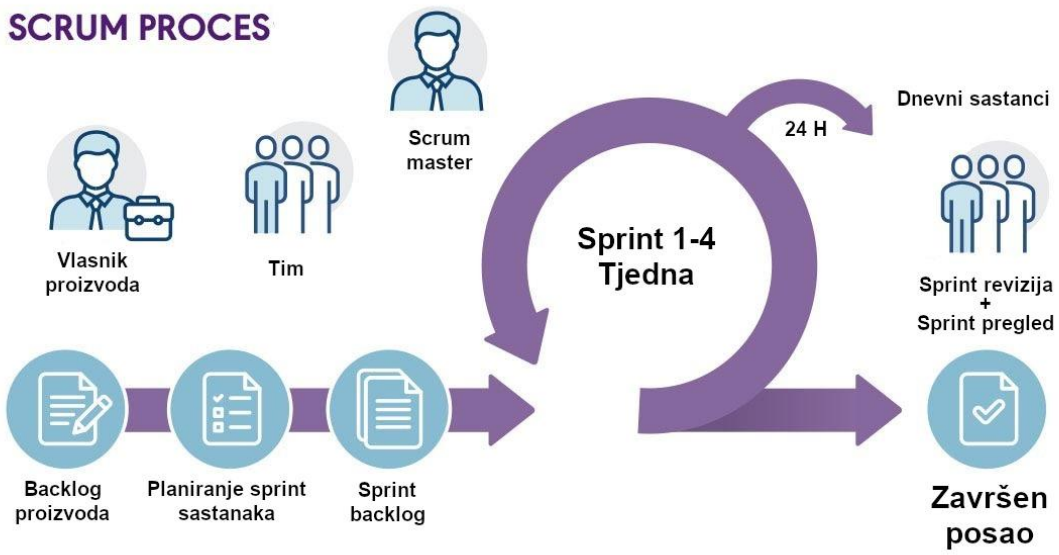
sprint slijedi odmah nakon što je prijašnji završio, a projekti kreirani Scrum tehnologijom mogu trajati svega dva do tri sprinta na manjim i 10-20 na većim projektima (Stellman, 2017)

Kao što svaka kuća mora imati temelje na kojima je izgrađena, nema iznimaka niti kod nastanka projekata temeljenih na Scrumu. Noseći stupovi svakog Scrum projekta su transparentnost, inspekcija i prilagodba. Pod pojmom transparentnost misli se na otvorenu i opuštenu komunikaciju, bez skrivenih loših vijesti, kako između radnih kolega i klijenta, tako i između onih na višim pozicijama. Uz to, omogućen je neprestani uvid u radno stanje i napredak projekta svim njegovim članovima, sve to za uspješan dolazak do zajednički postavljenih ciljeva. U procesu inspekcije, koja bi se trebala provoditi tijekom svakog Scrum događaja, sudjeluje cijeli tim u bliskoj suradnji sa Scrum Masterom koji ih treba podržavati u toku cijelog trajanja procesa. Isto vrijedi i za povratne informacije kupca koji na transparentan način ima uvid u proizvod na kraju sprinta. Na temelju dobivenih rezultata inspekcije Sprinta, provodi se prilagodba na donesene promjene, novu situaciju i okolnosti. U suštini, do izražaja bi trebao doći glavni razlog implementacije agilnih metoda bilo da je to želja za većim zadovoljstvom kupaca ili zaposlenika, smanjenje troškova vlasništva, brže vrijeme za reklamiranje svog proizvoda ili veći povrat ulaganja (Radhakrishan, 2023)

Kako je već spomenuto, uz specifične uloge kao dio Scrum terminologije postoje i artefakti i događaji. Artefakti su opipljivi dijelovi projekta, a to su:

- Popis funkcionalnosti proizvoda (eng. backlog proizvoda) – jest stalno promjenjiv popis novih značajki, poboljšanja, ispravka grešaka, zadataka ili radnih zahtjeva potrebnih za razvitak proizvoda. Ovaj artefakt se često ažurira, a za to je zadužen vlasnik proizvoda koji osim toga pomaže developerima razumjeti i odabrati komponente koje će najprije razvijati, a nastavno na to oni donose procjenu veličine posla te okvirno trajanje (Agille Alliance, 2023).
- Popis funkcionalnosti sprinta (eng. sprint backlog) – predstavlja popis svih zahtjeva koje je razvojni tim odabrao da će odraditi u određenom sprintu, a uz njega posao potreban za kreiranje inkrementa te plan za isporuku istog. Ovaj popis je zapravo plan koji su developeri napravili za developere. Kako sprint napreduje tako se i popis funkcionalnosti sprinta ažurira zato što je više informacija dostupno za manevriranje. Timu je u cilju imati dovoljno informacija za lakše prezentiranje obavljenog posla na dnevnom Scrum sastanku (eng. daily scrum) (Agile Alliance, 2023).
- Inkrement – nakon završetka sprinta nastaju djelići projekta koje nazivamo inkrementima. Naravno, tijekom jednog sprinta moguće je stvoriti više inkremenata, na koje se nadodaju svi prijašnji stvoreni inkrementi, i svaki od njih predstavlja dio funkcionalnog dijela softvera koji je razvojni tim dužan predstaviti korisnicima prilikom pregleda sprinta (eng. sprint review) (Stellman, 2017).

Svaki sprint (slika 6) započinje planiranjem sprinta (eng. sprint planning), tijekom tog događaja, definira se rad na projektu koji je potrebno odraditi tijekom sprinta. U njegovoj izradi sudjeluje cijeli Scrum tim, a rezultat je kao što i sam naziv događaja govori, plan sprinta. Nakon toga kreće se s provedbom tog plana, a napredak sprinta provjerava se izvođenjem dnevnih Scrum sastanaka. Osim toga, u tom 15-minutnom događaju, po potrebi se prilagođava planirani rad. Sljedeći bitan događaj koji se izvodi tijekom sprinta je pregled sprinta (eng. Sprint Review) u kojem se provjerava ishod sprinta, razmatraju se buduće prilagodbe te Scrum tim prezentira posao koji je do tog trenutka odrađen o raspravljaju o mogućim napredcima. Zadnji događaj iteracije je retrospektive sprinta (eng. sprint retrospective) u kojem se raspravlja o tome što se dogodilo tijekom Sprinta te o načinima napretka u smislu povećanja kvalitete i učinkovitosti (Stellman, 2017). Nakon što su svi događaji obavljeni dolazimo do finalnog događaja koji zaključuje cijeli projekta, a to je lansiranje proizvoda (eng. product release). On se može pojaviti na kraju sprinta ili nakon niza sprintova ovisno o spremnosti i dovršenosti softvera kojeg se razvija zajedno sa svim aktivnostima koje ga podupiru, primjerice prodaja, potpora, marketing i ostale (Aha! Labs Ins, 2023)



Slika 6: Slikovni prikaz Scrum procesa (prema pm-partners 2021)

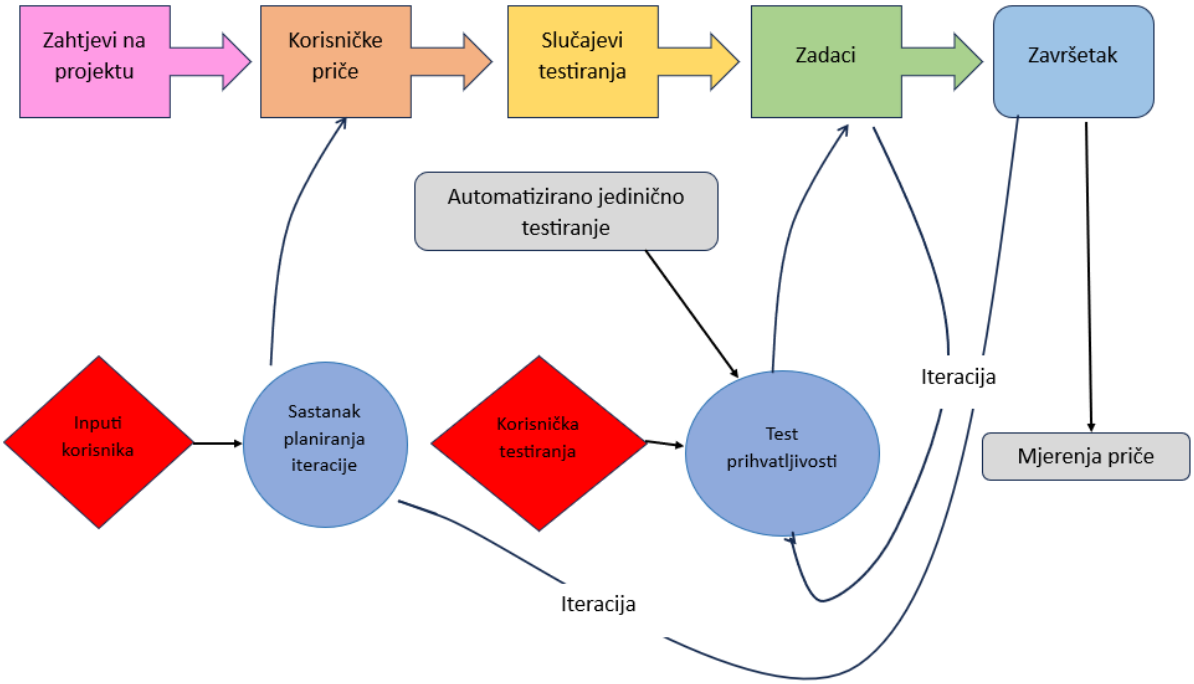
6.2. Ekstremno programiranje

„Ekstremno programiranje intenzivna je, disciplinirana i okretna metodologija razvoja softvera usredotočena na kodiranje unutar svake faze životnog ciklusa razvoja softvera“ (Technopedia, 2023)

Ekstremno programiranje (u daljnjem tekstu XP), budući da je također jedna od agilnih metoda razvoja, se izvodi u kratkim iteracijama gdje prva faza, iteracija, traje svega jedan dan ili jedan tjedan, dakle ekstremno kratko. Svrha ove iteracija svodi se na to da se ugrubo stvori dio koda koji je nepotpun te može, ali ne mora biti funkcionalan (Frank, 2021) Koristeći se riječima WIlfrida Hutagalunga (2006), *„At the end of the day, there has to be a program"* ili na hrvatskom: „Na kraju dana, mora postojati program".

Glavni fokus ove metode stavlja se na to da je cijeli softver razvijen od strane dvoje ljudi koje sjedi za istim računalom. Ovakvim načinom programiranja, jedna osoba programira, stvara kod, a druga osoba fizički stoji kraj te osobe i nadgleda, provjerava svaku liniju koda iste sekunde kada je ona unesena te pomaže kada je potrebno. Moglo bi se reći da je osoba koja kodira, vozač, a osoba koja nadgleda, promatrač ili navigator. Timovi koji su implementirali ovaj način programiranja u svoju rutina rada, dokazano imaju poboljšanu kvalitetu softvera, ali i kraće vrijeme izrade zato što mogu brže riješiti probleme zajedno i ne dolazi do nepotrebnog zabušavanja zbog nemogućnosti dolaska do rješenja samostalno (Agile Alliance, 2023).

Osim ove praktične tehnike postoje i još neke. Primjerice metafora kod koje programeri dobivaju pojednostavljenu sliku sustava u obliku metafore pomoću koje im je lakše shvatiti ideju iza komplicirane sheme te imaju bolju sliku željenog sustava. Planiranje verzija je isto jedna od tehnika gdje se korisničke priče osobe koja je naručila proizvod pretvaraju u plan izvođenja. Takvi planovi su često samo ugrubo doneseni i velike su šanse da će se kasnije dorađivati. Od velike je važnosti kod ove tehnike da se svi drže svojih obaveza. Također, možemo spomenuti i tehniku objavljivanja malih i čestih izdanja. Na taj način omogućujemo korisnicima da prije imaju uvid u proizvod koji se za njih razvija te nam unaprijed mogu dati komentare i otvoriti mogućnosti za bolja unaprjeđenja softvera (Kušek, 2010). Na slici 7 vidljiv je slikovit prikaz funkcioniranja XP procesa.



Slika 7: Proces XP-a (prema Hutagalung, 2006)

Prema konceptualnom istraživanju Sharme, Sarkar i Gupte (2012) XP je jedna od najuspješnijih metoda za razvoj agilnog softvera zato što za fokus ima samo zadovoljstvo klijenata. Upravo iz tog razloga su iteracije kod XP-a gotovo duplo kraće nego kod SCRUM razvoja. XP zahtjeva maksimalnu interakciju i sudjelovanje klijenata tijekom razvoja softvera. Isto tako, za razliku od SCRUM-a, tijekom iteracija XP-a klijenti su itekako dobrodošli da predlože promjene u bilo kojem stadiju razvojnog ciklusa (Sharma, Sarkar, Gupta, 2012).

Većina razvojnih metodologija funkcionira na način da developeri prvo razviju kod, izrade testove te ih onda provedu. No, kod XP-a javlja se još jedan način, a to je TDD, tj. razvoj temeljen na testiranju (eng. test driven development). Ovdje, developeri prvotno izrade testni slučaj za neku funkciju i tek onda rade kod kako bi mogli proći taj test. Taj proces ponavljaju sve dok uspješno ne prođu taj test. Pokazalo se da, pokušavaju proći taj test, programeri lakše dođu do pronalaska i rješavanja problema te na taj način smanjuju broj bugova koji bi im, u klasičnom scenariju, promakli (Agile Alliance , 2023).

Iako se koristeći XP metodu može pokazati efikasnijim, fleksibilnijim i predvidljivijim načinom dolaska do rješenja, kao i sve ostalo, XP ima svoje nedostatke poput (Hutagalung, 2006)

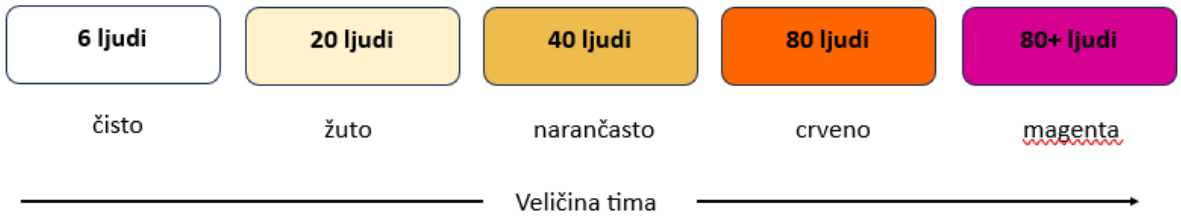
- Poteškoće kod koordinacije timova većih razmjera
- Ukoliko se njime ne upravlja ispravno može za rezultat imati projekt koji je nemoguće za dovršiti

- Zbog načina funkcioniranja XP-a (naglasak na kod), veće su šanse za manjkom dokumentacije
- Previđanje značajaka koje bi trebale biti dovršene u zadanom vremenskom periodu

Svi navedeni nedostaci nisu ništa novo u svijetu agilnog razvoja, to su tipični problemi svih agilnih metodika zato što koriste iterativni pristup. No unatoč tome, prema nekim istraživanjima pokazalo se da će upravo takav način, uz očite nedostatke, svejedno imati veću šansu za uspjehom upravo zbog tih iteracije zato što pružaju korisnicima rani uvid u proizvod koji se razvija i to je ono što se cijeni kod agilnog pristupa. (Hutagalung, 2006).

6.3. Kristalna obitelj metodologija

Kristalna obitelj metodologija smatra se agilnim okvirom (eng. *„Framework“*) koji je podosta lak sam po sebi, nema nekih složenih koncepata koji su potrebni za njegov rad. Glavni fokus ove metode jesu sami pojedinci i njihove međusobne interakcije, a sve se vrti oko toga koliko ljudi je uključeno i koji je potencijalni rizik za uspjeh projekta te prioritet. To sve označeno je bojama, a svaka od tih boja predstavlja jednu od metoda ove obitelji metodologija, odakle i naziv (Satyabrata, 2023).To je vidljivo na slici 8.



