SCRUM metodologija

Šiljevinac, Robert

SVEUČILIŠTE JURJA DOBRILE U PULI

FAKULTET INFORMATIKE

ROBERT ŠILJEVINAC

SCRUM METODOLOGIJA

Završni rad

PULA, 2018.

SVEUČILIŠTE JURJA DOBRILE U PULI

FAKULTET INFORMATIKE

ROBERT ŠILJEVINAC

SCRUM METODOLOGIJA

Završni rad

JMBAG: 2424013635, izvanredni student
Studijski smjer: Informatika

Predmet: Projektiranje informacijskih sustava
Mentorica: Prof. Dr. sc. Vanja Bevanda

PULA, 2018

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani Robert Šiljevinac, kandidat za prvostupnika informatike ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student

_____

U Puli, 18. veljače, 2018. godine

IZJAVA

o korištenju autorskog djela

Ja, Robert Šiljevinac dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj završni rad pod nazivom „Scrum Metodologija“ koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, 18. veljače, 2018. godine.

Potpis

_____

Sadržaj

1. Uvod 1
2. Scrum Metodologija 2
2.1. Agilne metodologije razvoja informacijskih sustava 2
2.2. Povijest Scrum metodologije 3
2.3. Scrum Proces 8
2.4. Scrum Tim 10
2.4.1 Vlasnik Proizvoda 11
2.4.2 Scrum Master 13
2.4.3 Razvojni tim 15
2.5. Scrum događaji 17
2.5.1 Sprint 18
2.5.2 Planiranje Sprinta 19
2.5.3 Dnevni Scrum 21
2.5.4 Pregled Sprinta 22
2.5.5 Retrospektiva Sprinta 24
2.6. Scrum artefakti 25
2.6.1 Backlog proizvoda 25
2.6.2 Backlog sprinta 27
2.6.3 Dijagram sagorijevanja 30
3. Praktični primjer 32
3.1. Radna okolina 33
3.2. Korišteni alat 34
3.3. Backlog Projekta 37
3.4. Sprint u praktičnoj okolini 38
4. Zaključak 42
5. Literatura 44
6. Popis slika 45
7. Popis tablica 46
8. Sažetak 47
9. Summary 48

1. Uvod

Scrum metodologija razvoja informacijskih sustava predstavlja moderan način pristupu razvoja programskih rješenja. Velik utjecaj ima na stvaranje radne okoline kako bi sam process razvoja bio kvalitetniji. Ponajviše zbog naglih promjena na samom tržištu i brzoj dinamici s kojom se svakodnevno susrećemo. Koristeći specifična pravila ali istovremeno dajući programerima odriješite ruke u toku samog procesa *Scrum* predstaclja jedan vrlo prilagodljiv i učinkovit pristup razvoju proizvoda. Međutim sama metodologija se može primijeniti u bilo kojoj grani gdje se radi na proizvodnji nekog kompleksnog proizvoda, pa čak i na razne okoline gdje prevlada velika dinamika uz promjene prioriteta u kratkim vremenskim rokovima.

Većina današnjih tvrtki se mora prilagođavati tržištu i uvjetima na njemu. Konkurencija je velika, zahtjevi na samom tržištu sve su veći, a dinamika je sve brža.

Cilj ovog rada je upoznati samu metodologiju, procese koji se javljaju, pravila koja su postavljena, te ostale čimbenike koji ovaj pristup čine učinkovitim u današnje vrijeme. Nakon samog uvoda i upoznavanja *Scrum* metodologije, njenim nastankom, procesom koji kroz ponavljane akcije stvara napredak u svakoj iteraciji. Članovima koji samo organizirajući se, prateći pravila i događaje stvaraju ogromne napretke u samom pristupu radu, pa tako i na cijelom proizvodu.

Nakon upoznavanja s teorijom metodologije slijedi praktičan primjer koji cijelu metodologiju smješta u jednu specifičnu i vrlo dinamičnu okolinu. Praktični primjer prikazuje okolinu Sistemskih Administratora kompleksnih informacijskih sustava koji se svakodnevno susreću s raznim problemima. Te se oni prilagođavaju što većoj automatizaciji ponavljajućih zadataka, kako bi si olakšali svakodnevni rad a uz to održavajući i unapređujući sustav na kojem svakodnevno rade.

Zaključak na samom kraju rada donosi opis učinkovitosti metodologije u specifičnoj okolini iz samog primjera.

2. Scrum Metodologija

Za bolje razumijevanje *Scrum* metodologije i utjecaj na razvoj informacijskih sustava određenu pažnju potrebno je posvetiti Agilnim metodama razvoja, jer *Scrum* predstavlja jednu od metoda Agilnog razvoja.

2.1. Agilne metodologije razvoja informacijskih sustava

Agilni pristup razvoja informacijskih sustava predstavlja moderni pristup razvoju, koji je prilagođen današnjem tempu na tržištu i naglim promjenama koje se na njemu javljaju. Većina agilnih metoda teži tome da smanji rizik pri razvoju informacijskih sustava počevši od: prekoračenja vremenskih rokova, programskih grešaka, zastarijevanja tehnologija i slično…

Oslanjajući se na takozvane *iteracije*, to jest razvoj gotovih rješenja u kratkim vremenskim periodima. Svaka iteracija je zasebni cjeloviti projekt programske podrške koja je ujedno i manji dio cijelog informacijskog sustava, pa se iz tog razloga ne može reći da je ona ujedno i gotov proizvod. Ona predstavlja jedan sastavni dio cijelog informacijskog sustava.

Projekti koji se razvijaju nekom od agilnih metoda nakon svake iteracije nastoje proizvesti novu verziju programa. Na taj način svakom iteracijom program odnosno informacijski sustav postaje bogatiji i napredniji za određenu funkcionalnost koja se prethodno razvijala. Svaka iteracija započinje planiranjem, zatim analizom zahtjeva, dizajnom, kodiranjem, testiranjem i za kraj završava dokumentiranjem određene funkcionalnosti koja unaprjeđuje ukupni sustav.

Komunikacija u realnom vremenu ima jako veliku važnost kod agilnih metoda razvoja, koja se često odvija licem u lice, a puno je manje pažnje posvećeno na izradu dokumentacije. Velika važnost je posvećena na direktnu komunikaciju klijenta odnosno naručitelja programskog rješenja s timom programera koji izrađuje programsko rješenje za klijenta. Klijenti su najčešće direktni naručitelji, manageri, poslovni analitičari, zaposlenici drugih odjela poduzeća, odnosno netko tko ima potrebu za programskim rješenjem.

Veliki naglasak kod agilnih metoda spada na proces upravljanja koji ima funkciju da prati i nadzire, kao glavni pokazatelj napretka cijelog projekta. Poslijednje u kombinaciji s težnjom prema komunikaciji licem u lice, pokazatelj je da agilne metode imaju jako malo pisane dokumentacije u odnosu na druge metode razvoja.

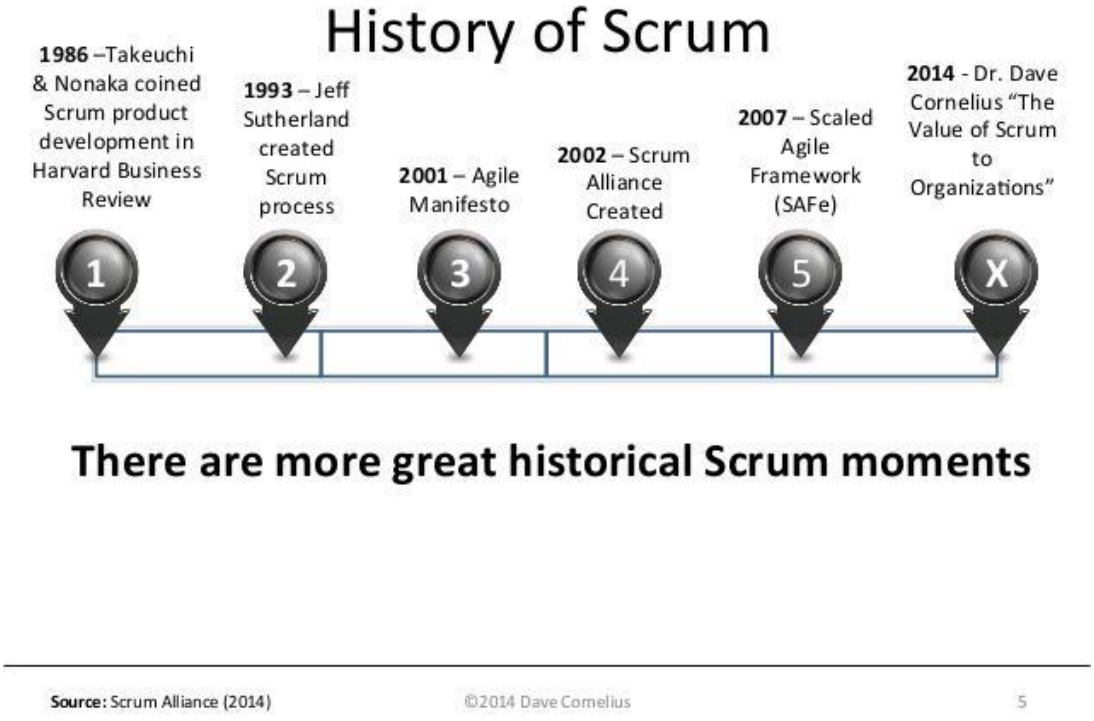
Agilne metode razvoja programskih rješenja se okarakteriziraju kao prilagodljivi tip razvoja, jer se brzo i učinkovito prilagođavaju na promjene koje se javljaju u stvarnosti vezano za projekt na kojem se primjenjuju. Tim stručnjaka koji razvoj temelji na agilnim metodama nalaze se u situaciji da znaju samo na čemu trenutno rade i koji su im ciljevi u narednom periodu od približno 14 dana.

Stručnjaci koji sudjeluju u razvoju nemaju predodžbu kakva će biti finalna slika proizvoda u nerdnim mjesecima, često zadanim rokom isporuke krajnjeg, gotovog proizvoda. Usredotočeni su na trenutne i kratkoročne ciljeve, što je i srž prilagodljivost prema okolini u razvoju programskih rješenja. Dobra učinkovitost agilnog razvoja uočena je kada broj članova unutar jednog razvojnog tima ne prelazi brojku od 10, također je bitan čimbenik da koriste jednu lokaciju na kojoj se sastaju povodom razvoja.

Projekti na koje se preporučuje agilna metoda često su projekti nepredvidljivih ili brzo mijenjajućih zahtjeva od naručitelja. Kod velikih projekta čest je slučaj postojanja većeg broja heterogenih timova, a svaki od timova radi na zasebnom dijelu projekta da bi se u konačnici sve spojilo u jednu zajedničku cjelinu.

2.2. Povijest Scrum metodologije

Sama metodologija ima nekoliko značajnih događaja koji su kroz povijest imali veliki utjecaj na razvoj metodologije, njeno stvaranje, te na koncu formiranje i unapređenje.



Slika 1: Najznačajniji povijesni Scrum događaji. Izvor: www.slideshare.net , www.scrumalliance.org

Scrum predstavlja najpopularniju agilnu metodu razvoja informacijskih sustava. Predstavili su ga ranih 90-ih godina prošlog stoljeća Jeff Stherland i Ken Schwaber. Međutim *Scrum* svoje korijene vuče još iz 70-ih godina podrijetlom iz Japana. Pod nazivom Sashimi[1] modelu kriju se korijeni današnjeg *Scrum*-a kakvog poznajemo.

„Sredinom 80-tih godina Hirotaka Takeuchi i Ilkujiro Nonaka definirali su fleksibilan i sve obuhvatnu strategiju izrade proizvoda gdje razvojni tim radi kao jedna cjelina kako bi postigli zajednički cilj. Opisali su inovaciju kao pristup za razvoj proizvoda kojeg su nazvali holistički „*rugby*“ koncept, „gdje jedan tim pokušava prijeći jednu razdaljinu kao cjelina, dodajući loptu nazad i naprijed. Bazirali su svoj koncept na proizvodnji i analizi pojedinih slučaja iz različitih industrija. Takeuchi i Nonaka su predlagali da taj pristup razvoja ne smije biti kao sekvencijalni štafeta, već bolje da bude analogan kao i igra rugby gdje cijeli tim radi zajedno, dodajući loptu nazad i naprijed dok se kreću kao jedna jedinica preko terena.“[2]

[1] Sashimi – japansko jelo od svježih morskih plodova narezanih na tanke listiće
[2] Satpathy Tridibesh, Scrum body of knowledge, SCRUMstudy™, Phoenix, Arizona, 2003.,str. 3.

Japanska tvrtka Xerox je u to doba koristila klasičan sekvencijalni vodopadni (*Eng. Waterfall*) model za izradu svojih fotokopirnih uređaja. Vodo padni model predstavlja metodu gdje se faze izrade izvršavaju točno planiranim redoslijedom i tek kada jedna faza završi kreće se u radove sa novom fazom što znatno produljuje proizvodni proces. Nakon što je korištenje gore navedenog modela razvoja dovelo do spoznaje da takav model nije adekvatan za brže i jednsotavnije isporučivanje gotovih rješenja, inženjeri su došli do ijdeje da preklapaju faze izrade, tj. da se više faza odvija istovremeno. To je zahtjevalo potrebu za puno većom komunikacijom i interakcijom između odjela (istraživački, razvojni, testni, prodajni…), a to je ujedno i jedna od najznačajnijih osobina agilnih metodologija.

Ovakav pristup na koncu doveo je do pozitvinih rezultata u samoj proizvodnji: kraći rokovi trajanja, veća fleksibilnost prema promjenama, promocija odgovornosti i suradnje, razmjena informacija ključnih za proizvod između različitih timova koji rade na istom proizvodu, do u konačnici kvalitetnijeg proizvoda. Model Sashimi je u kratkom vremenu evolvirao na način da umjesto preklapanja susjednih faza razvoja preklapa više faza istovremeno, a članovi tima koji sudjeluju u razvoju na različitim fazama zajednički dolaze do cilja. Model je postao dominantan, pa nije trebalo dugo da i druge Japanske tvrtke preuzmu istu metodologiju razvoja (npr. Honda i Canon).

Agile Manefisto

Agilne metode razvoja predstavljaju skup metoda kojima se razvijaju programske podrške. Od 11. do 13. veljače 2001. godine se sastalo sedamnaest ličnosti na jednom skijalištu imena Snowbird na planini Wasatch u blizini američkog grada Utah. Ti ljudi bili su poznati u svijetu po različitim metodama pristupa agilnog razvoja programskih rješenja. Kao što su npr. predstavnici Extreme Programming, SCRUM, DSDM, Adaptive Software Development, Crystal, Feature-Driven Development, Pragmatic Programming i drugi. Dugo su raspravljali kako bi se usuglasili, oko osnovnih načela za sve gore navedene metode te stvorili tako zvani „*Agile Manifesto*“, koji danas predstavlja osnovnu definiciju agilnog pristupa razvoja informacijskih sustava i programskih podrška. Ljudi nazočni na tom sastanku bili su: Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland i Dave Thomas.

Osnovna načela na kojim se temelji Agile Manifesto:

- Zadovoljstvo naručitelja je prioritet koji se postiže vrlo ranim i neprekidnim dostavljanjem programskih rješenja koji imaju svoju vrijednost
- Promjene na tržištu ili promjene zahtjeva naručitelja u bilo kojem trenutku ne predstavljaju problem u razvoju i uvijek im se može prilagoditi
- Isporuka upotrebljivih gotovih programskih rješenja je u što kraćim vremenskim periodima, od nekoliko tjedana do nekoliko mjeseci
- Tokom cijelog razvoja projekta razvojni inženjeri i poslovni ljudi moraju svakodnevno zajedno raditi
- Motivacija pojedinaca je osnova ostvarivanja projekta, koja se osigurava na način da je osigurano potrebno okruženje i njima potrebna podrška, te im se posao prepušta s povjerenjem
- Razgovori koji se vode licem u lice najučinkovitiji su način prijenosa informaciju prema razvojnom timu i unutar tima.[3]

Promatrajući načela koja su definirana unutar „*Agile Manifesto*“ možemo primijetiti da je cijela metodologija orijentirana na zadovoljstvo svih sudionika unutar razvoja programskih rješenja i prateći ta načela razvijaju se kvalitetni proizvodi. Međutim da li je tako nešto u praksi uopće izvedivo te kolika je učinkovitost takvog pristupa. Jer sigurno je jako teško pratiti sva definirana načela.

Scrum Alliance je osnovan je 2001. godine te je najveća, najcjenjenija i najutjecajnija organizacija za članstvo i certificiranje u agilnoj zajednici.

Njihova je vizija "Preobraziti svijet rada", s zadatkom usmjeravanja i poticanja pojedinaca, vođa i organizacija s praksama, načelima i vrijednostima koji stvaraju radna mjesta koja su radosna, prosperitetna i održiva.[4]

SAFe – *Scaled Agile Framework* je slobodno otkrivena on-line baza znanja dokazanih uzoraka uspjeha. Za ljude koji grade najvažnije svjetske programe i sustave. Pomaže tvrtkama u rješavanju problema tokom razvoja i isporuke programskih rješenja kako bi u najkraćem roku bili produktivni.[5]

„Prema *rugby* konceptu „*Scrum*“ (gdje se jedna skupina igrača okuplja za ponovno

[3] Agile Manefisto, http://agilemanifesto.org/. 18.12.2017.
[4] About Scrum Alliance https://www.scrumalliance.org/about-us 20.12.2017.
[5] SAFe http://www.scaledagileframework.com/what-is-safe/ 15.01.2018.

pokretanje igre) je predstavljen unutar proizvoda kako bi opisao autorov prijedlog da razvoj proizvoda bi trebao imati uključen „kretanje *Scrum*-om niz teren“.“[6] Naziv *Scrum* svoje porijeklo vuče iz *rugby*-ja, te predstavlja trenutak u toku utakmice kada se lopta vraća u igru a igrači obiju momčadi se strateški postavljaju na gomilu u borbu za posjed lopte. Sam *scrum* je orijentiran na funkcionalnost tima da bi ostvarili fleksibilnost sustava u okolini koji se neprestano mijenja.

Budući da *scrum* uključuje upravljačke aktivnosti koje sustavno uočavaju nedostatke u razvoju automatski poboljšava postojeći inženjerski postupak. *Scrum* se često naziva metodologijom, međutim on to nije. *Scrum* je okvir (*Eng. Framework*), metodologije razvojnog procesa koji služi za upravljanje razvojnim procesom. Uporabom *Scrum*-a mi ne definiramo detalje procesa, već razvojni tim ljudi stvara proces prilagođen sebi i uvjetima. *Scrum* je razvijen kroz primjenu empirijskih i teorijskih načela kontrole procesa razvoja sustava moderne industrije, čime je omogućio upravljanje i reguliranje procesima razvoja informacijskih sustava.