Slika 8: Prikaz pripadnika Crystal obitelji metodologija (prema Mrsic, 2017)

Ime su dobile po geološkim kristalima, a neki od njih su čisti, crveni, ljubičasti, žuti. Govoreći o povijesti koju ova metoda ima, razvijena je od strane čovjeka Alistaira Cockburna koji je je bio zadužen za to da proučava način na koji se softver razvija u tvrtki IBM davne 1991. godine te je na temelju svojih studija kasnije dokumentirano ovu metodu u svojoj knjizi „Crystal Clear: A Human-Powered Methodology for Small Teams“ (Wrike,).

Dakle, kao što je već rečeno, kristalna metodologija je zapravo grupa više metoda koje se razlikuju jedna od druge po veličini tima, tj. količini ljudi koja radi na projektu, a sukladno tome se određuje i veličina projekta. U suštini, svrha ove obitelji metoda je da pruža uvid u način vođenja koji bi mogao biti primijenjen na projektu, a ne nužno striktno određen skup pravila i krutu strukturu koja se mora pratiti. Nadalje, kako tim raste, promijeniti će se struktura razvojnog tima na projektu kao i sama složenost projekta. U tom slučaju vrlo je vjerojatno da će primjerice, tim prijeći iz korištenja Crystal Clear (tima od 6 ili manje ljudi) u korištenje Crystal Yellow metode gdje se u timu nalazi od 7 do 20 ljudi. Još jednom je ključno za napomenuti da kristalna obitelj metodologija samo navodi timove kako bi bilo dobro organizirati način na koji se upravlja projektom, te je iz tog razloga kao takva izuzetno fleksibilna i korisna (Kušek, 2010).

6.4. Kanban metoda

„Prije svega, Kanban je fleksibilan i stalno fokusiran na kretanje. Prioriteti se mijenjaju u skladu sa svakom novom dostupnom informacijom. U svakom trenutku možete vizualizirati, kontrolirati i optimizirati svoje radnje kako bi se projekt na kojem radite uvijek kretao u pozitivnom smjeru“ (KK Consulting, 2021).

Kanban je nastao u kasnim 1940. u tvornicama Toyote kao sustav za lakše baratanje rasporedom zahtjeva (Kanbanize, bez dat.). U doslovnom prijevodu s japanskog „kan“ i „ban“ imaju značenje riječi znak i ploča, što je ovaj pristup i predstavljao. U tvornicama su bile izložene velike ploče na kojima su bili obješeni papirići u raznim bojama od kojih je svaki sadržavao značenje o statusu zadatka ovisno o boji u kojoj je bio. Ovakav način uvelike je pomogao, kako pratiti statuse zahtjeva, tako i postaviti prioritete, prikazati neke važne datume, sitnije zadatke za daljnji razvoj i slično (KK Consulting, 2021).

Koristeći ovu tehniku prvo što moramo napraviti je kreirati Kanban ploču zato što je to zapravo mjesto gdje možemo vizualizirati sav rad koji treba biti obavljen, a minimalni broj stupaca po kojima se papirići na Kanban ploči dijele jesu:

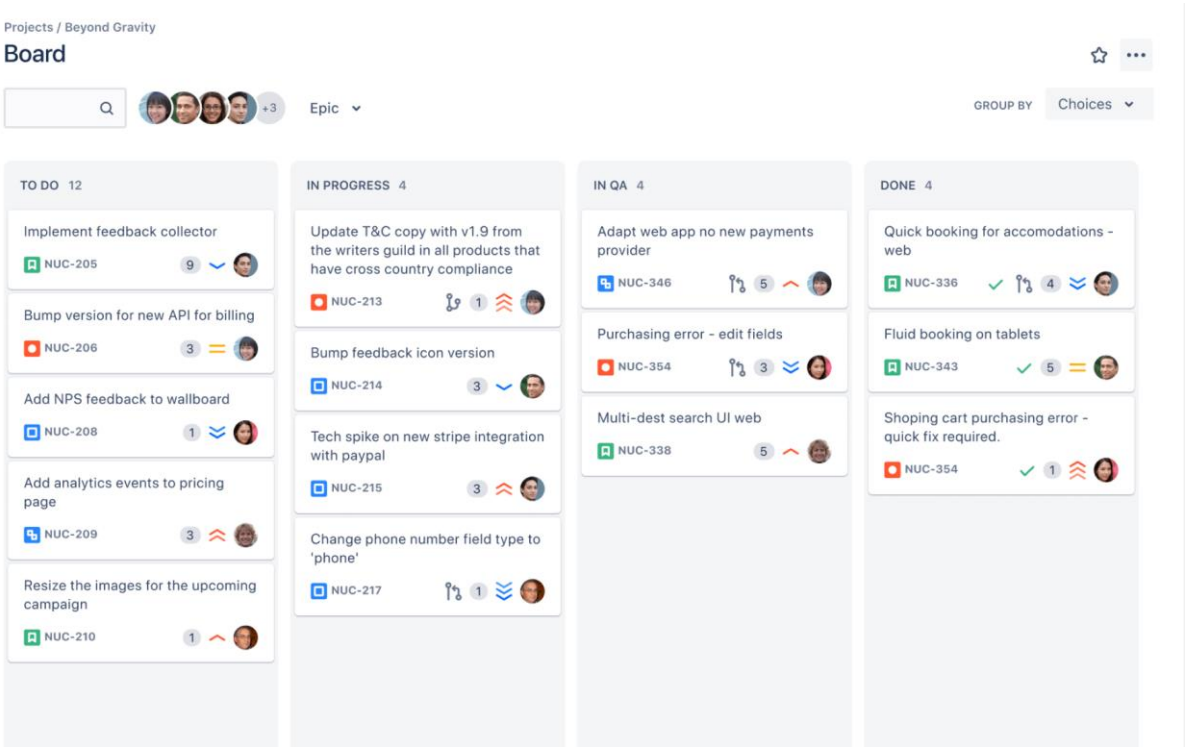
- Zatraženo tj. rad koji treba započeti
- U tijeku
- Završeno

Budući da fizički na ploči stane samo određen broj papirića, to bi značilo da je broj stavki koje su u tijeku (WIP-Work in Progress), ograničen. Zbog ove karakteristike Kanbana, kombiniranjem karakteristika Scrum-a, stvoren je Scrumban Budući da je u Scrum-u količina posla ograničena vremenom koje je određeno da će sprint trajati, kod Scrumban-a, nema vremenskih limita, već je kao i kod originalnog Kanban-a, ograničeno količinom posla koji stane na ploču (Parsons, Thorn, Inkila, MacCallum, 2018).

Na slici 9 jasno je vidljiv početni Kanban sustav iz neke od mnoštva japanskih tvornica automobila sa svojom Just in Time proizvodnjom.

Slika 9: Originalni Kanban sustav (Izvor: TOYOTA Global Website)

Kod Kanban tehnike sve se vrti oko neprestanog napretka gdje svi jasno razumiju što se događa na projektu. Budući da se na ploči koristi više manjih papirića, jasno je kakvi su zadaci na njima. Kanban metodom se, umjesto da, kao što je to uobičajeno kod tradicionalnih metoda, čini velike promjene odjednom, stvara set manjih inkrementalnih promjena kroz čitav razvoj projekta. Glavna prednost korištenja ove metode jest ta da je, zbog načina na koji funkcionira, cijeli projekt prikazan na jednoj ili više ploča čineći ga transparentnim, što s druge strane podiže razinu toka projekta i mogućnosti predviđanja. Isto tako, sukladno karakteristikama agilnih metoda razvoja, Kanban također potiče komunikaciju, kako unutar tima, tako i između razvojnih timova i korisnika sa svrhom da izbjegnu nesuglasice i probleme te da podignu zadovoljstvo i kvalitetu proizvoda (Kanbanize, 2023). Prikaz korištenja Kanban metode unutar alata Jira vidljiv je na slici 10.



Slika 10: Prikaz kanban ploče s projekta razvoja softvera u alatu Jira (izvor: Atlassian, 2023)

U nastavku rada detaljno ću objasniti agilnu procjenu zajedno sa tehnikama koje postoje za procjenjivanje trajanje projekata.

7. Agilna procjena

Kada govorimo o agilnosti i agilnoj filozofiji u radu neke tvrtke, misli se na skup vrijednosti i principa koji su osmišljeni kako bi u najvećoj mjeri pomogli ljudima da svoj posao mogu obaviti efikasnije i s manje muke. U širem smislu, agilna procjena se odnosi na profesionalno mišljenje o tome koliko je potrebno vremena, ovisno o njihovoj kompleksnosti, da neki dio projekta, zadatak, funkcionalnost na projektu, budu dovršeni.

Procjena je ključan aspekt planiranja te igra bitnu ulogu u provođenju agilnih metoda. Precizna procjena pomaže timu postaviti realistična očekivanja te bolje donošenje odluka kroz razvojni proces. Kao i sve ostalo, ovo može biti iznimno izazovno zbog određenih nesigurnosti koje se mogu javiti, a i zbog neočekivanog rasta u opsegu projekta. Agilne tehnike procjene služe za savladavanje ovih prepreka i te u konačnici uspješno finalizirati projekt (Kanbanize, 2023).

„Agilni luk“ prikazan na slici 10 najbolje slikovito prikazuje sve razine na kojima se može planirati projekt razvijen agilnim metodikama.

Slika 11: Agilni luk (prema Kanbanize, 2023)

U agilnom luku, kod vanjskog sloja, koji je **strategija**, dolazi do utvrđivanja strategijske vizije i ciljeva organizacije te kako će oni biti postignuti. Otkidanjem jednog sloja, dolazimo do **portfolija** gdje stariji menadžeri raspravljaju o portfoliju proizvoda i usluga koji će biti potpora prijašnjem sloju koji je kreiran. Na jednoj razini niže, kod **proizvoda**, timovi kreiraju plan i rastavljaju ga na manje djeliće s ključnim značajkama i funkcionalnostima kojima će u

konačnici doprinijeti ciljevima i strategiji koji su bili prethodno dogovoreni. Nakon toga, definiraju se ključne značajke koje će biti dio **isporuke** unutar nekog vremenskog perioda koji je bio dogovoren. Fokus sloja **iteracija** je da se pokuša odraditi posao koji je zadan unutar nekog razdoblja ili sprinta. Timovi tijekom iteracije odabiru probleme tj. korisničke priče i zadatke iz popisa funkcionalnosti koje će rješavati i kasnije isporučiti. I na zadnjem mjesto, u dubini luka, imamo **dnevne sastanke** čija je svrha da provedu tim kroz cijeli projekt te da dobe ažurne podatke o stanju na projektu. (Kanbanize, 2023).

Glavna zadaća agilne procjene jest da, kao što i sama riječ kaže, u mjeri vremena, procijenimo trud i napor koji je potreban za obavljanje nekog zadatka u popisu funkcionalnosti proizvoda, a kao rezultat te procjene, timovima je lakše planirati sprintove na projektu (Singh, 2022) Procjena i planiranje ključni su za uspjeh projekta u kojem razvijamo neki dio softvera. Plan koji kreiramo agilnim procjenama, ima svrhu voditi nas kroz naše investicijske odluke, a ponajviše jesmo li na pravom putu dostavljanja softvera na vrijeme i s funkcionalnostima koje su korisnici tražili od nas da budu dio samog projekta (Cohn, 2005).

Iako bez plana izlažemo svoj projekt mnoštvu problema, više puta se kroz provedbu planova, koji na prvu ruku, izgledaju savršeno i s malom šansom za neuspjeh, pokazalo suprotno. Od sredine druge polovice prošlog stoljeća, poznato je da su planiranje i procjena izrazito zahtjevni. Timovi dosta često odlaze u ekstreme kada se radi o planiranju rada na projektu, ili ne planiraju ništa, dopuste da stvari idu svojim tokom ili previše planiraju. Kada imamo na umu napor i muku koje planiranje i procjena projekta nose sa sobom, postavlja se pitanje: „Zašto uopće to raditi?“. Osim što organizacije stavljaju teret na nas da im isporučimo našu procjenu rada, kreiranje planova i rasporeda za izradu projekta dobro je kod kreiranja marketinških kampanja, zakazivanje okvirnog dana puštanja proizvoda u upotrebu. Kako bi developeri znali što trebaju izraditi, prije svega moraju raspravljati o značajkama, resursima koji su im na raspolaganju te rasporedu. No, to nije moguće sve odlučiti odjednom, već je potrebno da se to riješi iterativno i inkrementalno. To sve podupire dobar proces planiranja zato što njime možemo umanjiti rizik i nesigurnosti, stvara bolju koordinaciju unutar tima, pomaže nam da bolje donosimo odluke te služi za uspostavu povjerenja (Cohn, 2005)

Izrada agilne procjene izrazito je ključan dio razvoja projekta i uključuje cijeli tim. Svaki član tima, bilo da je developer, dizajner, tester ili da vrši neku drugu ulogu na projektu, donosi svoju, drugačiju perspektivu na proizvod kojeg se razvija te procjena koju oni donesu može uvelike promjeni finalnu odluku. Često se dogodi da vlasnici proizvoda zatraže nešto što se izvana čini kao jednostavna značajka, a zapravo je iza te značajke mnogo više posla nego što se nekom ne istreniranom oku može činiti. Stoga, važno je biti u stalnom kontaktu s vlasnikom proizvoda zato što on na temelju procjena koje razvojni tim donese za navedene zahtjeve, ima uvid u to koliko je truda potrebno uložiti u određene zahtjeve i sukladno tome može prilagoditi redoslijed njihova izvršavanja, tj. odlučiti koji su zadaci prioritetniji (Radigan, 2023). Uglavnom,

primarni cilj agilne procjene jest da poveća produktivnost i efikasnost svih segmenata projekta te da pokuša predvidjeti okvirno vrijeme završetka projekta.