Osnovni temelji empirijske kontrole procesa čine transparentnost, kontrola i prilagodba. Korisnici *Scrum*-a moraju često kontrolirati artefakte, te napredak prema cilju kako bi lakše uočavali moguće pojave neželjenih odstupanja čime postižu temeljnu kontrolu. Prilagodljivost je uočljiva unutar korekcija proizvoda ukoliko su potrebne kako bi smanjili i minimizirali moguća odstupanja. Artefakti predstavljaju listu zadataka koje je potrebno implementirati na proizvodu. Komunikacija na redovitim sastancima koji su svakodnevni, rasprave o tekućim problemima, te obećanja što će se odraditi do sljedećeg sastanka čine srž stojećih i kratkih sastanka među članovima tima. Čime se stvara povjerenje između članova tima (poslovni analitičari, testeri, programeri, dizajneri…) Rezultat toga su zadovoljni članovi, što osigurava brži i kvalitetniji razvoj, čime se ujedno zadovoljava naručitelja i korisnika programskog proizvoda.

Ove karakteristike čine *Scrum* vrlo jednostavnim, lako razumljivim i vrlo čvrstim razvojnim procesom koji je usredotočen na bitne funkcionalnosti koje razvojni tim sam procijeni da može implementirati.

[6] Prijevod iz: Satpathy Tridibesh, Scrum body of knowledge, SCRUMstudy™, Phoenix, Arizona, 2003.,str. 3.

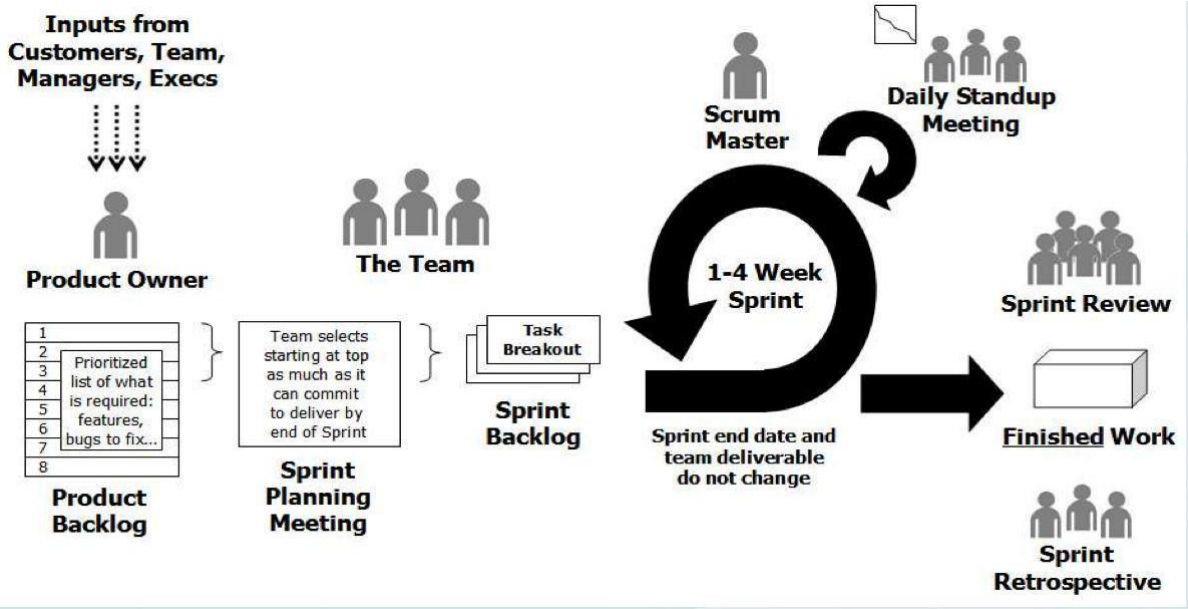
2.3. Scrum Proces

Scrum proces predstavlja skup radnji koji se odvijaju određenim redom, odnosno slijedom kako bi se proizveo jedan inkrement samog proizvoda. Od samog početka ideje proizvoda pa do konačnog gotovog proizvoda koji je spreman za isporuku.

Elementi koji čine *Scrum* proces su:

- *Scrum* tim
- *Scrum* događaji
- *Scrum* Artefakti

Svaki od tih elemenata je detaljnije opisan u daljnjem tekstu. „Svi ovi elementi se drže u jednoj cjelini uz pomoć **pravila**, koje najvećim dijelom *Scrum* tim sam postavlja i prema kojima se svi sudionici moraju jednako držati.“[7]



Slika 2: Scrum Proces. Izvor: https://www.linkedin.com/pulse/20141021025927-21583419-mobile-development-using-agile-scrum/

Sa slike je vidljivo da cijeli proces započinje samom idejom o nekom proizvodu. Ideja se prenosi na Product Owner-a[8] koji sastavlja Product Backlog[9] odnosno popis svih

[7] Scrum Proces – prijevod iz: Joachim Goll und Daniel Hommel, Mit Scrum zum gewünschten System, Springer Vieweg, Wiesbaden, Njemačka, 2015, str. 87

[8] Product Owner, Eng. Vlasnik Proizvoda

[9] Product Backlog, Eng. pozadinska lista proizvoda. Koja u našem slučaju predstavlja listu

elemenata koje taj proizvod treba sadržavati kako bi u konačnici predstavljao funkcionalnu i smislenu cjelinu da zadovolji sve potrebe naručitelja odnosno korisnika. Nakon izrade samog Backloga (koji se s vremenom nadopunjava, mijenja i prilagođava) Product Owner sa *Scrum* timom planira sprint. „**Sprint** predstavlja ciklus prema kojem se vrši razvoj. U tom vremenskom periodu *Scrum* team/razvojni tim odrađuje razvoj određenih funkcionalnosti sa Product Backloga u jednu cjelinu koja je potencijalno isporučiva i predstavlja dio proizvoda.“[10] Iz samog planiranja sprinta nastaje Sprint Backlog koji predstavlja zadatke koji su prema prioritetima izabrani i trebaju biti izvršeni u sljedećem razvojnom ciklusu odnosno sprintu. Kada sprint započne razvojni tim kreće sa izvršavanjem planiranog i predviđenog posla. Veliku ulogu ovdje ima *Scrum Master* koji pomaže timu u cijelom razvoju. Te sa njima vodi svakodnevne sastanke na kojima raspravljaju o samom razvoju i sa kojim problemima se susreću kako bi si međusobno pomogli u izvršenju cilja tog sprinta. Po završetku sprinta očekuje se da su svi zadaci/funkcionalnosti sa Sprint Backloga razvijene, testirane, dokumentirane i spremne za isporuku. Na samom sprint review-u se radi očitanje izvršenog posla. Događa se da sam tim nije bio u mogućnosti izvršiti određene zadatke koje su predviđene zbog nekih vanjskih utjecaj na koje nisu mogli utjecati, te se takvi zadaci vraćaju nazad na Product Backlog. Sprint Retrospektiva predstavlja događaj gdje se tim fokusira na sam ciklus koji je iza njih te analiziraju cijeli vremenski period te pokuašavaju pronaći odgovore postavljajući si pitanja:

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

Takvim pristupom i analizom stvaraju sebi okolinu koja im odgovara kako bi već sljedeća iteracija sprinta bila kvalitetnija i produktivnija. Te sami sebe unaprjeđuju kao tim i prilagođavaju okolini unutar koje rade na razvoju proizvoda.

Sprintevi na jednom proizvodu se ponavljaju dok svi zadaci sa Product Backloga nisu izvršeni i kompletirani. Kompletiranjem svih zadataka na kraju je vidljiv potpuno

funkcionalnosti koje neki proizvod treba imati

[10] Sprint – prijevod iz: Joachim Goll und Daniel Hommel, Mit Scrum zum gewünschten System, Springer Vieweg, Wiesbaden, Njemačka, 2015, str. 87

funkcionalni, testirani i dokumentirani proizvod koji je određene funkcionalnosti stjecao kroz cijeli proizvodni proces.

2.4. Scrum Tim

Za razliku od klasičnih metoda upravljanja projektima, *Scrum* ne posjeduje i nema potrebu za voditeljem projekta, projektnim menadžerom ili timskim voditeljem.

Scrum tim se sastoji od tri glavne uloge:

- Vlasnik proizvoda
- *Scrum Master*
- Razvojni tim

Slika 3: Tri osnovne uloge u Scrum-u. Izvor: www.smartsheet.com.

Ove tri osnovne uloge su ravnopravne između sebe i imaju svaka svoju odgovornost. Vlasnik proizvoda je zadužen za viziju projekta, određivanje prioriteta te zahtjeva, kontrolu nad troškovima te isplativosti ulaganja u sam proizvod. Vrlo je bitno da Vlasnik proizvoda precizno i temeljito prenosi informacije prema *Scrum* Masteru i *Scrum* timu sa što jasnijim pogledom i predodžbom prema samom proizvodu. *Scrum* Master rješava razne problematike koje se pojavljuju tokom razvoja, nosi odgovornost o pravilnom provođenju metodologije i poštivanju pravila iste. Također upućuje cijeli tim u daljnji razvoj. *Scrum* tim predstavlja samo organizirajuću jedinicu koja je odgovorna za razvoj i kvalitetu proizvoda. Osim ove tri glavne uloge postoje još promatrači ili savjetnici odnosno *Stakeholder*-i[11] koji prate razvoj projekta.