7.1. Agilne tehnike procjene

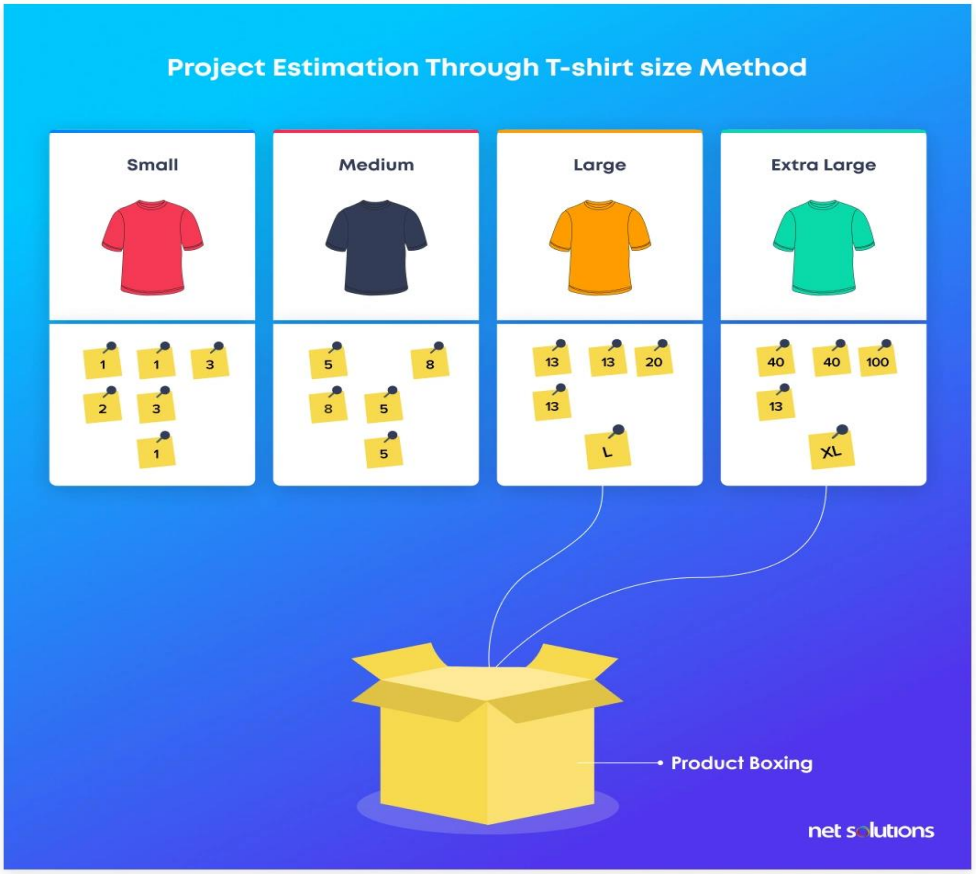
Vrijednost priče (eng. story point) je pristup koji je najučestaliji kod agilnih timova. Kod takvih timova, procjena završetka projekta i trajanja izrade zahtjeva, se ne računa u danima, tjednima ili mjesecima, već u story pointovima. Pomoću njih, oni mogu odrediti trajanje nekog zahtjeva na temelju truda koji moraju uložiti u njegovu izradu, složenost samog zadatka ili pak količini rizika nesigurnosti kojem se izlažu. Izbjegavanjem donošenja odluke o točnom datumu izrade određenog zahtjeva, iz jednadžbe je maknuta emocionalna vezanost za taj datum, što je dovelo do veće produktivnosti i donošenja više relativnih odluka i procjena. Iako možda na prvu ne zvuči privlačno ovaj pristup, pokazalo se pozitivnim zato što tjera timove da lakše donose teške odluke (Radigan, 2023). Nakon što se navede i raspravi o svim korisničkim pričama, na vlasniku proizvoda je da riješi sve nesuglasice koje su se potencijalno javile tijekom ovog procesa i da što preciznije objasni sve donesene zahtjeve kako bi se izbjegli budući sukobi i problemi kod izrade. Bitno je tijekom ovog procesa donijeti i odluku o tome koja tehnika procjene će se koristiti, a za tu svrhu se održava sastanak na kojem se raspravlja o kompleksnosti zahtjeva i broju korisničkih priča. Bitan faktor u tome je i iskustvo razvojnog tima te na što su sve spremni. Postoje brojne tehnike procjene, od kojih su neke: planiranje pokera, analogija, procjena veličine majice, Dot Voting, tehnika trokuta, Monte Carlo simulacija (Simplilearn, 2023).

7.1.1. Procjena veličine majice

Kada kao član tima želimo donijeti procjenu nekog zadatka na temelju njegove kompleksnosti i težine, taj proces dodatno nam otežava činjenica da ne znamo u kojoj jedinici mjere da to odradimo. Tada nam u pomoć uskače procjena pomoću veličine majice. Svi nosimo majice i otprilike znamo koji nam broj treba i najbolje odgovara te odmah znamo da je S puno manje od primjerice XL. Sada tu praksu moramo prevesti u procjenu okvirnog trajanja zadataka.

Ova tehnika funkcionira na način da, umjesto da za trajanje zadataka koristimo brojeve, s kojima se često asocira vrijeme, neki rok, dodatni pritisak, relativnim procjenama pomoću veličina majici, timovima je omogućeno razmišljanje u više dimenzija (slika 11). Dakle umjesto brojeva, zadacima se dodjeljuju veličine majica na temelju njihove kompleksnosti, truda koji je

potrebno uložiti, pod zadataka koje je potrebno obaviti prije samog započinjanja glavnog zadatka (Asana, 2023).



Slika 12: Slikovit prikaz metode procjene projekta veličinom majici (Izvor: Net Solutions, 2022)

Prvi korak procjene jest da se odrede veličine koje će se koristiti na projektu. Zatim, tim donosi odluku o tome što koja veličina predstavlja, je li to uloženi trud, složenost ili nešto treće (Indeed, 2023). Ovisno o strukturi tima, odlučuje se tko dodjeljuje veličine majici. Primjerice, za product backlog, vlasnik proizvoda dodjeljuje veličine zato što je on najbolje informiran o opsegu zadatka, za agilne timove tu je Scrum Master koji prije Sprinta provodi analizu donesenih procjena i zatim se procjena svodi na svakog od članova tima (Asana, 2023).

7.1.2. Monte Carlo simulacija

Kada kreiramo neki novi projekt, jedna od stavki koju je bitno odrediti jest njegovo trajanje kako bi naši budući korisnici ili potencijalni kupci okvirno, ako ne i precizno, znali kada će proizvod koji žele, biti završen. U tu svrhu, osmišljena je još jedna tehnika za procjenjivanje vjerojatnosti ishoda nekih događaja, Monte Carlo simulacija nazvana po svjetski poznatom casino gradu. Osobe koje su zaslužne za njezino osnivanje su John von Neumann, američki matematičar, i Stanislaw Ulam, američko-poljski matematičar. Tijekom drugog svjetskog rata, njih dvojica su bili prisiljeni doći na ideju metode kojom će unaprijediti donošenja odluka u

manje sigurnim situacijama te tako izvojevati pobjedu na bojištu (IBM, 2023). Tako mi danas u svijetu IT razvoja imamo metodu koja zamjenjuje varijablu nesigurnosti sa višestrukim vrijednostima, daje nam prosjek njih, te imamo broj koji nam može biti od koristi. Monte Carlo simulacija uvelike je korištena za procjenjivanje vjerojatnosti povećanja troškova na velikim projektima. Također ima i upotrebu u telekomunikacijskim tvrtkama, koje ih koriste za optimiziranje svojih mreža (Kenton, 2023)

Prvi korak kod ove metode jest postaviti naš model predviđanja na način na odredimo ovisnu varijablu, za koju želimo donijeti predviđanje, i neovisne varijable, a to su ulazne, rizične i predviđajuće varijable. Njih koristimo za samo pokretanje predviđanja. Ako je to moguće, analitičari se mogu poslužiti podacima iz prošlosti i/ili subjektivnom ocjenom analitičara kako bi definirali raspon vjerojatnih vrijednosti i svakoj od njih dodijelili težinu vjerojatnosti. Zatim, ponavljamo ovu simulaciju nekoliko puta, generirajući nasumične vrijednosti nezavisnih varijabli i tako sve dok se ne prikupi dovoljno rezultata da bi mogli stvoriti reprezentativni uzorak od gotov beskonačnog broja mogućih kombinacija (IBM, 2023).

Monte Carlo simulacije koje su po svojoj prirodi podosta konceptualno jednostavne, sposobne su rješavati izuzetno kompleksne sustave. Izrazito su korisne kod dugoročnih predviđanja zbog svoje točnosti te su dobra alternativa za strojno učenje u slučaju kada nemamo dovoljno podataka da učinimo strojni model za učenje funkcionalnim i točnim (Lutkevich, 2023)

7.1.3. Planiranje pokera

Veoma popularna tehnika koja se često koristi kod razvojnih timova naziva se planiranje pokera. To je u suštini gamificirana tehnika za procjenu vrijednosti bodova priče (Tran, 2023). Grenning kaže kako on smatra da je najbolji način za agilne timove da provedu procjenu nekog projekta, korištenje tehnike planiranje pokera (Grenning, 2002). Planiranje pokera ujedinjuje raznolikost mišljenja svih članova tima koji sudjeluju na projektu i kroz maksimalno tri iteracije, rijetko kada je to veći broj, donosi zajedničku procjenu (Cohn, 2005). Idealna je za procjenjivanje relativno malog broja zadataka (maksimalno 10) i za timove do osmero ljudi. U slučaju da je više od toga, preporuča se da se tim podijeli na dvije manje grupice ljudi koji će onda odraditi isti posao (Simplilearn, 2023). Funkcionira na način da se svakom od članova dodijeli set karata sa brojevima Fibonaccijevog niza od 0 do 100 (1, 2, 3, 5, 8, 13, 20, 40, 100). Nakon toga, najpoželjnije je kada je na tom mjestu vlasnici proizvoda, ali nije nužno zato što to nije toliko bitna zadaća, opisuje korisničke priče timu, koji naravno mogu postaviti pitanja na tu temu. Kada je korisnička priča iznesena, svaki od članova, bez znanja ostalih, odabire jednu od kartica koji predstavlja njegovu procjenu trajanja određenog

zadatka. Kartica s najviše glasova biti će postavljena kao zaključna procjena trajanja toga zadatka (Singh, 2022). Nije rijetkost da se procjene nekih članova tima ne poklapaju i to je zapravo dobra vijest, zato što u tom slučaju ti članovi čije se procjene razlikuju mogu raspravljati zašto su dali te odgovore, što je zapravo i cilj ove tehnike, poticanje na raspravu. Na taj način, ako su obrazloženja zadovoljavajuća, član tima koji je dao višu procjenu ili član koji je dao nižu, mogu prilagoditi svoje procjene te tako doći do nekog zajedničkog broja za koji se obojica mogu složiti da odgovara prezentiranom zadatku (Cohn, 2005).

Zanimljiv način primjene ove tehnike pronašao je Kent Beck, koji je predložio da se na zid s obavijestima objesi koverta sa zadacima koje je potrebno evaluirati. Članovi tima, ovisno o svojim mogućnostima i slobodnom vremenu, u nasumično doba dana mogu uzeti iz koverte jednu ili dvije korisničke priče te do kraja dana mogu dostaviti svoje procjene (Cohn, 2005).

U idućem poglavlju ću objasniti koji problemi nastaju kod agilnog planiranja.

7.2. Izazovi i greške agilnog planiranja i procjene

Agilno planiranje je postalo sveprisutno u razvoju softvera i mnogim drugim industrijama zbog svoje sposobnosti da prilagodi projekte promjenjivim uvjetima i zahtjevima okoline. Međutim, iako agilne metodike pristupa mogu donijeti brojene koristi u svijetu softverskog razvoja, njihova procjena i planiranje nezaobilazno se susreću s nizom izazova i grešaka koje mogu otežati, ali potencijalno i ugroziti uspjeh projekta.

Neki od izazova agilne procjene su sljedeći (Tzemach, 2021):

- Ljudska priroda
- Ego
- Nemogućnost prihvaćanja relativne procjene
- Razne unutarnje i vanjske zavisnosti
- Nesigurnosti
- Rad pod pritiskom

Ljudi po svojoj prirodi imaju potrebu da budu ili optimisti ili pesimisti, dok ljudi koji su dobri u procjenjivanju obično teže biti realisti. Međutim, ako ste ikada bili dio agilnog procesa procjenjivanja unutar agilnog tima, već biste trebali znati da između članova tima uvijek postoji ego. Jedan klasičan primjer toga je "znalac" u timu koji smatra da zna bolje od svih ostalih i stoga odbacuje mišljenja drugih, nebitno bili oni u pravu ili ne. Također, neuspjeh u prihvaćanju modela relativnih procjena može usporiti razumijevanje agilnih timova o stvarnom značaju i različitim tehnikama i metodama za njegovo korištenje. Zavisnosti, kako interne tako i eksterne, poput dobavljača, integracije s drugim timovima i međusobne zavisnosti između priča, mogu značajno utjecati na sposobnost tima da procijeni određenu priču jer tim više nije

ovisan samo o sebi. Nesigurnosti često igraju ključnu ulogu u bilo kojem procesu procjene, a važno je razumjeti da mogu izravno utjecati na procjenu tima tijekom samog procesa. Rad pod pritiskom, bilo od strane menadžmenta, od strane samog tima ili od strane "zlostavljača" (eng. Bully) s jačom osobnošću iz drugog tima, također može utjecati na sposobnost tima da provodi učinkovit i praktičan proces procjene. U teoriji, Scrum tim se sastoji od različitih ljudi koji imaju potrebne vještine i znanje za procjenu priča i isporuku istih. Međutim, potrebno je vremena da se takvi timovi formiraju, a čak i tada možda neće imati sva potrebna znanja za učinkovit proces procjene (Tzemach, 2021).

Kada je u pitanju agilno upravljanje projektima, iako vrlo učinkovito u promoviranju prilagodljivosti i suradnje, suočava se s nekoliko ključnih izazova. Prvo, postoji potreba za uravnoteženim pristupom planiranju. Iako agilnost naglašava fleksibilnost, nedostatak početnog planiranja može otežati odobrenje projekta i procjenu troškova. Agilni timovi trebaju uključiti planiranje unutar okvira planiranja sprintova i kontinuirane prilagodbe, uključujući obuhvaćanje zahtjeva, dizajn, arhitekturu i procjene vremena i troškova za sprintove i prekretnice. Drugo, promjene u zahtjevima mogu poremetiti tijek projekta, posebno ako se pojave kasno u sprintu. Učinkovita analiza rizika i regresijsko testiranje trebaju se koristiti kako bi se ublažio taj izazov, osiguravajući da se ključne značajke stavljaju na prvo mjesto i testiraju rano (Pandey, 2022).

Još jedna velika prepreka je nedostatak podrške menadžmenta, jer vođe organizacije moraju razumjeti i aktivno podržavati prijelaz na agilne metodologije. Osim toga, odjeli ljudskih resursa trebali bi koristiti agilne tehnike prilikom zapošljavanja radnika ili promoviranja trenutnih. Agilne metodike su korisni prilikom ove vrste posla zato što omogućavaju pojedincima da se istaknu te daju prednost zaposlenicima koji motivaciju crpe iz stvarnog rada na projektu, radom s nekim posebnim voditeljem ili grupom s kojom imaju najbolje rezultate. Česte regresijske cikluse zahtijevaju učinkovite testne prakse, a dosljednost među članovima tima i odjelima od ključne je važnosti za održavanje zadovoljstva korisnika. Odjeli za financije mogu se suočiti s izazovom nesavladive naravi agilnih projekata, što zahtijeva poboljšanu komunikaciju i restrukturiranje. Konačno, varijacije u performansama mogu se pojaviti zbog zanemarivanja potreba krajnjih korisnika, što naglašava važnost testiranja opterećenja i automatiziranih tehnika za identifikaciju i rješavanje problema s performansama. Rješavanje ovih izazova ključno je za uspješno upravljanje agilnim projektima (Pandey, 2022).

Kako bismo ujedinili agilnu procjenu i agilne metodike te sve zajedno pristupačno i na zanimljiv način prikazali, potreban nam je izuzetno moćan alat. Neke od njih, najpopularnije i najkorištenije detaljnije ću opisati u poglavlju koje slijedi.

8. Alati za agilni projektni menadžment

U svijetu današnjice, mnoge softverske tvrtke u svoje su poslovanje implementirale korištenje agilnih metoda razvoja i tehnologija za kreiranje svojih proizvoda. To im je omogućilo da brže, lakše i sigurnije odrade posao (Mihalache, 2017). Kada koristite neku od agilnih metoda, treba vam i moćan alat koji će je savladati. Tu u pomoć dolaze razni alati za agilni razvoj koje tvrtke moraju imati kako bi cijelo poslovanje teklo glatko. Ovime su stvorili bolji uvid u samo poslovanje, unaprijedili kolaboraciju timova i članova unutar tima i povećali transparentnost rada (Kanbanize, 2023).

Prema nekim istraživanjima, pokazalo se da su alati za agilni razvoj izvrsna zamjena tradicionalnim metodama koje su kritizirane da se ne fokusiraju na pravu vrijednost poslovanja. Isto tako, kritizirane su i agilne metode da ne podupiru cjelovit životni vijek sustava. Kasnije je dokazano da su ipak agilne metode dovoljno dobre da na vrijeme i s pravilnim pristupom maksimiziraju poslovnu vrijednost. Prema Alexandri Mihalache, na provedenom uzorku od više od šest tisuća ljudi, anketa je pokazala da su alati za agilni menadžment u velikoj potražnji kod ljudi koji se bave više od jednim agilnim projektom istovremeno te da im u velikoj mjeri pomaže organizirati i svoje vrijeme i vrijeme koje treba biti utrošeno na izradu zadataka. Izbor alata za kreiranje agilnih planova ovisi o timu, no pokazalo se da samo korištenje tih alata uvelike povećavaju suradnju, razvoj i olakšavaju upravljanje. Posebice iz razloga što se fokusiraju na, kako na međusobnu komunikaciju članova tima, tako i izvan tima, s korisnicima i klijentima, čime se grade čvrste veze i odnosi između tih partija. Bitno je za spomenuti da je to je ujedno i jedna od glavnih značajki agilnih metoda razvoja (Mihalache, 2017).

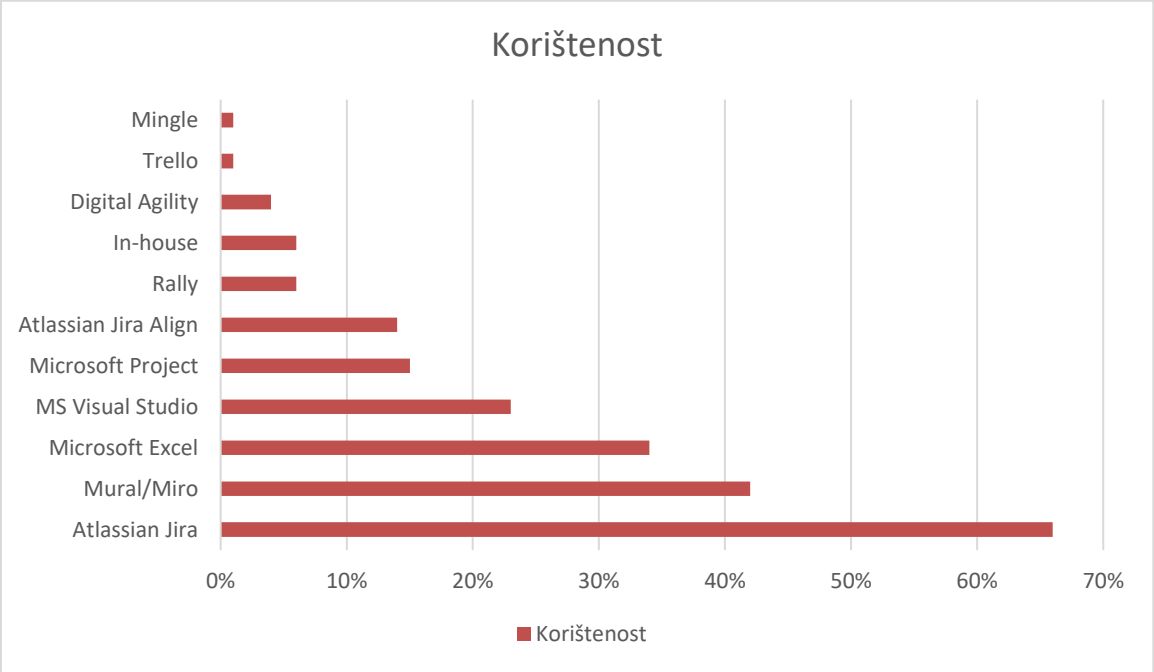
Alati za agilni razvoj su moraju biti mnogo drugačiji od alata za tradicionalni razvoj. Razlog tome leži u načinu na koji agilne metode funkcioniraju, sukladno tome, agilni alati se moraju prilagoditi dinamičnosti i prilagodljivom načinu rada te brzoj i efikasnoj razmjeni informacija. Veći i kompliciraniji projekti zahtijevaju upotrebu moćnih specijaliziranih alata za svoje upravljanje i dijeljenje informacija. Na sreću, danas je to lako za pronaći zato što postoje razni programi za komercijalnu i profesionalnu upotrebu, od sofisticiranih aplikacija fokusiranih na razvoj web aplikacija do jednostavnih i specijaliziranih (Mihalache, 2017).

Prema rezultatima istraživanja Azizyan, Magarian i Kajko-Mattson (2011) pokazalo se da je izbor alata za agilni menadžment veoma ključan kako za finalni ishod projekta, tako i za developere i menadžere na svim razinama. Također, rezultati su pokazali kako timovi koji se nalaze na istoj lokaciji, primjerice u istoj tvrtki, kao alat koriste jednostavnije alate za agilni menadžment i u svakodnevnoj komunikaciji, češće koriste fizički zid i papir nego virtualne ploče ili alate. Dok su kod timova koji su distribuirani na više lokacija, što i ima smisla, u većoj mjeri korišteni alati za agilni projektni menadžment kako bi srušili tu barijeru udaljenosti i nesmetano

mogli komunicirati u svakom trenutku. U nastavku rada pobliže ću opisati neke alate, prikazati učestalost njihovog korištenja u poduzećima te detaljnije opisati najpopularnije.

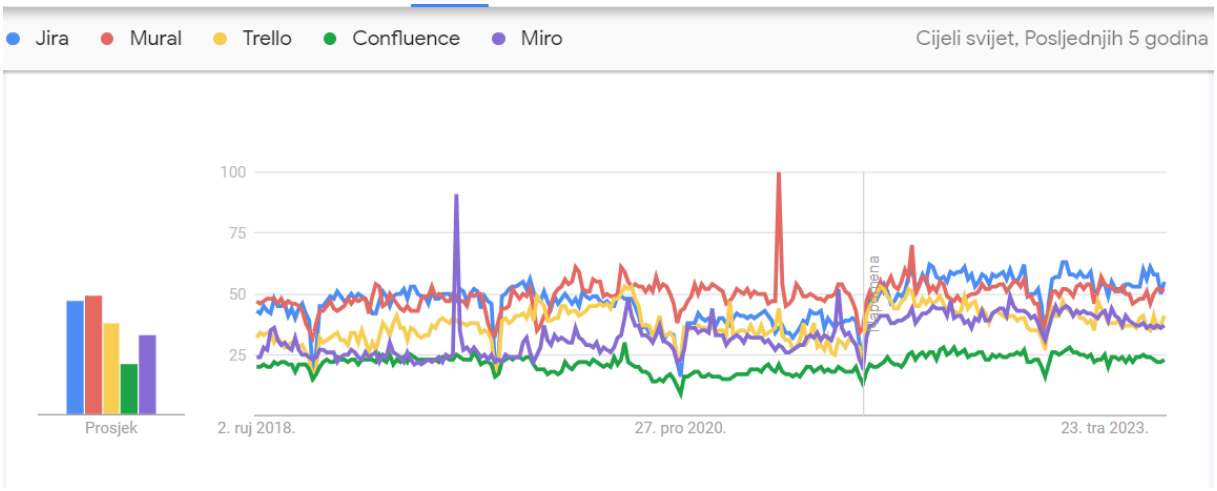
8.1. Usporedba alata

Količina alata za agilni razvoj koja nam je dostupna je stvarno velika. U ovom poglavlju prema izvješćima koja su nam dostupna na internetu možemo izdvojiti nekolicinu najpopularnijih alata koji se koriste u softverskom razvoju kao i u ostatku IT područja. Sukladno tome, na dijagramu 5. prikazani su podaci State of Agile izvješća iz 2022. godine iz kojeg se jasno vidi da čak dvoje od troje ljudi koji su sudjelovali u ovoj anketi, koristi Jiru kao glavni softver za provođenje agilnog razvoja unutar svog tima ili poduzeća.



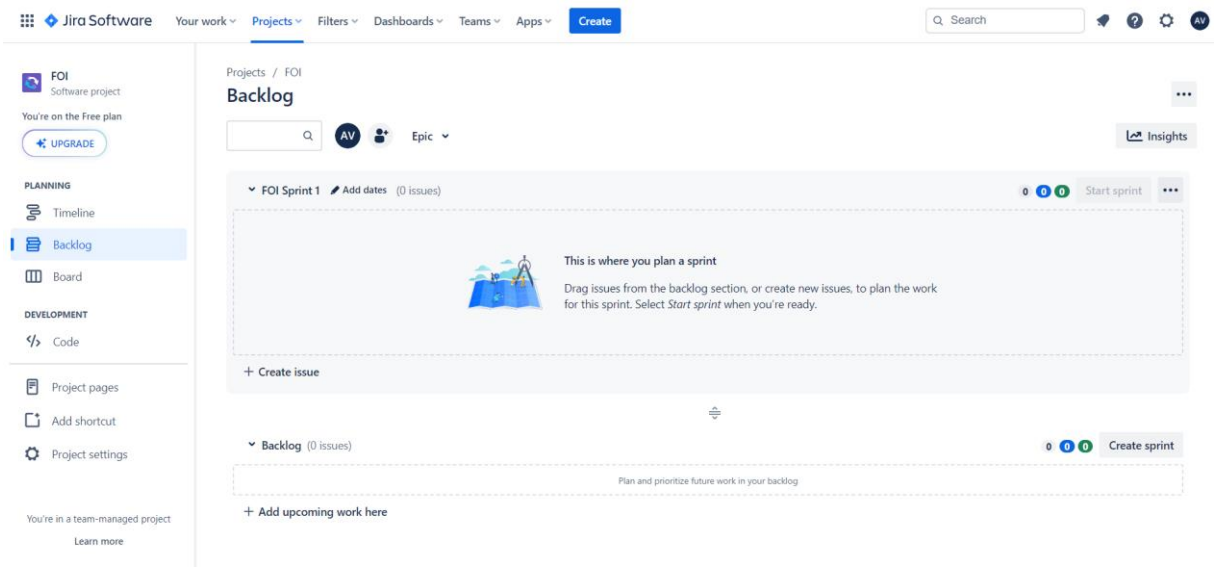
Dijagram 5: Usporedba korištenja agilnih alata (prema State of Agile, 2022.)

Na slici zaslona 1 vidljiv je interes kroz vrijeme unazad 5 godina za cijeli svijet te kako se kretao u tom vremenskom periodu. Rezultatima dobivenim google trends alatom možemo vidjeti popularnost pretraživanja određenih pojmova, sukladno tome, jasno je vidljivo kako dobiveni rezultati odgovaraju rezultatima iz State of Agile izvješća te Jira i dalje dominira u svijetu alata za agilni razvoj.



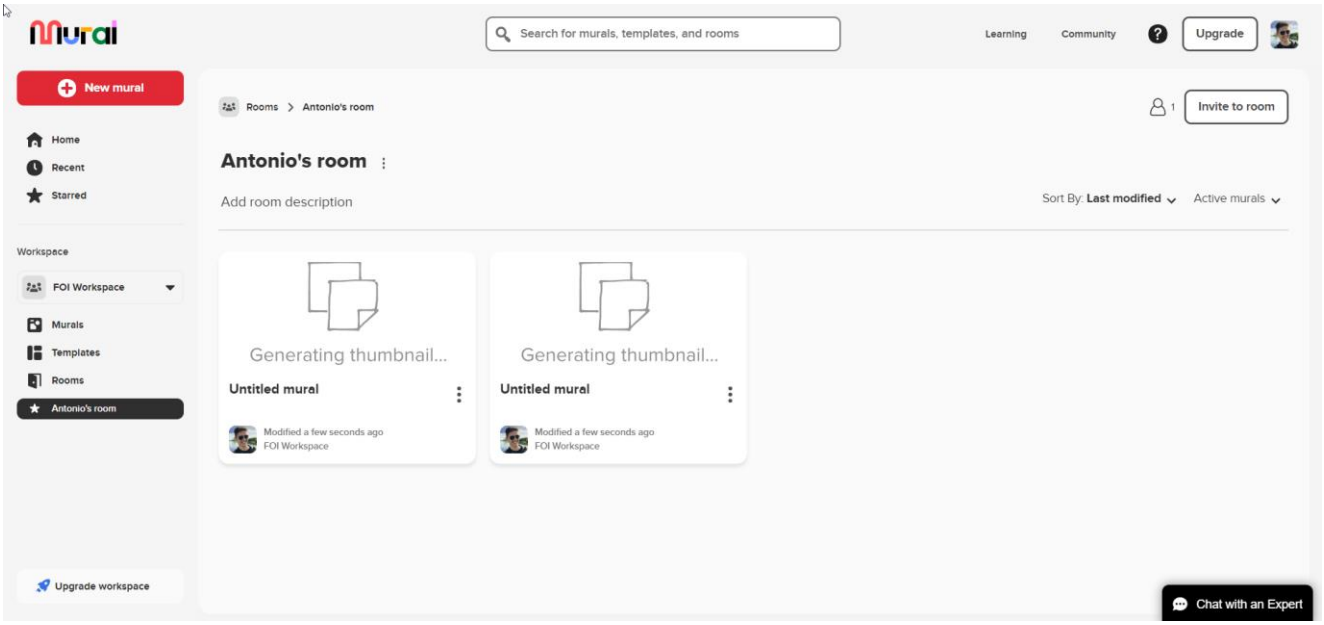
Slika zaslona 1: Usporedba alata po Google pretraživanjima (izvor: Google Trends)

Sudeći prema ovim informacija, detaljnije ćemo usporediti najkorištenije alate, a ostale ukratko opisati, počevši sa Jirom. Dakle, **JIRA Software** (Slika zaslona 2) osim što je jedan od najpopularniji alata današnjice u agilnom menadžmentu, također je i među najmoćnijima te nudi mnogo raznih značajki koje mogu biti iznimno korisne za tvrtke (Ozkan, Mishra, 2019). Razvijena je 2002. godine od tvrtke Atlassian koja je prvotno krenula kao tvrtka za tehničku podršku. U svojim početcima imali su problema s financiranjem pa su prodali svoj softver kako bi krenuli u razvoj Jire. Ideja za ime je potekla od riječi „Bugzilla“, koji je bio naziv softvera za uklanjanje „bugova“. Jira je prvotno bila osmišljena da pomogne programerima u planiranju razvojnih rješenja, no kasnije se razvila na razna druga područja te su sada dostupni softveri za poslovne timove (Jira Work Management), timove koji nude IT usluge (Jira Service Management) kao i timove za upravljanje isporučivanjem proizvoda (Jira Align). (ProductPlan, 2023). Nitko ne voli greške kod svog planiranja, pogotovo one koje nakon mukotrpnog i dugotrajnog traženja, nikako ne možemo otkopati. Sa Jirom, i to je postalo problem prošlosti. JIRA nam nudi informacije o greškama u stvarnom vremenu i daje nam načine kako bi mogli određeni segment projekta optimizirati. Na temelju statistika i sakupljenih informacija, daje precizne procjene o trajanjima sprinta ili o načinu na koji stvari rade. Uza sve to, kao i mnoštvo drugih alata, omogućuje praćenje grešaka kroz razvojni proces (Mihalache, 2017).