2.4.1 Vlasnik Proizvoda

Jedna od najvažnijih uloga, kako bi *Scrum* metodologija bila uspješna u realizaciji i stvaranju projekta, čini vlasnik proizvoda, koji čini sučelje između tima i drugih zainteresiranih strana odnosno *Stakeholder*-a. Može se reći da u tvrtkama koje koriste *Scrum* metodologiju zadaci i odgovornost koju ima vlasnik proizvoda nikada nisu isti.

Ulogu vlasnika proizvoda preuzimaju specifične osobe sa određenim vještinama koje moraju proći i određeni trening, kako bi preuzeli veliku odgovornost na sebe. Uloga vlasnika tima je ujedno i najkompleksnija uloga u izvedbi te procedure.

Vlasnik proizvoda se često nalazi u „borbama“ na dvije strane. Dok tim u određenom vremenskom periodu radi na projektu zaštićeni od *Scrum Master*-a. Vlasnik projekta se bavi marketingom, menadžmentom ili korisnicima od kojih izvlači razne informacije (pogled korisnika na aplikativno rješenje te njegove potrebe) koje su mu potrebne kako bi se mogli stvoriti preduvjeti za razvoj software-a koje zatim mora što preciznije prenijeti timu. Vlasnik proizvoda također je i odgovoran o povratu uloženih sredstava. Zatim potvrđuje gotova rješenje, provjerava kvalitetu te dali je prihvatljivo programsko rješenje za krajnjeg korisnika ovisno o točki gledišta.

Također odlučuje o važnosti pojedinog svojstva u odnosu na prioritete kako bi što bolje tim shvatio kako bi proizvod trebao izgledati te koja je konačna vizija krajnjeg projekta. Budući da tim mora što učinkovitije raditi, to vlasnik proizvoda mora brzo reagirati sa

[11] Stakeholder – zainteresirana strana, dionik, nositelj interesa, vlasnik udjela, udioničar, udjelničar itd.

povratnim informacijama. Stoga on ispunjava ulogu komunikatora, te je stalno u kontaktu sa svim *Stakeholder*-ima odnosno zainteresiranim stranama, sponzorima te za kraj i sa projektnim timom. Uostalom to i je njegov zadatak da koordinira financijsku stranu razvoja proizvoda, što uspješno provodi kroz kontinuiran rad i stvaranjem prioriteta kroz određene radnje.

Svi ti raznovrsni zahtjevi pokazuju koliko ja važan odabir odnosno izbor prave osobe za uspješnost projekta. Vlasnik proizvoda unutar *Scrum*-a nije samo menadžer, već je i vlasnik zbog čega je on direktno odgovoran za cijeli proizvod.

Slika 4: Osobnosti Vlasnika proizvoda. Izvor: www.quickscrum.com

Biti vlasnik proizvoda znači:

- Preuzeti odgovornost za uspjeh proizvoda koji mu dostavlja razvojni tim
- Donositi poslovne odluke velikih važnosti prema prioritetima
- Voditi projekt u pravom smjeru
- Izvrstan timski igrač
- Prenijeti viziju projekta prema Razvojnom timu (*Eng. Scrum team*)
- Brine o potrebama korisnika i poslovnim ciljevima
- Vlasnik proizvoda mora posjedovati znanja iz različitih domena
- Kontrolira rezultate i provjerava kvalitetu proizvoda
- Brzo reagira sa povratnim informacijama
- Komunicira na kontinuiranoj osnovi sa svim zainteresiranim stranama,

financijerima te timom

- Također kontrolira i cijelu financijsku stranu proizvoda odnosno projekta.
- Ima važnu ulogu u izradi te održavanju liste „*Product Backlog*-a“ (vidi str. 24.), te se brine da svi koji sudjeluju u projektu i razumiju radne zadatke naznačene na njemu.
- Njegova je konačna odluka o redoslijedu zadataka razvrstanih po prioritetima iz *Product Backlog* liste, te procjenjuje koliko je napora potrebno uložiti za izvršavanje pojedinih zadataka sa liste kako bi se zadatci mogli programski implementirati od razvojnog tima u određenom vremenskom periodu.

2.4.2 Scrum Master

Već samo ime nam govori kako *Scrum Master* zapravo nije voditelj tima. Voditelj tima najčešće predstavlja osobu koja vodi tim i postavlja zadatke, dok *Scrum Master* promatra da tim izvršava zadatke prema *Scrum* metodologiji.

On se ne miješa u razvoj proizvoda, već sudjeluje kao savjetnik *Scrum* timu. Aktivno se uključuje ako netko od članova tima ili od zainteresiranih strana (*Eng. Stakeholder*) ne poštuje u cijelosti pravila *Scrum*-a. Dok voditelji timova često dodjeljuje zadatke i snosi odgovornost za njihovo provođenje, iskusni *Scrum* master unutar cijelog procesa daje impulse i savijete timu kako bi ih usmjerio na pravi put da bi u konačnici koristili najbolje metode i tehnologije za dobrobit proizvoda i projekta.

Njegovo djelovanje na tim je više usmjereno prema savjetniku tima nego voditelju tima. Glavni zadatak mu je da vodi tim kroz *Scrum* process te pomaže općenito razvojnom timu da u globalu stvore uspješnu dinamiku napretka. Također organizira i moderira sastanke, kako bi mogao utjecati na samu motivaciju i produktivnost svih članova tima.

Slika 5: Scrum Master. Izvor: http://amitsinghmalik.blogspot.hr/2013/06/scrum-masterrolesresponsibilities.html.

Slika 5. prikazuje koje poslove i vještine *Scrum Master* mora imati:

- Izvrstan voditelj
- Nadgleda i prati posao
- Izvještava i komunicira sa svim članovima
- Izvrstan poznavatelj procesa
- Kontrolira kvalitetu
- Rješava raznovrsne prepreke
- Rješava sukobe ako se pojave
- Štiti razvojni tim
- Daje povratne informacije o napretku

Vrlo važan zadatak koji obavlja jest da oslobodi cijeli tim od bilo kakvih prepreka koje bi mogle poremetiti njihov nesmetani rad. Obično se problemi mogu svrstati u tri različite kategorije. Prvi problem sa kojim se tim npr. može suočiti jest nedostupan ili ne spreman Hardware za bilo kakva izvođenja ili testiranja proizvoda, IT odjel ne može

omogućiti *Bug tracker*[12], ili naručeni *software* još uvijek nije stigao do razvojnog tima. Još jedna od mogućih prepreka npr. jest pojavljivanje marketinškog ili prodajnog menadžera koji opet traži da se još jedno značajno svojstvo integrira na brzinu u proizvod. Drugi problem nastaje kao rezultat loše organizacijske strukture ili unutar loših strateških odluka.

Takve situacije znaju nastati kada tim nije u mogućnosti održavati važne sastanke ili postoje trzavice u odnosu između pojedinih članova tima, pa smatraju da je *Scrum Master* odgovoran za pojedine članove razvojnog tima. Učestalo se javljaju takvi problemi zbog krivog shvaćanja da je *Scrum Master* klasičan voditelj tima, što dovodi do sukoba interesa što je protiv glavnog principa *Scrum*-a.

Razvojni tim posjeduje menadžersku ulogu unutar *Scrum* metodologije te je ravnopravan prema *Scrum Master*-u i Vlasniku proizvoda. A drugi aspekt kao problem može predstavljati nedovoljna propusnost interneta za novi projekt. Treći problem se pojavljuju individualno izazvani iz različitih primjera. Nekome je potrebna pomoć oko *debugging*-a[13]. Netko nije u mogućnosti odraditi određene zadatke sam pa mu je potreban još jedan član za programiranje. Također i čisti banalni primjer gdje netko drugi ne vezan za tim mora resetirati određeni server zbog značajnih promjena na istome. Čak i u situacijama gdje *Scrum Master* nije u mogućnosti samostalno riješiti određene zahtjeve, i dalje je on odgovorna osoba koja se brine o tome da razvojni tim ima određene kriterije za nesmetani rad.

Takav posao često iziskuje jako puno vremena, veliki autoritet i jaku kralježnicu. Na njemu je da omogući optimalne radne uvijete za razvojni tim, te ostane odgovoran na održavanju tih uvjeta kako bi se svi ciljevi unutar sprinta ostvarili.

2.4.3 Razvojni tim

Za razliku od drugih razvojnih metodologija unutar *Scrum*-a, razvojni tim nije samo izvršni organ koji prima svoje zadatke od voditelja projekta, već samostalno odlučuju koje prohtjeve odnosno nadogradnje potrebne krajnjim korisnicima odrađuju unutar

[12] Bug tracker – aplikacija koja prati te stvara izvješća o softwareskim greškama u razvojnim projektima

[13] Debugging – je proces pronalaženja i rješavanja grešaka ili nedostataka koji sprječavaju ispravan rad računalnog softvera ili sustava

jednog sprinta. Razvojni tim izgrađuje zadatke te je odgovoran za njihovu permutaciju. Na taj način razvojni tim djeluje kao menadžer.