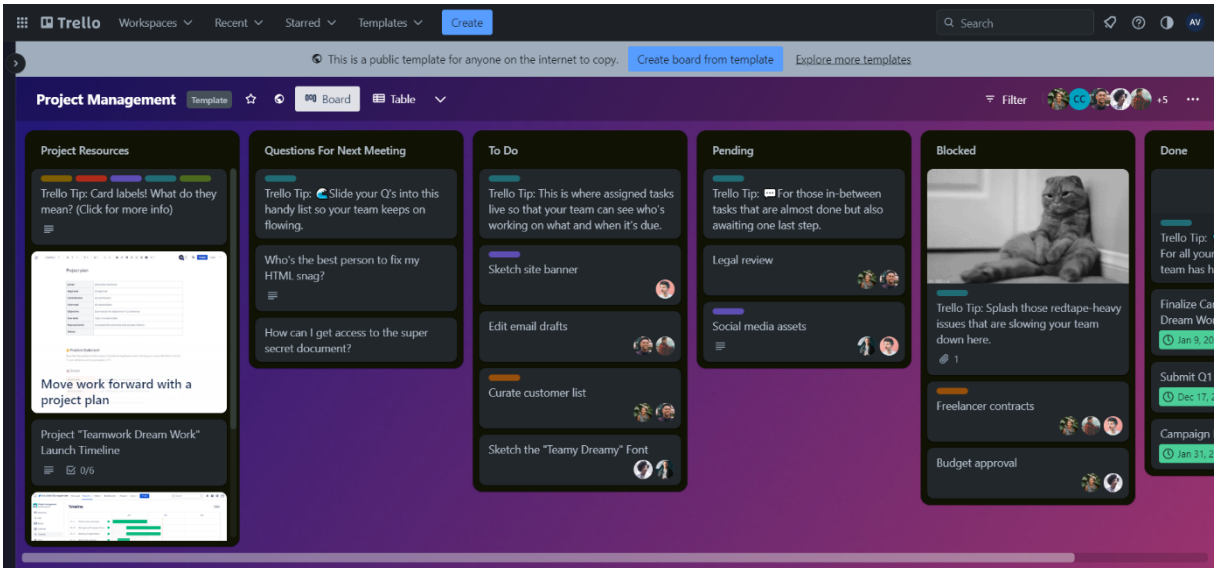
Slika zaslona 2: Početni zaslon nakon kreiranja Scrum projekta u Jira alatu (https://avalent.atlassian.net/jira/software/projects/FOI/boards/2/backlog)

Zatim imamo **Miro** i **Mural** (Slike zaslona 3) alate. Virtualne ploče (eng. Whiteboard) i alate za suradnju koji se smatraju jednima od najboljih na tržištu kada ste u potrazi za alatom koji vam nudi sve za organizaciju ideja prilikom kreiranja softvera. Osim toga, nude olakšano kreiranje dijagrama, crteža, dijagrama toka i slično. Ukoliko korisnik to želi, može koristiti već izrađene predloške koji su mu dani na slobodno korištenje te najbitnije, nude virtualnu komunikaciju i suradnju s timovima. Svaki od alata ima značajke koje ga odvajaju od ostalih, ali na kraju odabir alata koji će se koristiti ostaje na tome što nam na tom projektu treba, što želimo postići, koji su nam ciljevi te prioriteti. Također ovisi i o tome ima li klijent neki od softvera koji je prije koristio ili ne pa se i prema tome prilagođava. Nastoji se koristiti Mural za ljude koji su relativno novi u korištenju sličnih alata, a Miro za veće i duže projekte (Eich, 2023).



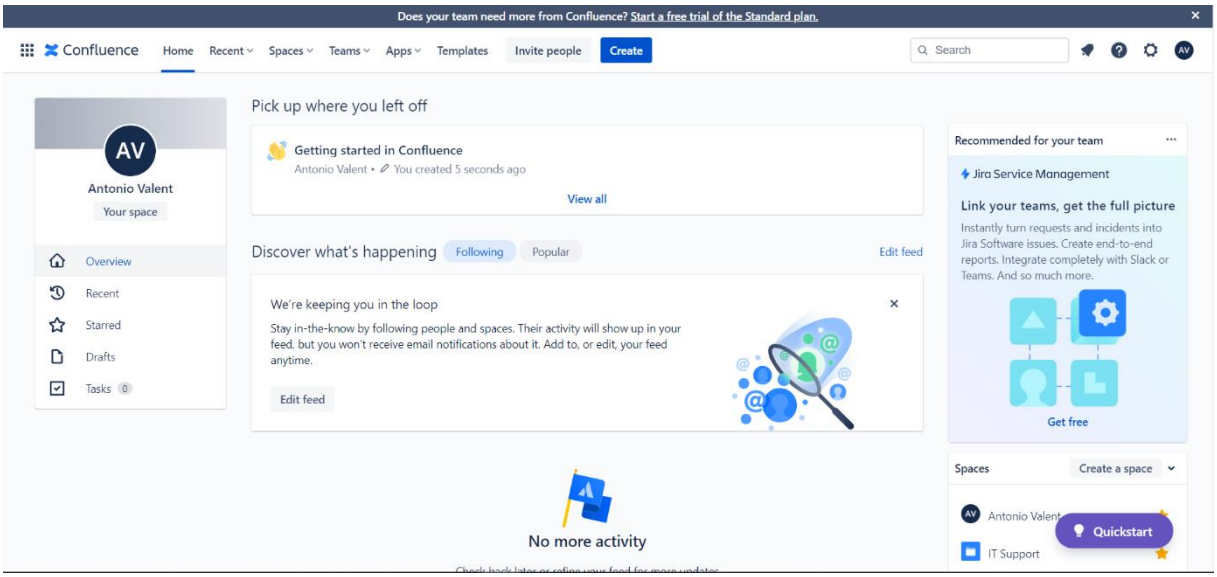
Slika zaslona 3: Početni zaslon alata Mural
(https://app.mural.co/t/foiworkspace6268/r/1693388276274)

Po popularnosti slijedi **Trello** (Slika zaslona 2), još jedan alat popularne tvrtke Atlassian. Problem s kojim se susreću mnogi softveri jest kompleksnost i slabo razumijevanje korisničkog sučelja, korištenjem Trella, olakšan je pristup i cijeli pregled procesa. Poseban po svojoj jednostavnosti korištenja, moćan alat koji kao i njegovi konkurenti, omogućuje timovima, koji ga odluče koristiti, organiziranje projekata i sve vezano za njih. Svojim korisnicima, njih 40 milijuna, pruža bogat pregled informacija, od slika i privitaka do rokova i ostalih statusa zadataka koje je nužno pratiti tijekom razvoja projekta. Podržava povezivanje s mnoštvom aplikacija i usluga treće strane (eng. third-party) poput Google Drive-a, OneDrive-a, Github-a, itd. Važna značajka koju nudi jest povezivanje ploča kreiranih unutar servisa sa ostalim softverima za agilni menadžment, poput Jire ili Confluence-a te time dodatno olakšava praćenje događaja na projektu. Svojstvo zbog kojeg je možda toliko u upotrebi jest to da unutar sučelja povezuje sve timove u kojima se pojedinac nalazi i daje mu ažurne podatke o situaciji unutar tima. Također je dostupan i u mobilnoj verziji što ga čini izuzetno lako dostupnim i prihvatljivim (Atlassian, 2023).



Slika zaslona 4: Početni zaslon nakon kreiranja projekta u alatu Trello (https://trello.com/b/1x4Uql2u/project-management)

Confluence (Slika zaslona 3), još jedan vrhunski program za agilni menadžment razvijen od strane tvrtke Atlassian, tvrtke koja je zaslužna i za razvitak programa poput JIRA – e, Trella i još mnoštva drugih korisnih aplikacija. Promovira se kao proizvod s kojim timovi mijenjaju način na koji surađuju, a to postižu time što svojim korisnicima pružaju lakoću suradnje, komuniciranja i rada na zajedničkim projektima. Pomoću Confluence – a, ljudi koji rade na istom projektu u svakom trenutku mogu pristupiti svim podacima i dokumentima koji su dostupni te ih organizira sa svojim predlošcima na način da budu što efikasniji te tako svi mogu biti u toku s razvitkom. Ovakvim načinom rada, kada se u stvarnom vremenu mijenjaju podaci i ažuriraju statusi, teško je ostati u doticaju s napretkom projekta, no Confluence je i za to pronašao rješenje tako što korisnici u svakom trenutku mogu pratiti napredak, upravljati projektima i organizirati tim na što optimalniji način. Isto tako, kada bi tim poslao neki „project proposal“ na pregled, morao bi čekati dugo vremena prije nego što bio dobio povratne informacije, a ovako u stvarnom vremenu unutar dokumenta mogu vidjeti komentare odmah kako su dodani. S ciljem da povećaju razinu kompatibilnosti, tvrtka Atlassian je stvorila trgovinu za ekstenzije i plugin – ove za „3rd party“ aplikacije, uz to i integraciju s JIRA softverom te time učinila Confluence još lakšim za uklopiti u svoju radnu okolinu i organizaciju. Kako bi ovakva suradnja između članova bila moguća, program mora biti što dostupniji na što više platforma te je tako moguće Confluence instalirati na svom računalu, laptopu i mobitelu te svugdje biti sinkroniziran u svakom trenutku (The Business Blocks, 2023).



Slika zaslona 5: Početni zaslon u alatu Confluence (https://avalent.atlassian.net/wiki/home)

U tablici 2 ukratko su opisani i prikazani ostali alati za agilni razvoj koji su manje popularni, ali ne toliko manje, ako ne i jednako, snažni. Svaki od njih ima svoje prednosti i mane, ali odabir onog za korištenje ovisi o preferencijama pojedinca ili tvrtke.

ActiveCollab	Komercijalan, web – based, alat, moćan i lak za korištenje s pouzdanim sučeljem. Tvrtke ga koriste kada im treba alat za planiranje i praćenje napretka, upravljanje zadacima, kontroliranjem ulazne pošte i slično. Komunikacija s ostatkom tima uvelike je olakšana i lako dostupna (Shaikh, Khan, Farah Siddiqui, Quershi, 2021).
Agilo for Scrum	Idealan je za timove koji se nalaze na raznim lokacijama zato što pruža uvid u Scrum ploče u stvarnom vremenu, odmah kako su izmijenjene, s veoma malim vremenskim odstupanjima. Isto tako podržava i pregled aktualnih proizvoda i sprinteva koji su u tijeku (Ozkan, Mishra, 2019).
Pivotal Tracker	Savršen je za mobilne i web developere zato što s lakoćom prati sve događaje do trenutka isporučivanja njihove aplikacije. Glavna značajka kojom se mogu pohvaliti jest da odličnim procjenama i određivanjem važnosti zadataka (Shaikh, Khan, Farah Siddiqui, Quershi, 2021).
Ice Scrum	Open – source agilni alat koji u svojoj suštini spaja principe Kanbana, Scuma – a i XP – a (Extreme Programming) kako bi mogao timovima ponuditi efikasno upravljanje task – ovima,

	organizaciju svog vremena, a i vremena obavljanja zadataka na projektu. Korisnicima nudi razne značajke poput poharne u oblaku, praćenje product backlog – a, popis izvješća, isporuka, sprinteva,i sličnog. Omogućava lako korištenje sa softverima poput JIRa – e, Jenkins – a, Remine – a, itd (Ozkan, Mishra, 2019).
VersionOne	Izrazito koristan, a i moćan softver korišten u profesionalnoj okolini, najčešće od strane timova koji se ne nalaze na istim lokacijama, a koji moraju biti u toku s napretkom projekta. Njegov korisnici mogu svoje projekte izraditi pomoću Kanban metode ili hibridno, spojem Kanban – a i primjerice Scrum – a (Shaikh, Khan, Farah Siddiqui, Quershi, 2021).
Taiga	Također jedan open – source alat idealan za razvojne programere, dizajnere. Kao i VersionOne, podržava Scrum i Kanban pristup na projektu (Ozkan, Mishra, 2019).
Planbox	Agilni alat koji je kompatibilan jednako za male i srednje tvrtke kao i za velike. Svojim moćnim značajkama i funkcionalnostima, članovima tima omogućava lako integriranje razvoja softvera u agilni ciklus. Također, uključuje značajke poput to – do popis zadataka, vremensku crtu, nadgledanje iteracija, sprinteva, proračun procjene, razne prilagodbe i slično (Ozkan, Mishra, 2019).
Asana	Jednostavan i lak za korištenje, Asana alat je jedan od najboljih kada je u pitanju upravljanje zadacima na projektu. Timovi koji odluče koristiti Asanu u svrhe izrade zadataka, ostvaruju poboljšanu komunikaciju i koordinaciju te praćenje razvoja zadataka, a uz to imaju priliku koristiti njene posebnosti (Ozkan, Mishra, 2019).

Tablica 2: Ostali manje korišteni alati (izrada autora)

U poglavlju koje slijedi, na praktičnom primjeru prikazat ću izradu projekta u već dobro poznatom alatu, Jiri, tvrtke Atlassian.

9. Praktični dio

Kao što je navedeno na početku rada, praktični dio rada napravljen je u Jira alatu za agilni razvoj. Zadatak koji je riješen u nastavku dobiven je od sumentorice iz tvrtke IBM iX, a on glasi:

„Projektni tim dobio je zadatak izraditi rudimentaran webshop u kojem korisnik može dodavati niz željenih proizvoda i obaviti kupnju. Korisnik webshopa se može registrirati na stranicu kako bi obavio kupnju iako može kupnju dovršiti i kao gost.

Webshop se sastoji od glavne stranice s tekstualnim i multimedijskim sadržajem (Header, Footer, Article, Teaser), listom proizvoda koje je moguće filtrirati (Product Listing Page) i stranicom s više detalja za svaki proizvod (Product Detail Page). Korisnik u svakom trenutku može otvoriti košaricu i pregledati sadržaj, a kada je spreman može završiti kupnju odabirom opcije Checkout. Registrirani korisnik treba se moći logirati u sustav i pregledati svoje registrirane podatke kao i podatke o svojim kupnjama.

Izrada webshopa je definirana i podijeljena po Epic zadacima:

- *Navigation Menu & Footer*
- *Content & Service Pages*
- *Homepage • Product List Page*
- *Product Detail Page*
- *Registration*
- *User login/logout*
- *My account*
- *Cart*
- *Checkout*

Projektni tim radi na nizu zadataka planiranih unutar Sprinta i na kraju Sprinta prezentira kreirani sadržaj klijentskom timu. Sprint traje 2 tjedna.

Projektni tim čine:

- *1 Scrum Master*
- *1 Product Owner*
- *1 DevOps Engineer*
- *2 back-end Developer*
- *2 front-end Developer*
- *2 Quality Assurance Engineer / Tester*

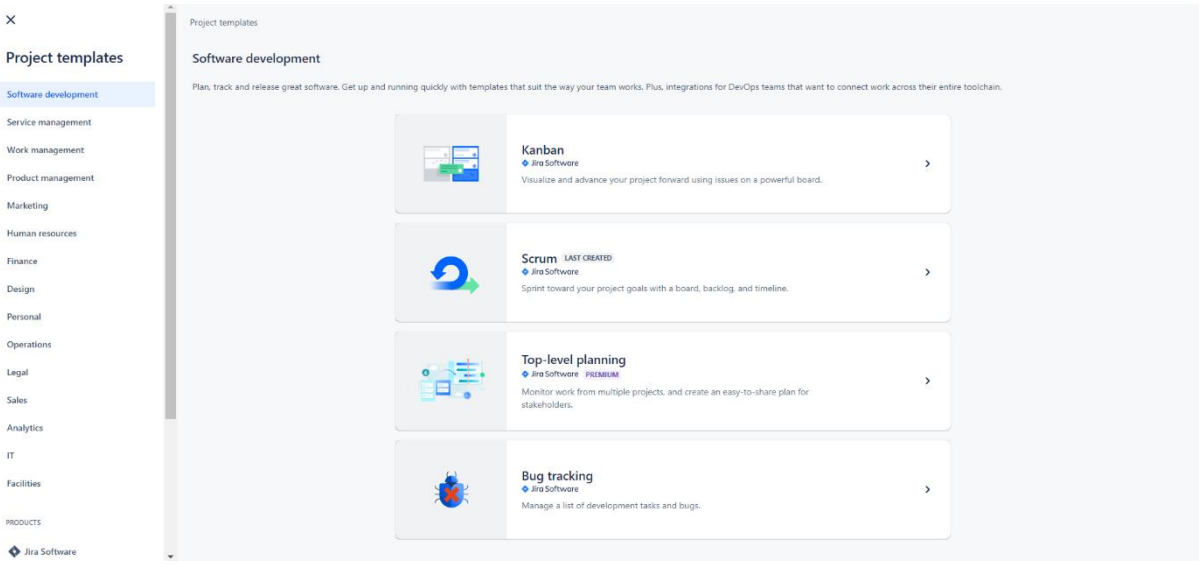
Potrebno je:

1. Definirati (pod)zadatke za svaki Epic i svakom zadatku dodijeliti pripadajuću Story Points vrijednost kako bi bilo moguće planirati Sprinteve.

2. Prema dostupnim podacima predložiti plan izvršenja zadatka. U koliko sprinteva projektni tim može odraditi zadane zadatke iz backloga? Na koji način biste planirali popravljanje bugova?

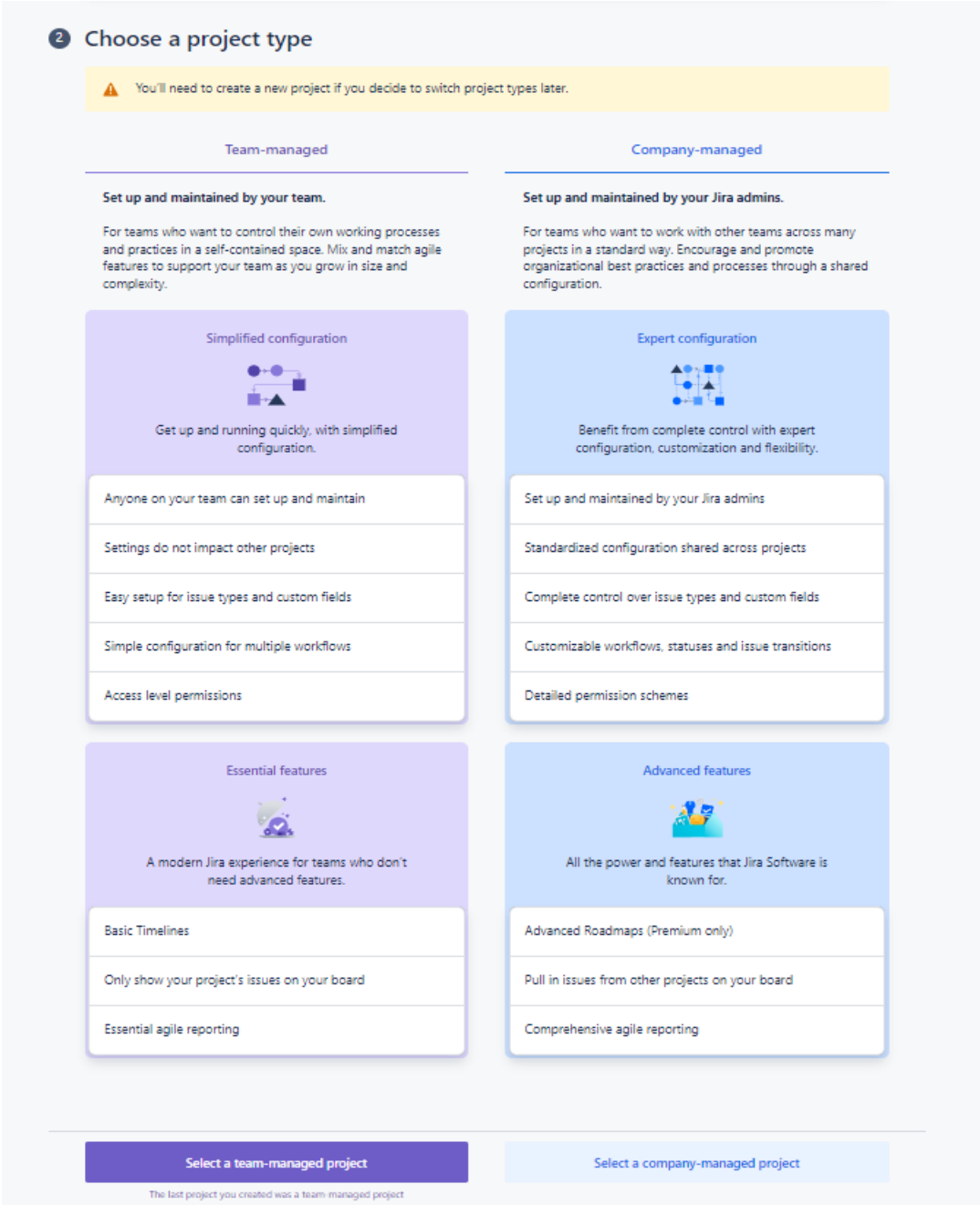
Napomena: S obzirom na to da se s izradom webshopa započinje ispočetka potrebno je pripaziti na raspored izrade komponenti jer su neke komponente ovisne jedna o drugoj.“

U skladu s time, kako bi započeli s izradom projekta, moramo ga stvoriti. To možemo učiniti pritiskom na gumb „Create project“ te odabirom predloška tj. omogućeno nam je da odaberemo metodiku agilnog razvoja s kojom želimo nastaviti. Projekt će biti kreiran Scrum metodom pa od ponuđenih (Slika zaslona 6) odabiremo „Scrum“.



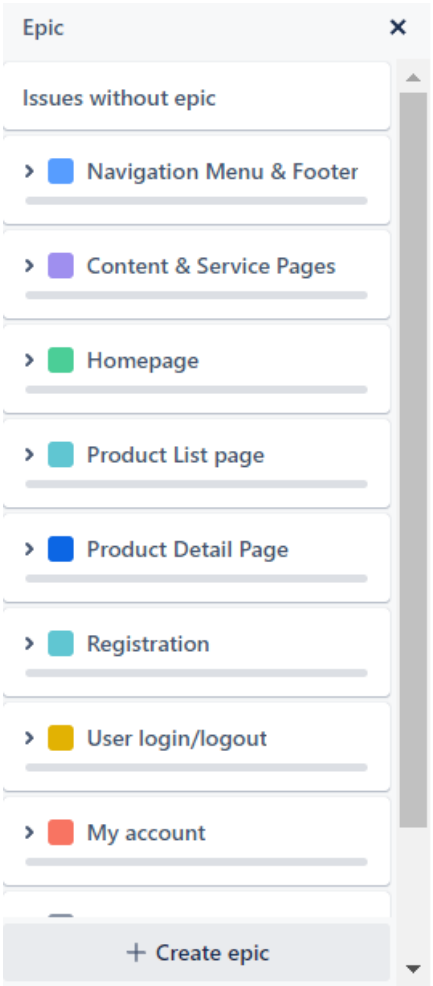
Slika zaslona 6:Odabir predloška projekta (https://avalent.atlassian.net/jira/projects)

Nakon što smo odabrali predložak, Jira nam nudi da odaberemo želimo li kreirati projekt koji je upravljan od strane poduzeća ili neke organizacije ili od strane tima (Slika zaslona 7). Odabrao sam opciju upravljanja od strane tima kako bih sam mogao urediti zadatke (eng. tickete) unutar projekta. Nakon toga nam je dano da odaberemo koji naslov želimo dati projektu te mu dajemo ime „Webshop“.



Slika zaslona 7: Odabir tipa projekta (https://avalent.atlassian.net/jira/projects)

Nakon što smo kreirali projekt, prvo što trebamo napraviti je kreirati epove (eng. Epic) koji su nam dani u zadatku (Slika zaslona 8).



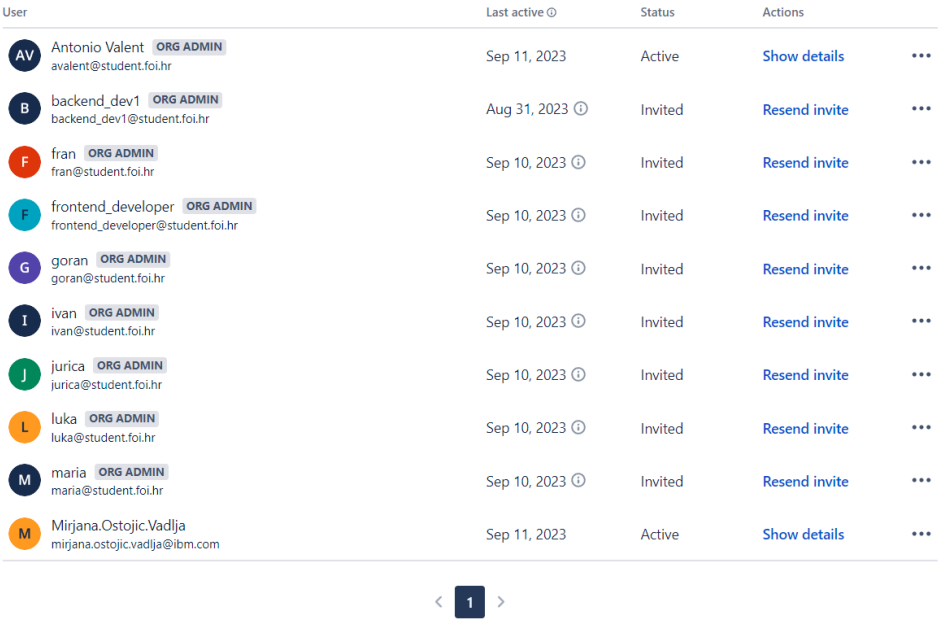
Slika zaslona 8: Dodani epovi
(https://avalent.atlassian.net/jira/software/projects/FOI/boards/2/backlog?epics=visible)

Prije nego što možemo dodijeliti neki zadatak članu tima, moramo kreirati i tim. Sukladno tome, na slici 9 vidljiv je popis članova tima. Tako imamo:

- Luka – frontend razvojni inženjer
- Goran – frontend razvojni inženjer
- Jurica – backend razvojni inženjer
- Maria – backend razvojni inženjer
- Fran – tester
- Ivan – tester

DevOps inženjer, Scrum Master i vlasnik proizvoda unutar Jire nemaju toliko dodijeljenih zadataka kao što to imaju ostali članovi razvojnog tima, ali su njihove uloge podjednako važne kroz razvoj projekta. DevOps inženjer je većinu vremena ima zadaću da postavljanja potrebna razvojna i testna okruženja (QA, Dev i UAT), zatim osigurava članovima tima da imaju pristup potrebnim aplikacijama i autentifikaciji te rad na serveru. Scrum Master vodi veliku većinu sastanaka koji su bitni za to da tim bude u toku s razvojem projekta te se

brine o njihovom zadovoljstvu i potrebama. Vlasnik proizvoda je osoba koja ima jak dobar uvidu želje i potrebe klijenata zato što je on glavna kontakt osoba te sudjeluje u izradi i postavljanja prioriteta većine zadataka na sprint ploči nakon sastanaka i dogovora s klijentom. (Ostojić Vadlja, 2023).

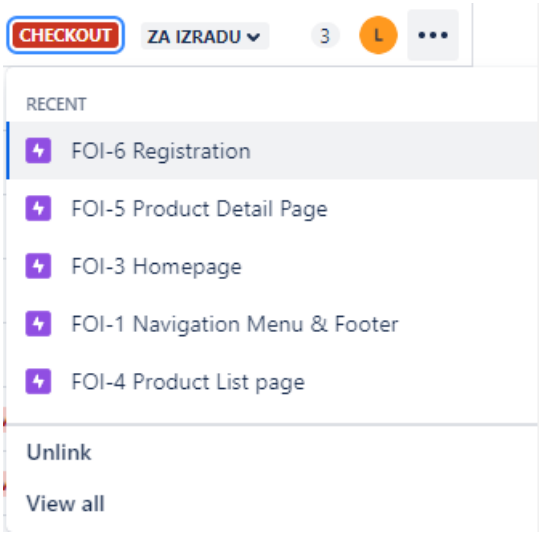


Slika zaslona 9: Popis članova tima (https://admin.atlassian.com/o/9939ck89-k18a-1a6j-ja4a-1k3bd99j4628/users?status=ACTIVE)

Sada kada imamo kreirane epove možemo početi s dodavanjem zadataka (eng. issue) za pojedine članove tima. Na slici zaslona 10 prikazan je popis funkcionalnosti sprinta (eng. backlog). Možemo kreirati koliko god zadataka smatramo da je potrebno kako bi se ispunili svi zahtjevi klijenta. Prozorčić koji je otvoren, daje nam uvid u stanje projekta te unutar popisa funkcionalnosti svi članovi tima mogu vidjeti kako napreduje koji segment projekta.

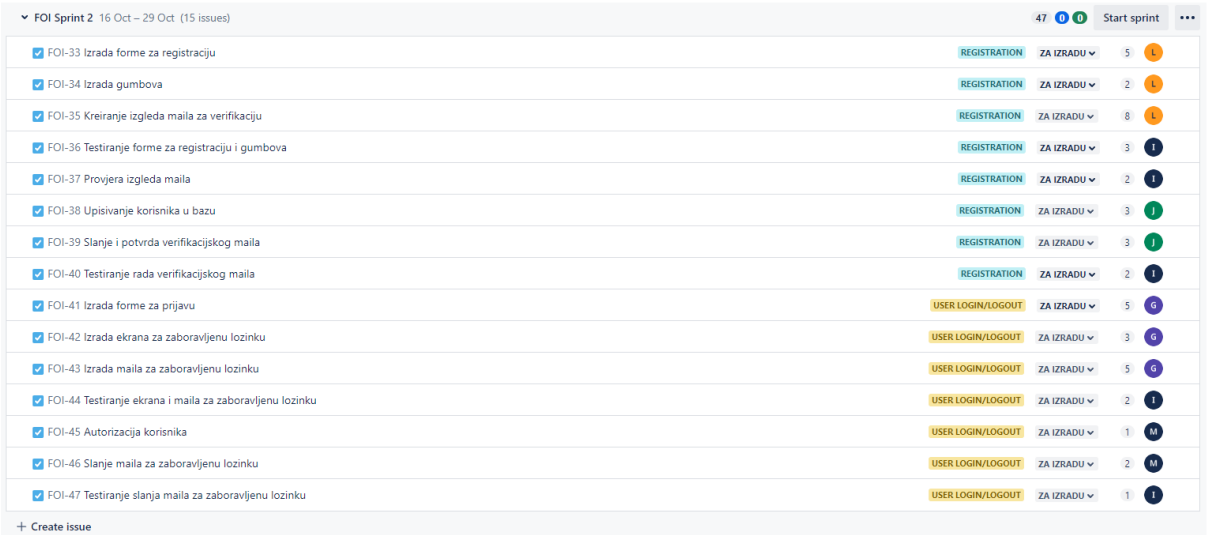
Slika zaslona 10: Prikaz popisa funkcionalnosti (https://avalent.atlassian.net/jira/software/projects/FOI/boards/2/backlog?epics=visible)

Osim što imamo uvid u stanje projekta, unutar popisa funkcionalnosti dodjeljujemo zadatke članovima tima, nekom od metoda procjene odredimo im vrijednost priče (eng. story point) te ih dodajemo u neki od kreiranih epova kojem pripadaju (slika zaslona 11).



Slika zaslona 11: Dodavanje zadatka u ep
(https://avalent.atlassian.net/jira/software/projects/FOI/boards/2/backlog?epics=visible)

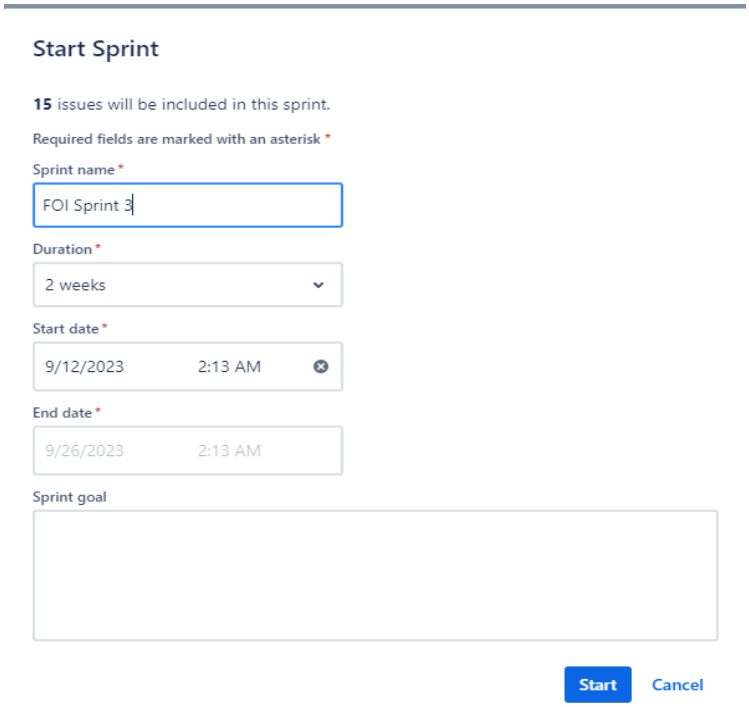
Nakon što smo dodijelili zadatke epovima kojima pripadaju, vrijeme je da izradimo sprint, a to možemo pritiskom na gumb „Create sprint“ te samo označimo zadatke koje želimo i povučemo ih preko ekrana u područje sprinta. Kada to učinimo dobijemo nešto slično kao na slici 12.