Takav moderni koncept samostalnog djelovanja razvojnog tima koji imaju poredanu listu zadataka i odgovornost u potpunosti mijenja ulogu voditelja tima odnosno projektnog menadžera. *Scrum Master* ne mora delegirati nad razvojnim timom kako da izrađuju proizvod, on im osigurava optimalne uvijete za samostalni rad te ih savjetuje pazeći da se ne krše *Scrum* pravila. Promijenjena percepcija uloga je najbitniji aspekt ovakve metodologije kada netko želi shvatiti *Scrum* kako bi ga uveo u tvrtku koja se bavi razvojem. Razvojni tim se u idealnim uvjetima sastoji od 7 ± 2 članova.

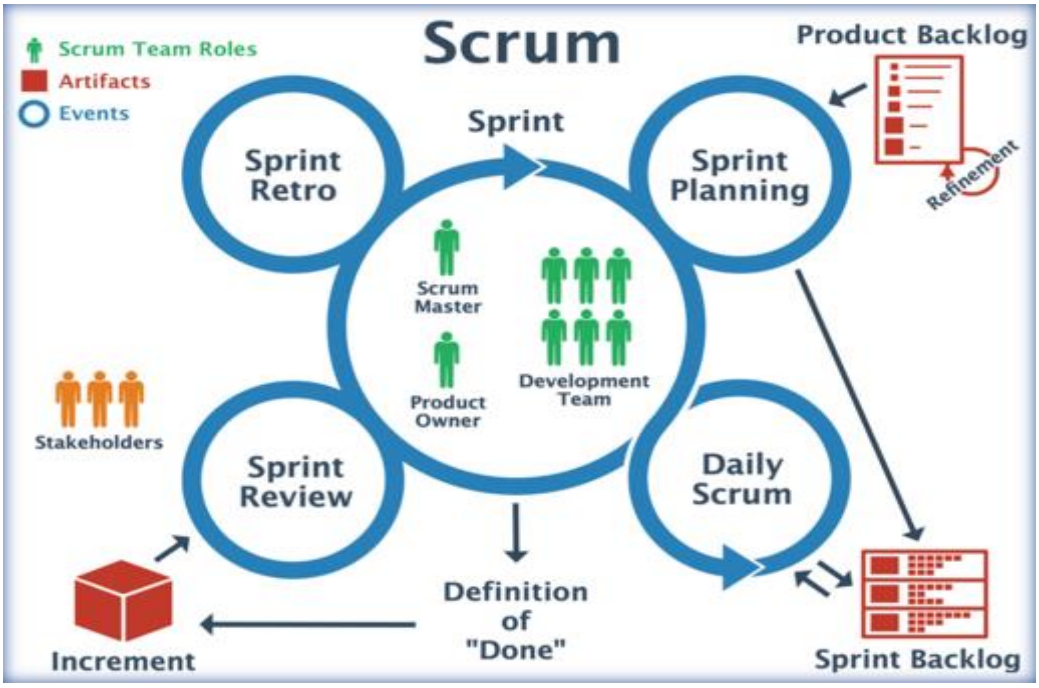
Broj članova može biti i troje, ali nikako ispod tri člana jer postoji velika mogućnost da nemaju dovoljno znanja ili vještina kako bi mogli na vrijeme do kraja sprinta izvršiti svoj posao. U slučaju da je preko devet članova onda postoji vrlo velika mogućnost da se pojavi problem oko koordinacije. Veliki broj članova unutar tima izaziva veliku složenost za empirijski proces upravljanja. U ove brojke nisu uključeni Vlasnik proizvoda niti *Scrum Master*.

Članovi tima posjeduju različite vještine te se međusobno podupiru kako nitko od njih ne bi predstavljao usko grlo tokom razvoja proizvoda. Najčešći tipičan razvojni tim uključuje mješavinu softverskih inženjera, arhitekata, programera, analitičara, eksperta za osiguranje kvalitete, programski testeri, i dizajneri grafičkog sučelja. Vrlo čvrsti *Scrum* timovi pristupaju svakom proizvodu sa čvrstim stavom „mi“. Između njih mora vladati vrlo jaki kolegijalni odnos i kvalitetan timski rad kako bi bili što više usredotočeni na proizvod koji tokom jednog sprinta moraju odraditi.

Kao što smo već spomenuli, *Scrum* tim sam upravlja planom za svaki sprint. Oni predviđaju koliko posla imaju te koliko im je vremena potrebno za određeni razvoj unutar jednog sprinta, koristeći se iskustvom iz vlastite povijesti u prethodnim sprintevima. Zadržavajući iteracije sprinteva na jednakim duljinama razvojni tim dobiva vrlo korisne povratne informacije o tome koliko vremena moraju potrošiti za razvoj određenih funkcionalnosti proizvoda. Na taj način oni s vremenom sve preciznije mogu prognozirati potrebno vrijeme i količinu posla koji moraju odraditi u toku sprinta.

2.5. Scrum događaji

Događaji koji su propisani u *Scrum* metodologiji imaju svrhu da reguliraju cijeli proces i čine ga što pravilnijim, također i smanjuju na minimum potrebu za drugim sastancima koji nisu definirani unutar metodologije. Svi događaji su vremenski uokvireni, tako da svaki događaj ima i maksimalno trajanje. Npr. nakon što je sprint definiran i počne, njegovo trajanje je fiksno te nije moguće više skratiti ili produljiti njegovo vrijeme trajanja. Ostali događaji mogu završiti kad god se postigne svrha tog događaja, što se osigurava na način da se stavi na raspolaganje dovoljno vremena a da to vrijeme nije uzaludno utrošeno.



Slika 6: Scrum događaji. Izvor: https://www.linkedin.com/pulse/20141021025927-21583419-mobile-development-using-agile-scrum/

Osim samog sprinta koji je ujedno i spremnik za sve ostale događaje unutar *Scrum* metodologije, ostali događaji predstavljaju „formalne prilike“ za pregled i adaptaciju daljnjeg razvoja proizvoda. Kao što je sa slike 6. vidljivo osim samog sprinta postoje još sljedeći događaji:

- Planiranje sprinta
- Dnevni *Scrum*
- Pregled sprinta

- Retrospektiva sprinta

Ti događaji su specifično razvijeni kako bi omogućili transparentnost i nadzor nad razvojem. Izostavljanje bilo kojeg od tih događaja rezultira s gubitkom transparentnosti te gubimo i mogućnost da se pregleda dosadašnji rad, utvrde nedostaci te adaptira ostatak radnji kako bi bio što efektivniji daljnji razvoj.

2.5.1 Sprint

Srž *Scrum* metodologije predstavlja Sprint. Sprint je iteracija odnosno vremenski okvir koji traje najduže dana, te je predviđen da unutar njega bude u potpunosti razvijen, upotrebljiv te spreman za isporuku, jedan inkrement (vidi str. 17.) odnosno jedna cjelina proizvoda. Svi sprintovi u toku razvoja proizvoda sadrže jednaku duljinu trajanja radi što učinkovitijeg mjerenja obujma posla. Novi sprint započinje neposredno zaključivanjem prethodnoga.

Sprint se sastoji od:

- Sastanci za planiranje sprinta
- Dnevni *Scrum*
- Vrijeme provedeno za razvoj od strane razvojnog tima
- Revizija sprinta
- Retrospektiva sprinta

Za vrijeme sprinta nisu dozvoljene promjene koje bi na bilo koji način mogle ugroziti uspješno završavanje cilja sprinta. Također je zabranjeno smanjivanje kvalitete predviđenog cilja. Jedine promjene koje su moguće u toku sprinta se odnose na poboljšanje kvalitete zadanog cilja u komunikaciji između vlasnika proizvoda i razvojnog tima. Za svaki sprint se unaprijed definira što se točno izrađuje te na koji način odnosno s kojim tehnologijama. Prolaskom kroz sprint vide se sve faze razvoja koje se ponavljaju svaki put kada novi sprint opet započinje.

Faze sprinta su: planiranje, razvoj (programiranje), testiranje i isporuka jedne cjeline proizvoda odnosno inkrementa. Kako je cijeli razvojni proces podijeljen na sprintove smanjuje se rizik od isporuke lošeg proizvoda, svaka razvijena cjelina uvijek je prilagođena trenutnoj situaciji na tržištu te se ujedno i lakše prilagođava potrebama tržišta koje su danas vrlo dinamične. Također je lakša prilagodba zahtjevima krajnjim

korisnicima, kojih uvijek ima i neizbježni su. Sprintevi su dodatno podijeljeni na još manje cjeline koje traju točno 24 sata, a njihov početak i kraj čini dnevni *Scrum*.

Prekid sprinta je moguć prije nego što vremenski period trajanja sprinta završi. Jedino vlasnik proizvoda ima autoritet da otkaže tekući sprint ili zbog utjecaja od zainteresiranih strana, razvojnog tima ili *Scrum Master*-a. Najčešći razlog za prekid sprinta je zastarjelost cilja. Takve situacije nastaju ako tvrtka mijenja smjer poslovanja te se onda mijenja i proces poslovanja što automatizmom utječe i na sam proizvod. Također promjene na tržištu ili razvojnih tehnologija mogu utjecati na razvoj situacije. Generalno sprint se otkazuje onoga trenutka kada stvarno više nema smisla s obzirom na okolnosti. Ali s obzirom na kratko trajanje sprinta, njegovo otkazivanje u rijetkim slučajevima ima smisla. U slučaju da se sprint otkaže svaki završeni dio mora biti pregledan.