Slika zaslona 12: Prikaz sprinta
(https://avalent.atlassian.net/jira/software/projects/FOI/boards/2/backlog?epics=visible)

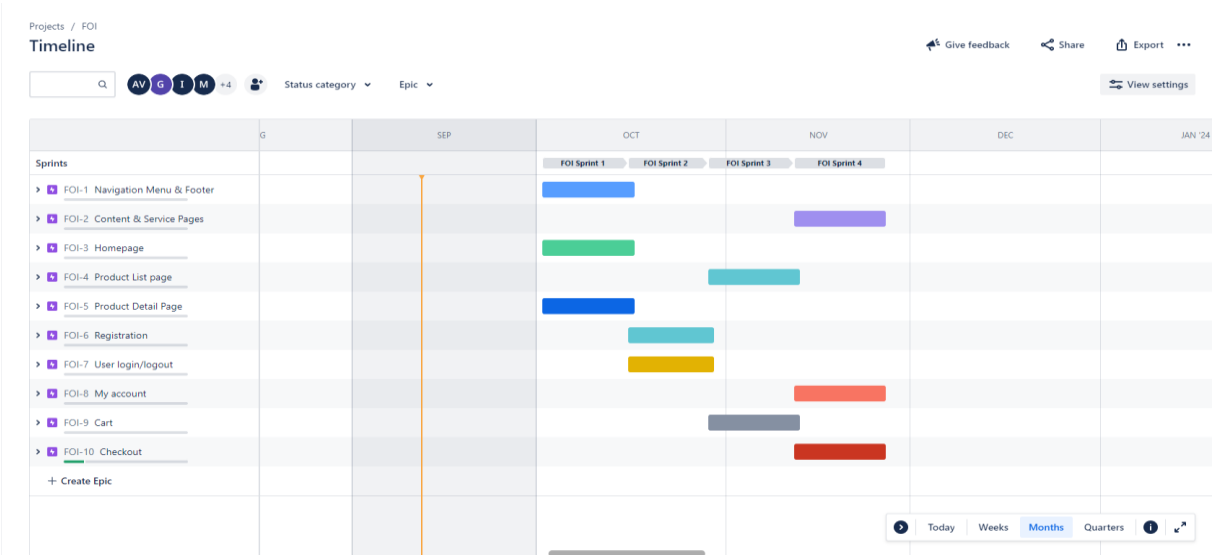
Prije nego što započnemo sprint moramo odrediti njegovo trajanje te datum početka i kraja. To možemo postići pritiskom na gumb „Start sprint“ nakon čega nam se otvara prozor

na slici zaslona 13. Dajemo mu naziv, odredimo trajanje od 2 tjedna te početni datum po želji, kako nam odgovara.



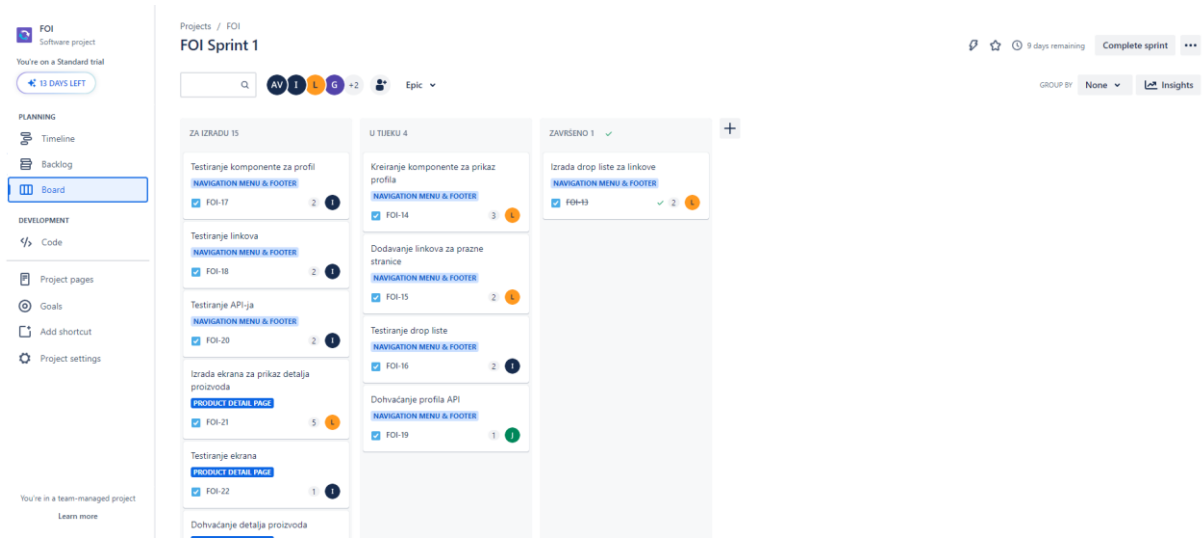
Slika zaslona 13: Prozor za početak sprinta
(https://avalent.atlassian.net/jira/software/projects/FOI/boards/2/backlog?epics=visible)

Na slici zaslona 14 vidljiv je tijek projekta pomoću vremenske crte (eng. timeline). Točno možemo vidjeti kada se počinje s izradom čega, primjerice u prvom sprintu, kreće se s izradom navigacije i podnožja, početne stranice i stranice o detaljima proizvoda. U drugom sprintu slijedu izrada registracije i prijave. U trećem sprintu kreira se stranica s popisom proizvoda te košarica i u zadnjem, četvrtom sprintu, izrađuju se stranice za račun korisnika, stranica za kupovinu te stranice sadržaja i usluga.



Slika zaslona 14: Vremenska crta projekta
(https://avalent.atlassian.net/jira/software/projects/FOI/boards/2/timeline)

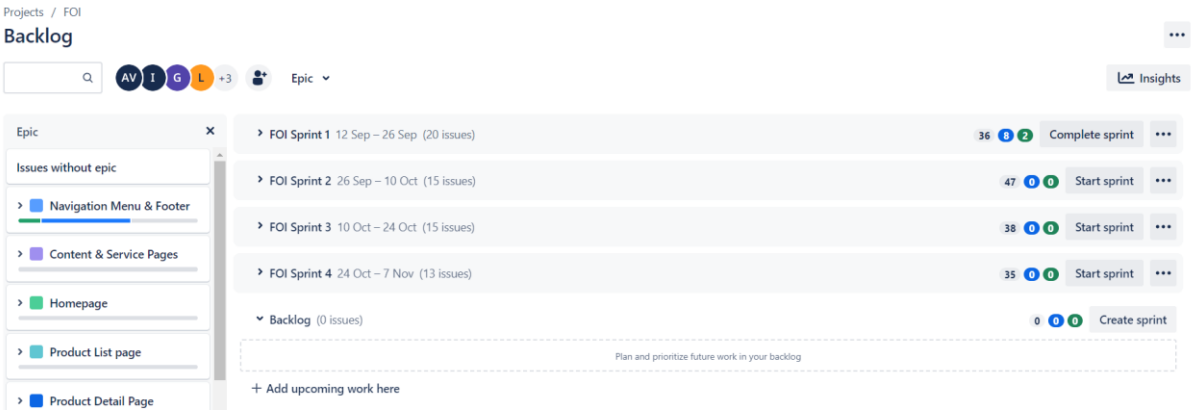
Na slici zaslona 15 prikazana je ploča zadataka (eng. board) na kojoj su vidljivi zadaci koji su na čekanju, u tijeku i koji su završeni. Prema tome, za potrebe ovog rada, smisleno su dodijeljeni statusi zadacima kako bi mogli prikazati funkcioniranje ploče te kako bi lakše vizualizirali što sve Jira nudi.



Slika zaslona 15: Prikaz ploče zadataka
(https://avalent.atlassian.net/jira/software/projects/FOI/boards/2)

Nakon što smo uspješno stvorili sve zadatke, dodijelili im vrijednost priče i člana tima koji će biti zaslužan za njegovo izvršavanje, možemo odgovoriti na drugo pitanje u zadatku od kolegice Mirjane. Prema tome, kao što možemo vidjeti na slici zaslona 16, ovaj projekt koji smo kreirali ima planirani završetak nakon 4 odrađena sprinta. Budući da je Jira veoma moćan alat, kao i sve ostalo, i rješavanje grešaka (eng. bug) nam je olakšano. Identificirati greške

može i najjeftiniji softver, no ono što Jira nudi kao vrhunski alat za agilno planiranje jest to što nam omogućuje da postavimo prioritete njihova rješavanja. Nakon što su greške identificirane na vlasniku proizvoda je da odredi važnost njihova rješavanja. Iako se popravljanje grešaka smatra da je izvan sprinta, ako članovi tima, posebice frontend i backend razvojni inženjeri, ne posvete svo svoje vrijeme ili više (100% + 50%) na projekt, već nešto manje, primjerice 50%, na taj bi način mogli osigurati da će imate vremena da ispravljanje grešaka. Također, greške bi trebalo tretirati kao i ostale zadatke na projektu te ih ubaciti u popis funkcionalnosti sprinta, kasnije i u sam sprint (Stack Overflow, 2010).

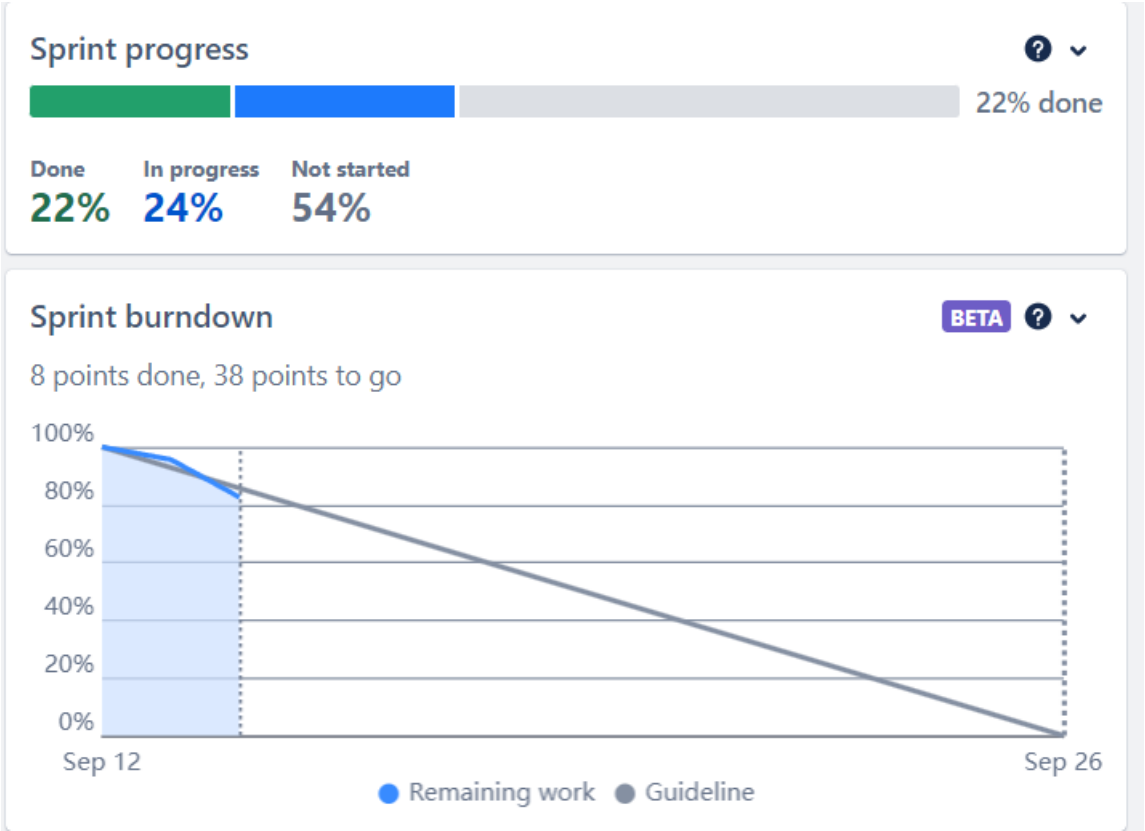


Slika zaslona 16: Prikaz popisa sprintova
(https://avalent.atlassian.net/jira/software/projects/FOI/boards/2/backlog?epics=visible)

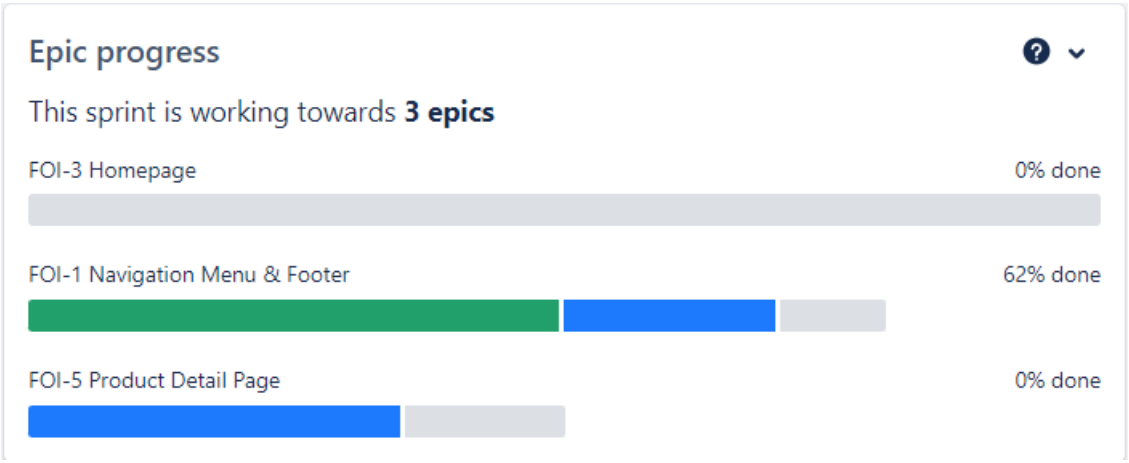
Nakon što je sprint završen provodi se replaniranje i reprocjena projekta. To nam omogućuju izvješća koja su dostupna unutar Jira alata, a to su shema sagorijevanja (eng. burndown chart) i napredak epa (eng. epic progress) (Slike zaslona 17 i 18). Tada tim može vidjeti koliko su zadataka riješili, odnosno koliko su vrijednosti priča postigli, te na temelju tih informacija i dodatnim podacima o dostupnosti članova tima mogu planirati daljnje sprintove. Sukladno tome, ako tim otkrije da su riješili premalo zadataka, tj. postigli premalo vrijednosti priča, nego što je prvotno bilo isplanirano, idući sprint bi isplanirali puno strože na način da se posvete manjem broju zadataka i vrijednosti priča te bi time bili sigurniji da će ih ispuniti u zadanom vremenu (Ostojić Vadlja, 2023).

U suprotnom, ako bi tim primijetio da su riješili sve zadatke te da su imali dio sprinta bez dovoljne količine posla da zadovolje cilj, tada je jasno da su sprint isplanirali prestrogo te da imaju prostora u idućem sprintu povećati broj vrijednosti priča po članu tima (Ostojić Vadlja, 2023).

Također, dobra je praksa ponovo evaluirati starije zadatke koji su nekoć prošli fazu usavršavanja (eng. refinement phase), ali još nisu došli na red za razvoj, kako bismo osigurali da je procjena i dalje ispravna ili je potreban ispravak procjene. Na primjer, moguće je da se dogodi da je neki zadatak procijenjen da ima 8 vrijednosti priča, ali se kroz vrijeme pokazalo da su neki dijelovi funkcionalnosti veća razvijeni te da će za taj zadatak biti potrebno manje vremena da se riješi nego što je prvotno bilo utvrđeno. Naravno, moguće je da se dogodi i suprotna situacija, u tom je slučaju također potrebno ponovo procjenjivanjem, ali na višu ocjenu. Postoje razne situacije koje mogu zadesiti tim te je iz tog razloga ključno pratiti kako tim funkcionira i kako se mijenja kroz svaki sprint te kako to pomaže u budućim planiranjima sprintova (Ostojić Vadlja, 2023).