Ako su pojedini dijelovi dovoljno razvijeni da mogu u uporabu Vlasnik proizvoda ih inače prihvaća. Svi nedovršeni dijelovi u toku sprinta se ponovno procjenjuju i vraćaju na popis zadataka (*Eng. Product Backlog*). Otkazivanje sprinta iznimno troši resurse, jer nakon toga je potrebno pregrupiranje te ponovno planiranje za novi sprint. Prekidi sprinta ostavljaju negativne tragove za cijeli *Scrum* tim te su iz tog razloga vrlo rijetki.

Cilj Sprinta predstavlja smjer kojim će se kretati Sprint kako bi se implementirali elementi s *Product Backlog*-a. Na takav način pruža vodstvo razvojnom timu zašto izrađuju određeni inkrement i daje svrhu cijelo ukupnom poslu.

Cilj Sprinta se izrađuje u toku planiranja sprinta, te razvojnom timu daje fleksibilnost s obzirom na funkcionalnosti koje je potrebno implementirati u toku sprinta. Izabrani elementi s *Product Backlog*-a daju jedinstvenu funkciju koja ujedno može biti Cilj Sprinta. Svrha Cilja sprinta je da udruži razvojni tim kako bi radili na istom cilju te zajedničkim angažmanom integrirali sve funkcionalnosti. Ako se ispostavi da posao koji obavljaju nije kakvim su ga predviđali zajedno s Vlasnikom proizvoda dogovaraju daljnje kretanje po Product Backlog-u u trenutnom Sprintu.

2.5.2 Planiranje Sprinta

Posao koji je potrebno izvršiti na proizvodu unutar jednog sprinta definira se kod planiranja sprinta. Na stvaranju plana sprinta sudjeluju i uključeni su svi članovi *Scrum*

tima. Planiranje Sprinta se izvršava u vremenskom periodu sa maksimalnih 8 sati za Sprint koji traje mjesec dana. Za kraći Sprint ujedno je i kraće vrijeme planiranja. *Scrum Master* je zadužen da se događaj održi i da svaki član tima razumije svoju svrhu. Ujedno se brine i podučava sve članove kako ne bi izašli iz vremenskog okvira koji je predviđen za događaj. Za vrijeme planiranja najbitnije stavke koje moraju biti dogovorene među članovima su:

- Koji dio proizvoda (inkrement) treba na sljedećem sprintu odraditi te isporučiti.
- Na koji način se želi pristupiti izradi novog inkrementa proizvoda

Slika 7: Planiranje Sprinta, Izvor: http://www.continuousautomation.com/agile-101-effective-sprint-planning-sessions/

Razvojni tim za vrijeme planiranja prognozira koje se funkcionalnosti razvijaju u toku Sprinta. Vlasnik proizvoda diskutira o cilju kojeg treba izvršiti u narednom Sprintu koji ujedno predstavlja i stavku s popisa proizvoda (*Eng. Product Backlog*). Cijeli *Scrum* tim surađuje kako bi svi shvatili što je cilj sprinta. Ulazni resursi na ovom sastanku su: „*Product Backlog*“, posljednji izrađeni dio proizvoda (inkrement), predviđeni kapacitet posla koji razvojni tim može podnijeti u toku sprinta, te učinkovitost cijelog razvojnog tima iz prethodnih Sprintova. Isključivo razvojni tim odlučuje koje sve stavke s *Product Backlog*-a ulaze u razvoj tokom sljedećeg Sprinta, jer jedino oni mogu precizno odrediti

što je realno ostvarivo.

Nakon što razvojni tim donese odluku o stavkama i obujmu posla ubrzo je i poznat cilj Sprinta, koji ujedno odgovara i *Product Backlog*-u te na takav način i navodi razvojni tim kako bi razumjeli zašto uopće izgrađuju inkrement. Nakon što je određen cilj sprinta i uključeni su elementi s *Product Backloga* razvojni tim odlučuje kako namjeravaju implementirati sljedeće funkcionalnosti u jedan gotovi inkrement u toku sprinta.

Elementi koji su izabrani za izradu s *Product Backlog*-a te plan na koji način se to namjerava implementirati se zove *Sprint Backlog*. Razvojni tim razvoj započinje s dizajniranjem sustava te izradom inkrementa koji je sačinjen od elemenata s *Product Backlog*-a, zbog toga svoj rad raščlanjuju na jedinice na dnevnoj razini koje predlažu za vrijeme planiranja. Kako bi što bolje precizirali na koji način pristupaju implementaciji te se točno zna koji dan što točno rade.

Vlasnik proizvoda im pomaže da što bolje razumiju elemente sa *Product Backlog*-a, te ako razvojni tim zaključi da za pojedine elemente im treba više ili manje vremena on ih u zajedničkom dogovoru mogu pojedinim elementima izmijeniti poziciju na *Product Backlog*-u. Razvojni tim ima i mogućnost pozvati na sastanka i druge stručne ljude koji ih mogu savjetovati oko raznih tehnikalija unutar domene poslovanja. Na kraju planiranja Sprinta, razvojni tim mora biti u mogućnosti da u potpunosti objasne Vlasniku Proizvoda i *Scrum Master*-u kako namjeravaju kao samo organizirajući tim dostići Cilj Sprinta te izraditi očekivani inkrement proizvoda.

2.5.3 Dnevni Scrum

Dnevni *Scrum* predstavlja sastanak razvojnog tima koji traje 15 minuta kako bi konstruktivno raspravili o poslu kojeg su obavljali od zadnjeg dnevnog *Scrum*-a i o poslu kojeg imaju namjere napraviti sljedećih 24 sata odnosno do sljedećeg dnevnog *Scrum*-a. Sastanak na dnevnoj razini se uvijek održava u isto vrijeme kako ne bi stvarao dodatne komplikacije ili pomake u rasporedu pojedinih članova tima.

Za vrijeme sastanka svaki član razvojnog tima prolazi sljedeće teme:

- Što sam od prethodnog sastanka napravio kako bi pomogao timu ostvariti Cilj Sprinta?
- Što ću napraviti do sljedećeg sastanka kako bi se približili ostvarivanju Cilja Sprinta?

- Dali vidim prepreke koje bi nas mogle spriječiti pri ostvarivanju Cilja?

Za vrijeme dnevnog Scrum-a razvojni tim ima priliku analizirati napredak koji su napravili prema ostvarivanju cilja, te kako se kreću u vremenskom okviru s obzirom na *Product Backlog*. Takvim pristupom razvojni tim svakim danom se samoorganizira kako će postupati u zajedničkom poslu.

Nakon završetka dnevnog *Scrum*-a pojedini članovi razvojnog tima se sastaju i detaljno raspravljaju o daljnjem toku, prilagodbi, ili problematici koja se može pojaviti u toku razvoja za vrijeme sprinta. *Scrum Master* je zadužen da razvojni tim održi sastanak, a razvojni tim je s druge strane zadužen da konstruktivno provode dnevni *Scrum*. Također *Scrum Master* podučava razvojni tim da vrijeme utrošeno za Dnevni *Scrum* ne traje više od 15-tak minuta, te da na sastanku isključivo samo sudjeluje razvojni tim i nitko više.

Cilj dnevnog *Scrum*-a jest poboljšanje komunikacije među članovima, izbjegavanje dodatnih sastanka u toku radnog dana, prepoznavanje i uklanjanje prepreka koje mogu usporiti razvoj, isticanje donošenja brzih odluka, te povećanje razine znanja cijelog razvojnog tima. U suštini ovaj sastanak predstavlja priliku za analizu dosadašnjeg rada te moguću prilagodbu za daljnji razvoj.

2.5.4 Pregled Sprinta

Pri završetku Sprinta održava se sastanak pod nazivom „Pregled Sprinta“, na ovom sastanku se pregledava razvijeni inkrement te se prilagođava *Product Backlog* ako je to potrebno. Za vrijeme sastanka razvojni tim i zainteresirane strane surađuju i raspravljaju o inkrementu koji je razvijen u toku sprinta.

Slika 8: Pregled Sprinta. Izvor: http://www.full-stackagile.com/2016/03/02/the-sprint-review-the-product-owners-meeting/

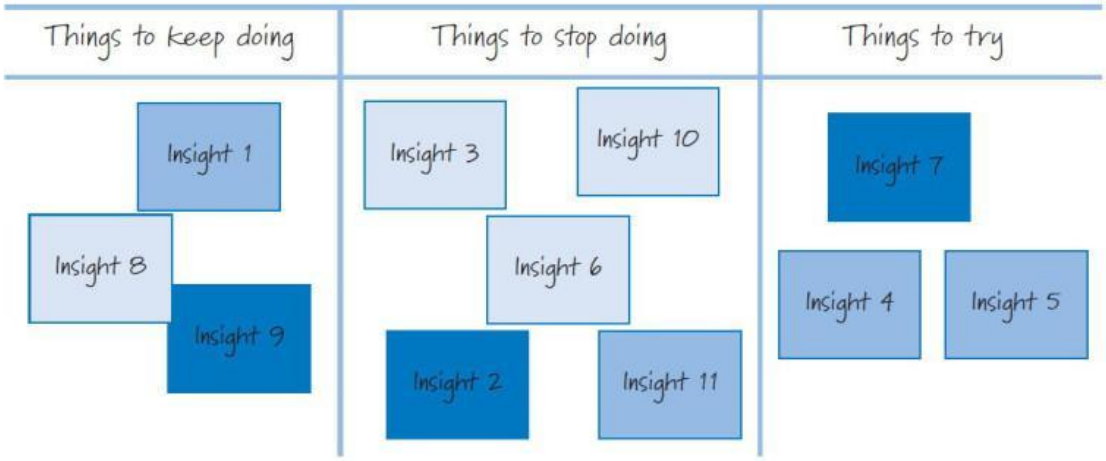
Pregled sprinta je zapravo neformalan sastanak koji potencira suradnju te izmjenu informacija između sudionika. Sastanak u prosjeku traje četiri sata za sprint od duljine četiri tjedna. Ako je sprint bio kraći onda i sastanak treba biti kraći. Zadatak *Scrum Master*-a je da se sastanak održi, da svi sudionici sastanka znaju svoju ulogu te ih podučava kako da ostanu unutar vremenskog okvira koji je predviđen za ovakav tip sastanka.

Na sastanku sudjeluje razvojni tim, *Scrum Master* te Vlasnik Proizvoda koji ujedno poziva i zainteresirane strane. Vlasnik proizvoda objašnjava koji ciljevi s *Product Backlog*-a jesu završeni te koji još preostaju. Razvojni tim diskutira o tome što je dobro prošlo u toku sprinta, koji problemi su im se javljali te na koji način su te probleme rješavali. Zatim demonstriraju inkrement koji su izradili te odgovaraju na pitanja vezana za isti. Vlasnik proizvoda raspravlja o vremenskim rokovima na *Product Backlog*-u ovisno o tome kako napreduje cijeli rad te prema tome može pomjerati pojedine rokove planirane za budućnost. Svi sudionici sudjeluju u daljnjem planiranju i sugeriraju što bi se sljedeće trebalo napraviti, tako da ovaj sastanak donosi jako puno materijala za

planiranje sljedećeg sprinta. Zatim se analizira i procjenjuje kakav utjecaj tržište ima na proizvod te koji bi sljedeće korak bio najbitniji s obzirom na okolnosti koji se javljaju na tržištu. Pri kraju sastanka se još prolazi tematika daljnjeg vremenskog toka, budžeta, potencijalne mogućnosti, te razvoj tržišta do izlaska sljedećeg inkrementa proizvoda.

2.5.5 Retrospektiva Sprinta

Predstavlja sastanak koji je retroaktivan, te se osvrće na prethodno izvršeni Sprint. Cijeli *Scrum* tim ima mogućnost analizirati i protumačiti prethodni sprint te na temelju toga planirati poboljšanja za njihovo zajedničko djelovanje u sljedećem Sprintu.



Slika 9: Retrospektiva sprinta. Izvor: https://scrumtalks.blog/2015/06/22/sprint-retrospective/

Sastanak se održava nakon *Sprint Review*-a i prije planiranja novog Sprinta. Trajanje sastanka je svedeno na tri sata za sprint koji je trajao mjesec dana, te ako je kraći sprint održan ujedno je i kraći sastanak. Kao i na prethodnim događajima *Scrum Master* se brine da se sastanak održi, da svi članovi sastanka razumiju svoje uloge, te da se održava u predviđenom vremenskom okviru. Također sudjeluje u sastanku kao ravnopravni član tima s odgovornošću prema cijelom *Scrum* procesu. U toku sastanka članovi *Scrum* tima raspravljaju o tome kakve su sve osobe sudjelovale, o međusobnim odnosima, te alatima koje su koristili. Osvrću se na sve uspješne ciljeve koje su izvršili te poboljšanja koja su izradili, i stvaraju plan kako bi poboljšali cijelo ukupni način na koji *Scrum* tim djeluje i funkcionira kao cjelina. *Scrum master* potiče cijeli tim na poboljšanja unutar cijelog *Scrum* procesa. Kako bi samostalno poboljšali

vlastiti razvoj i učinkovitost te time i zadovoljstvo cijelog tima za sljedeći Sprint.

Pri završetku sastanka *Scrum* tim dolazi do zaključka na temelju analize njihovog rada i djelovanja koje promjene namjeravaju implementirati već u sljedećem sprintu kako bi unaprijedili vlastiti rad i djelovanje na zadatke odnosno ciljeve s *Product Backlog*-a. Osvrću se na probleme na koje su naišli, te kako su se snalazili u rješavanju istih. Retrospektiva Sprinta je prilika za analizu i unaprjeđenje djelovanja *Scrum* tima na njihov vlastiti rad i cijelo ukupni proces. Kao što slika 9. prikazuje razgovaraju o tome koje stvari su dobro radili i koje će i dalje koristiti. Koje stvari nece vise raditi jer se ustanovilo da su ne učinkovite ili spore. I najvažnije od svega predlažu nove stvari koje će pokušati implementirati i vidjeti kako će to utjecati na njihov rad u sljedećem sprintu.

2.6. Scrum artefakti

Scrum artefakti su posebno razvijeni elementi/dokumenti odnosno popratna dokumentacija koja služi za mjerenje posla ili vrijednosti. Svrha tih elemenata je da osiguraju maksimalnu transparentnost kako bi imali mogućnost za analizu i adaptaciju odnosno prilagodbu posla i zajedničko promicanje samoga posla. Praćenjem samih artefakta lako se može pratiti stanje razvojnog tima u bilo kojem vremenu odnosno razdoblju rada.

2.6.1 Backlog proizvoda

Posložena lista koja sadrži sve što je potrebno za proizvod te jedini izvor sa zahtjevima i promjenama koje je potrebno implementirati na proizvodu se zove *Backlog* proizvoda (*Eng. Product Backlog*). Odgovornost za *Product Backlog* ima vlasnik proizvoda, te on određuje sadržaj, raspoloživost i redoslijed zahtjeva. Na početku izrade proizvoda na popisu se nalaze samo zahtjevi koji su do tada inicijalno poznati. Kako proizvod ili okolina evoluiraju tako je i *Product Backlog* sklon evoluiranju i prilagodbi. Zbog toga nikada nije konačan, već je dinamičan. Promjene su konstantne jer se u kontinuitetu radi na proizvodu koji mora biti primjeren, kompetitivan, koristan i s odgovarajućim funkcionalnostima koje zadovoljavaju potrebe korisnika. Dok god postoji proizvod prisutan je i njegov pripadajući *Product Backlog*. Sadržaj čini lista koja se sastoji od raznih funkcionalnosti, mogućnosti, unaprijeđena, popravka ili zahtjeva koje je u budućnosti potrebno implementirati na proizvodu.

- Redni brojevi, procjene i atributi opisa čine stavke na *Product Backlog*-u
- Sadržaj je sortiran prema važnosti odnosno vrijednosti, prioritetu, nužnosti ili riziku
- Na vrhu se nalazi uvijek stavka trenutnih aktivnosti *Scrum* tima
- Stavka za koju je potrebno više razmatranja, za koju je potrebno više odobrenja oko njene vrijednosti se na popisu nalazi niže u redoslijedu
- Više rangirane stavke su najčešće jasnije i detaljnije obrazložene na *Product Backlog*-u. Te su preciznije definirane procjene zbog jasnijih činjenica i formiranih detalja. Kako stavke idu prema dnu popisa tako je sve manje detalja
- Stavke pri vrhu, koje se odnose na sljedeći sprint su fino posložene i raščlanjene kako bi sve stale i na vrijeme bile završene unutar vremenskog okvira koji je predviđen za sprint
- Na sastanak za planiranje sprinta se uzimaju u obzir stavke s vrha liste koje razvojni tim stigne implementirati unutar jednog sprinta

U situacijijama s velikim i kompleksnim projektima odnosno proizvodima često se susrećemo s više *Scrum* timova koji rade istovremeno na istom proizvodu. Tada koriste jedan za svih isti *Product Backlog* koji ima daje opis nadolazećih poslova koji slijede u skoroj budućnosti na proizvodu. Za bolje snalaženje se onda koriste „atributi“ koji imaju funkciju da grupiraju više stavki i olakšaju snalaženje te podjelu zadataka. *Product Backlog* se održava na način što vlasnik proizvoda dodaje detalje uz odgovarajuće stavke, te procjene i sortiranje stavki prema prioritetu, hit noći ili važnosti.

To zapravo izaziva stalni proces promjena koji nastaju raspravom vlasnika proizvoda i razvojnog tima oko detalja koji se javljaju u toku samog razvoja. Tokom održavanja sve se stavke provjeravaju u više navrata, međutim vlasnik proizvoda u bilo kojem trenutku može raditi izmjene po vlastitoj procjeni.

Održavanje spada u usputnu proceduru odnosno aktivnost koju provodi Vlasnik Proizvoda s Razvojnim timom tokom provođenja Sprinta. *Scrum* tim samostalno odlučuje gdje se održava održavanje. Najčešće Razvojni tim posjeduje samo domensko znanje za održavanje, te održavanje ne prelazi više od 10% kapaciteta kojeg razvojni tim ima na raspolaganju.

Međutim odgovornost za procjenu leži isključivo na razvojnom timu, jer su oni ti koji u konačnici obavljaju razvoj. Utjecaj na njih vrši Vlasnik proizvoda kao potpora u razjašnjavanju stavki te odabiru kompromisa.