Slika zaslona 17: Prikaz sheme sagorijevanja (https://avalent.atlassian.net/jira/software/projects/FOI/boards/2/backlog?epics=visible&isInsightsOpen=true)



Slika zaslona 18: Prikaz napretka epa (https://avalent.atlassian.net/jira/software/projects/FOI/boards/2/backlog?epics=visible&isInsightsOpen=true)

Ovime je završen praktični dio izrade projekta u Jiri, pa zaključno s time, možemo istaknuti da je korištenje agilnih metodologija u ovome alatu donijelo značajne prednosti u ubrzanju razvojnog procesa i poboljšanju timskog koordiniranja, istovremeno omogućavajući bolju prilagodljivost i efikasnost u upravljanju projektima.

10. Zaključak

U ovom radu istražili smo ključne aspekte agilnog planiranja i procjena u razvoju softvera, ističući izazove i pravilno primijenjene prakse koje prate ovu metodologiju. Agilno planiranje i procjena su se pokazali izuzetno važnim alatima za timove koji žele uspješno upravljati projektnim rizicima i promjenama. Njihova sposobnost prilagođavanja promjenljivim zahtjevima i kontinuiranog unapređenja procesa razvoja softvera čini ih neizostavnim djelićem bilo koje softverske tvrtke koja svojim projektima želi konkurirati u IT svijetu. Izazovi agilnog planiranja uključuju nedostatak komunikacije, nejednakost u razumijevanju agilnosti, promjenjivost te neusuglašenost resursa. Ključno je da timovi prepoznaju te izazove i primjenjuju prilagođene strategije kako bi uspješno mogli zaobići te prepreke te kako bi ostali na pravom putu prema uspjehu.

Procjena u okviru agilnog razvoja softvera također je od velikog značaja. Implementacija relativnih procjena i kontinuiranog testiranja omogućava timovima da bolje razumiju svoju sposobnost i bolje planiraju svoje iteracije. Međutim, timovi moraju razviti bolje razumijevanje agilnih metodika, održavati stalnu komunikaciju, postići ravnotežu promjena koje ih zadese, upravljati resursima, pažljivo planirati te moraju obratiti pažnju na čimbenike poput zavisnosti, nesigurnosti i pritiska kako bi osigurali realističke procjene.

Napredak u agilnom planiranju i procjeni zahtjeva, ne samo razumijevanje agilnih principa i alata, već i podršku rukovodstva i neprestano učenje i prilagođavanje. Iako postoje izazovi i greške na putu, agilno planiranje i procjena ostaju moćan resurs za timove u razvoju softvera koji teže bržem, efikasnijem i prilagodljivijem pristupu projektima. Kroz kontinuirano usavršavanje svojih vještina i prilagođavanje specifičnostima projekata, timovi mogu ostvariti uspjeh u ovom dinamičnom okruženju i ostvariti vrhunske rezultate u razvoju softvera.

Popis literature

1. M. C. Layton, S. J. Ostermiller, D.J. Kynaston *Agile Project Management for dummies.* Wiley. 2017.
2. G. Aslam, F. Farooq, *A comparative study on Traditional Software Development Methods and Agile Software Development Methods,* Master Thesis, lipanj 2011. (Dostupno na http://www.diva-portal.org/smash/get/diva2:423243/FULLTEXT01.pdf)
3. A. Altvater, Stackify, „What is Agile Methodology? How It Works, Best Practices, Tools“, ožujak 2023. [Na internetu]. (Dostupno na https://stackify.com/agile-methodology/)
4. Agile Alliance, [2023] „What is Agile Software Development?“ [Na internetu]. (Dostupno na https://www.agilealliance.org/agile101/)
5. Agile Alliance, [2023] „What is Extreme Programming(XP)“, [Na internetu]. (Dostupno na https://www.agilealliance.org/glossary/xp/#q=~(infinite~false~filters~(postType~(~'post~'aa_book~'aa_event_session~'aa_experience_report~'aa_glossary~'aa_research_paper~'aa_video)~tags~(~'xp))~searchTerm~'~sort~false~sortDirection~'asc~page~1))
6. Agile Alliance, [2023] “What is Scrum“ (Dostupno na https://www.agilealliance.org/glossary/scrum/#q=~(infinite~false~filters~(postType~(~'page~'post~'aa_book~'aa_event_session~'aa_experience_report~'aa_glossary~'aa_research_paper~'aa_video)~tags~(~'scrum))~searchTerm~'~sort~false~sortDirection~'asc~page~1))
7. J. Sutherland, *Scrum The Art of Doing Twice the Work in Half the Time.* New York: Crown business. 2014.
8. P. Narasimman, *Agile vs Traditional Project Management [Top Differences]*, lipanj 2023. (Dostupno na https://www.knowledgehut.com/blog/agile/agile-project-management-vs-traditional-project-management)
9. M. McCormick, „Waterfall vs. Agile Methodology“, rujan 2012. [Na internetu]. (Dostupno na http://www.mccormickpcs.com/images/Waterfall_vs_Agile_Methodology.pdf)
10. D. Milosevic, „Agilna metodologija“, 2018. [Na internetu]. (Dostupno na https://dusanmilosevic.com/agilna-metodologija/)
11. Kanbanize, [2023] „Why Agile Project Planning Is the Key to Adaptable Work Processes“, [Na internetu]. (Dostupno na https://kanbanize.com/agile/project-management/planning)
12. Sustinere d.o.o. „Što je agilno poslovanje i kako početi raditi agilno?“ siječanj 2019. [Na internetu]. (Dostupno na https://www.sustinere.hr/blog/sto-je-agilno-poslovanje-i-kako-poceti-raditi-agilno)

13. E. Markovinovic, „Agilne metode“, *Metode agilnog razvoja softvera*, str 4-6. Osijek, 2018. [Na internetu]. (Dostupno na https://mrkve.etfos.hr/pred/ozm/si/sem24.pdf)
14. M. Cohn, *Agile Estimating and Planning*, Prentice Hall, New Jersey, USA. 2005
15. S. Singh, „Top 8 Agile Estimation Techniques“, prosinac 2022. [Na internetu]. (Dostupno na https://www.netsolutions.com/insights/how-to-estimate-projects-in-agile/)
16. D. Radigan, [2023] „Story points and estimation“, [Na internetu]. (Dostupno na https://www.atlassian.com/agile/project-management/estimation)
17. L. Yu Beng, L. Wooi Khong, T. Wai Yip, T. Soo Fun, Software Development LifeCycle Agile vs Traditionaln Approches, Singapore. 2012. [Na internetu]. (Dostupno na http://ku-fpg.github.io/files/agile-traditional.pdf)
18. Eternal Sunshine of The Mind, [2013], „Traditional Methods of Software Development“ [Na internetu]. (Dostupno na: https://eternalsunshineoftheismind.wordpress.com/2013/02/04/traditional-methods-of-software-development/)
19. T. Tester, [2023] Agile M&A: Agile Project Management & Traditional Project Management: Understanding the Difference [Na internetu]. (Dostupno na https://www.mascience.com/agile-m-a-book/agile-project-management-traditional-project-management-understanding-the-difference)
20. Pm-partners admin, „The Agile Journey: A scrum overview“, lipanj 2021. [Na internetu]. (Dostupno na https://www.pm-partners.com.au/the-agile-journey-a-scrum-overview/)
21. Scrum alliance, [2023] „What Is Scrum: A Guide to the Most Popular Agile Framework“, [Na internetu]. (Dostupno na https://www.scrumalliance.org/about-scrum)
22. A. Stellman, *Head first Agile: A Brain-Friendly Guide to Agile,* str 74, 181. Sebastopol, CA: O'Reilly Media, Inc. 2017.
23. Technopedia, [2023] „Što je ekstremno programiranje“, [Na internetu]. (Dostupno na https://hr.theastrologypage.com/extreme-programming)
24. H. Frank, „Razlika između ekstremnog programiranja i SCRUM – a“, ožujak 2021. [Na internetu]. (Dostupno na https://hr.strephonsays.com/extreme-programming-and-vs-scrum-7937)
25. S. Sharma, D. Sarkar, D. Gupta, *Agile Processes and Methodologies: A Conceptual Study,* International Journal on Computer Science and Engineering (IJCSE), May 2012.
26. KK Consulting, [2021] „Kanban – efikasnije vođenje projekata“, [Na internetu]. (Dostupno na https://kanban.com.hr/)
27. Kanbanize, [2023] „What is Kanban?“, [Na internetu]. (Dostupno na https://kanbanize.com/kanban-resources/getting-started/what-is-kanban)

28. A. Tran, „Planiranje online pokera | 5 aplikacija sa savjetima za najbolju praksu u 2023“, srpanj 2023. [Na internetu]. (Dostupno na https://ahaslides.com/hr/blog/planning-poker-online/?fbclid=IwAR1wi2r2yklXnP-dinfz1TTa4ML0MjJzNXLCdl8EoRJeEgWZkl40TdiBuHw)
29. Simplilearn, [2023] „A Guide to Agile Estimation Techniques“, [Na internetu]. (Dostupno na https://www.simplilearn.com/tutorials/agile-scrum-tutorial/agile-estimation-techniques)
30. J. Grenning, [2002] „Planning Poker or How to avoid analysis paralysis while release planning“ [Na internetu]. (Dostupno na https://mail.renaissancesoftware.net/files/articles/PlanningPoker-v1.1.pdf)
31. D. Parsons, R. Thorn, m. Inkila, K. MacCallum, prosinac 2018. „Using Trello to Support Agile and Lean Learning with Scrum and Kanban in Teacher Professional Development“, 2018. IEEE International Conference on Teaching
32. Asana, [2023] „How to use T-Shirt Sizing to Estimate Projects“, [Na internetu]. (Dostupno na https://asana.com/resources/t-shirt-sizing)
33. Indeed, [2023] „10 Agile Estimation Techniques for Businesses To Use“ [Na internetu]. (Dostupno na https://www.indeed.com/career-advice/career-development/agile-estimation-techniques)
34. A. Mihalache, [2017] „Project Management Tools for Agile Teams“, [Na internetu]. (Dostupno na https://pdfs.semanticscholar.org/aaba/28b1f1070fe759b59764c7879321581ce485.pdf)
35. Kanbanize, [2023] „The Best 9 Agile Project Management Tools of 2023“, [Na internetu]. (Dostupno na https://kanbanize.com/agile/project-management/tools)
36. G. Azizyan, M. K. Magarian, M. Kajko-Mattson, [2011] „Survey of Agile Tool Usage and Needs“, [Na internetu]. (Dostupno na https://www.researchgate.net/profile/MiganoushMagarian/publication/252038812_Survey_of_Agile_Tool_Usage_and_Needs/links/58 3c57c908ae1ff45982d738/Survey-of-Agile-Tool-Usage-and-Needs.pdf)
37. M. Shaikh, A. Khan, I. Farah Siddiqui, Z. Quershi, 2021. „Comparative Analysis of Trending Agile Model Tools for Sofware Development Life Cycle“, ICONI 2021. (Dostupno na https://www.researchgate.net/profile/Mehwish-Shaikh/publication/357974756_Comparative_Analysis_of_Trending_Agile_Model_Tools_for_Software_Development_Life_Cycle/links/61e9bf3b9a753545e2e524a1/Comparative-Analysis-of-Trending-Agile-Model-Tools-for-Software-Development-Life-Cycle.pdf)

38. D. Ozkan, A. Mishra, „Agile Project Management Tools: A Brief Comparative View“, Bulgarian Academy of Science, Sofia, Bugarska, 2019.
39. Atlassian, [2023] „Trello: Bring the power of a visual perspective to your team“, [Na internetu]. (Dostupno na https://www.atlassian.com/software/trello)
40. The Business Blocks, [2023] „Confluence: A Remote-friendly Team Workspace For Better Colaboration“. [Na internetu]. (Dostupno na https://thebusinessblocks.com/knowledge-management/confluence/)
41. IMB, [2023] „What is Monte Carlo simulation“. [Na internetu] (Dostupno na https://www.ibm.com/topics/monte-carlo-simulation)
42. W. Kenton, [2023]. „Monte Carlo Simulation: History, How it Works, and 4 Key Steps“. [Na internetu]. (Dostupno na https://www.investopedia.com/terms/m/montecarlosimulation.asp)
43. Doc. Dr. sc. Mario Kušek, „Agilne metode razvoja programa“, FER, Zavod za telekomunikacije, travanj 2010. (Dostupno na https://www.aes.hr/_download/repository/AgilneMetode2010.pdf)
44. Aha! Labs Inc [2023], „Product releases vs. Sprints in scrum“ [Na internetu]. (Dostupno na https://www.aha.io/roadmapping/guide/release-management/product-manager-sprint)
45. G. Radhakrishnan, [2023], „Three Pillars of Scrum: Transparency, Inspection and Adaptation“, [Na internetu]. (Dostupno na https://www.knowledgehut.com/blog/agile/scrum-pillars#the-3-pillars%E2%80%AFof-scrum)
46. A. Dennis, B. Wixom, R. M. Roth, „System Analysis and Design 5th edition“, John Wiley & Sons, Inc, SAD, 2012. (Dostupno na http://www.uoitc.edu.iq/images/documents/informatics-institute/Competitive_exam/Systemanalysisanddesign.pdf)
47. Education-Wiki [2023], „Waterfall Model“. (Dostupno na https://hr.education-wiki.com/4611920-waterfallmodel)
48. A. Oppermann, [2023], „What is V-Model in Software Development“[Na internetu]. (Dostupno na https://builtin.com/software-engineering-perspectives/v-model)
49. A.K.M. Zahidul Islam, A. Ferworn [2020], „A Comparison between Agile and Traditionaln Software Development Methodologies“, Global Journal of Computer Science and Technology, Global Journals. (Dostupno na https://globaljournals.org/GJCST_Volume20/2-A-Comparison-between-Agile.pdf)
50. J. Strasser [2022], „Comparing Project Management Methods: Agile, Traditional, Hybrid“. (Dostupno na https://www.theprojectgroup.com/blog/en/agile-project-management-methods/)

51. M. Mrsic [2017], „Crystal Methods“. (Dostupno na https://activecollab.com/blog/project-management/crystal-methods)
52. ProductPlan [2023], „What is Jira“. (Dostupno na https://www.productplan.com/glossary/jira/)
53. D.J. Eich [2023], „Miro vs Mural: A features Comparison“. (Dostupno na https://www.innovationtraining.org/miro-vs-mural/)
54. W. Hutagalung [2006], „Extreme Programming“. (Dostupno na https://www.umsl.edu/~sauterv/analysis/f06Papers/Hutagalung/)
55. B.Lutkevich [2023], „Monte Carlo Simulation“. (Dostupno na https://www.techtarget.com/searchcloudcomputing/definition/Monte-Carlo-simulation)
56. S. Pandey [2023], „10 Common Challenges Faced in Agile Project Management“. (Dostupno na https://www.tutorialspoint.com/10-common-challenges-faced-in-agile-project-management)
57. Wrike [2023], „What is Agile Crystal Methodology“. (Dostupno na https://www.wrike.com/agile-guide/faq/what-is-agile-crystal-methodology/)
58. D. Tzemach [2022], „Challenges with Estimation in Agile project“. (Dostupno na https://www.agilequalitymadeeasy.com/post/challenges-with-estimations-in-agile-projects-david-tzemach)
59. Stack Overflow [2010], „Best ways to fit bug fixing into a Scrum project“. (Dostupno na https://stackoverflow.com/questions/1593328/best-ways-to-fit-bug-fixing-into-a-scrum-process)
60. Mirjana Ostojić Vadlja, osobna komunikacija, 4.9.2023.

Popis slika

Popis tablica i dijagrama