Na sljedećoj slici prikazan je primjer jednog *Product Backlog*-a:

ID	Theme	As a/an	I want to...	So that...	Notes	Priority	Status
1	Functional	Student	To have a calendar archive of all activities for different clubs	I can know all the activities I can attend		High	Not coded yet
1	Functional	Student	Add my activities of interest to my personal calendar	I can organize my personal events	Club meetings, internship fairs, appointments, etc	High	In progress
1	Functional	Student	Have reminders of my events	I remember them		Low	To do
1	Non-functional	Student	Be able to group my events and color-code it	I can view my calendar easier		Low	To do
2	Functional	Student	Have a platform so I can view other student stories	I can hear and learn from others and understanding where they are coming from	Not limited to storyboards, video, audio, and other visuals	High	Done
2	Functional	Student	Be able to upload my story and perspective on the website	I can share my experience with others		High	In progress
2	Non-functional	Advisor	Be able to like and share stories I see	Show other struggling students		Low	To do
2	Non-functional	Student	Be able to view all stories in one place	It's easier for me to view it		Medium	Done
2	Functional	MyStory contributor	Be able to upload files larger than 200 MB	I can show a longer video/audio		Low	To do
2	Functional	MyStory contributor	Be able to edit my MyStory post	I can update new events that occurred		High	In progress
2	Non-functional	Student	View MyStory in sub categories by majors and/or experiences	I can view the stories most related to me	Experiences: how to get an internship, job, etc	Medium	Done
~~3~~	~~Functional~~	~~Student~~	~~Have a general Q&A forum for any question~~	~~I can ask mentors questions~~		~~Medium~~	~~To do~~
3	Functional	Student	Have a Q&A forum for each individual post	I can ask the MyStory additional questions if their post does not answer all my questions		Medium	Not coded yet
~~4~~	~~Functional~~	~~Senior in high school~~	~~Be able to contact an advisor~~	~~I can have an early overview of the university and create a close connection to my advisor~~	~~See their email, number, and office hours~~	~~Low~~	~~To do~~
5	Non-functional	Bothell Student	See Seattle/Bothell's campus resources	I have extra help if I need it		Low	Not coded yet
5	Function	Bothell Student	Add resources to resources lift if that already listed	I can share it to everyone of what helped me		Low	To do
6	Functional	MyStory contributor	Have an anonymous option	I can still have my privacy		High	To do
~~7~~	~~Functional~~	~~Student~~	~~Geo-tagged locations for my events~~	~~I can find them easily~~	~~Across Bothell and Seattle campus~~	~~Low~~	~~To do~~

Tablica 1: Primjer Product Backlog-a. Izvor: https://students.washington.edu/sle96/wordpress/product-backlog/

Na samom *Product Backlog*-u su točno vidljive funkcionalnosti nekog proizvoda, njihov prioritet pa i status koji označava dali je taj inkrement završen ili ne. Neki *Product Backlog*-ovi imaju i predviđeno vrijeme za određenu funkcionalnost.

2.6.2 Backlog sprinta

„Popis zadataka odnosno lista koju razvojni tim mora izvršiti, razviti i implementirati u nadolazećem Sprintu naziva se popis zdataka sprinta (*Eng. Sprint Backlog*). Učestala

je praksa da se *Sprint Backlog* najčešće vizualno prikazuje na zidnoj ploči, što osigurava da je konstantno vidljiv prikaz stanja korisničkih priča na *Backlog*-u. Također su u *Sprint Backlog* uvršteni svi rizici asocirani na razne zadatke. Sve ublažujuće aktivnosti koje služe za prepoznavanje pojedinih rizika također su navedene kao zadatci na *Sprint Backlog*-u. Jednom kada je *Sprint Backlog* finaliziran i izvršen od strane *Scrum* tim-a, nije moguće više dodavati nove korisničke priče; kako god, zadatci koji možda nedostaju ili su nezamijećeni od strane korisničkih priča mogu biti dodane ako je potrebno. Ako se u toku Sprinta pojave novi zahtjevi oni se dodaju na cjelokupni *Product Backlog* i izvršavaju u sljedećem Sprintu. "[14]

User Story	Tasks	Day 1	Day 2	Day 3	Day 4	Day 5	...
As a member, I can read profiles of other members so that I can find someone to date.	Code the ...	8	4	8	0		
	Design the ...	16	12	10	4		
	Meet with Mary about ...	8	16	16	11		
	Design the UI	12	6	0	0		
	Automate tests ...	4	4	1	0		
	Code the other ...	8	8	8	8		
As a member, I can update my billing information.	Update security tests	6	6	4	0		
	Design a solution to ...	12	6	0	0		
	Write test plan	8	8	4	0		
	Automate tests ...	12	12	10	6		
	Code the ...	8	8	8	4		

Tablica 2: Sprint Backlog. Izvor: http://magnificent11.com/2016/05/15/scrum-there-are-three-artifacts-in-scrum/

Sprint Backlog predstavlja zbir stavki zadataka koje se nalaze na cjelokupnom *Product Backlog*-u koje se izvršavaju na sljedećem Sprintu zajedno s planom kako izvršavaju inkrement, te ciljem Sprinta. Razvojni tim zapravo dokumentira procjenu koji dio funkcionalnosti izrađuju u sljedećem sprintu i na koji način razvijaju sljedeći inkrement proizvoda. Sav posao koji je njima potreban za ostvarivanje cilja naveden je na *Sprint Backlog*-u. Na njemu je vidljivo dovoljno detalja kako bi se na razini dnevnog *Scrum*-a

[14] Satpathy Tridibesh, Scrum body of knowledge, SCRUMstudy™, Phoenix, Arizona, 2003.,str. 228.

mogle uočavati promjene u toku rada. U toku Sprinta dolazi i do promjene sadržaja na *Product Backlogu*. Jer razvojni tim dodaje zadatke u toku sprinta, kako rade po planu često se nalaze u situaciji da nauče nešto novo o poslu koji im je neophodan da bi ostvarili Cilj Sprinta. Kako se pojavljuje potreba za novim nepredviđenim poslom razvojni tim ga dodaje na *Sprint Backlog*.

Fleksibilno pristupaju ažuriranju istoga. Npr. kada se smatra da su pojedini elementi nepotrebni znaju se i izbaciti s popisa stavki. Međutim samo Razvojni tim ima pristup i mogućnost mijenjanja sadržaja na *Sprint Backlog*-u u toku Sprinta. U globalu *Sprint Backlog* predstavlja vidljivu sliku posla u realnom vremenu na kojem radi Razvojni tim sa svim namjerama kako planiraju izvršiti Cilj sprinta. Zbog takvog pristupa i ažuriranja na dnevnoj bazi u bilo kojem trenu može se izraziti preostali posao na Sprintu. Razvojni tim prati posao na dnevnoj osnovi i procjenjuje napredak te na takav način može i manipulirati odnosno upravljati njegovim napretkom prema Cilju. *Scrum* se ne zamara potrebnim vremenom za izvršenje pojedine stavke, već preostali posao i datum isporuke čine jedine varijable o kojima se vodi računa.

Inkrement predstavlja popis stavki s *Product Backlog*-a koji su uvršteni u jedan Sprint. Te u tom Sprintu moraju biti potpuno završeni i spremni za uporabu kao novi sastavni dio cijelog proizvoda. U idealnom slučaju u toku sprinta se izrade, testiraju i dokumentiraju sve stavke s *Product Backlog*-a koje su za taj sprint predviđene, te se kao dodatna funkcionalnost pridodaje prethodno izrađenim inkrementima proizvoda. Prednost ovakvog pristupa razvoja gdje se izrađuju potencijalno dostavljive verzije nekog proizvoda leži u činjenici što je puno veća kontrola u strateškom planiranju daljnjeg razvoja u slučaju da kupca nešto izmjenjuje kako bi se daljnji razvoj kretao u željenom smjeru. Te prema inkrementima i sam kupac bolje prepoznaje svoje prave zahtjeve koji su mu potrebni.

Definicija izvršenoga (*Eng. Definition of Done*) ne predstavlja artefakt već definiciju pravila koju tim dogovara na početku samog planiranja. „Na početku samog projekta se definira i obrazloži točno značenje za *Definition of Done* od strane cijeloga tima. Ona točno opisuje koje će čimbenike kvalitete sadržavati svaki inkrement, koje moraju biti ispunjene kako bi se jedna stavka s *Product Backlog*-a mogla smatrati gotovom

(***DONE***).“[15] Vrlo je važno da to bude konkretno dogovoreno na razini cijeloga tima. Po završetku jednog sprinta *Product owner* ima obavezu ispitati sve funkcionalnosti samog inkrementa kako bi utvrdio kvalitetu na dogovorenoj razini definicije gotovog. Jedan *Scrum* tim se na primjer može dogovoriti da definicija izvršenoga za jedan inkrement znači, odrađeni posao, izvršeno programiranje, dokumentiranje i testiranje samog inkrementa.

2.6.3 Dijagram sagorijevanja

Dijagram sagorijevanja (*Eng. Burn-Down chart*[16]) već dugo vremena nije više oficijalni artefakt *Scrum* metodologije. Međutim još uvijek se često zna upotrebljavati u praksi. Poznata su dva tipa ovog dijagrama:

- ***Sprint Burndown Chart***[17]: ovaj dijagram služi kako bi razvojni tim mogao popratiti napredak posla unutar jednog sprinta

- ***Release Burndown Chart***[18]: prikazuje napredak u razvoju jednog proizvoda od verzije do verzije, u nekim slučajevima se uzima i veći vremenski period i onda se promatra napredak svih funkcionalnosti prema prioritetima

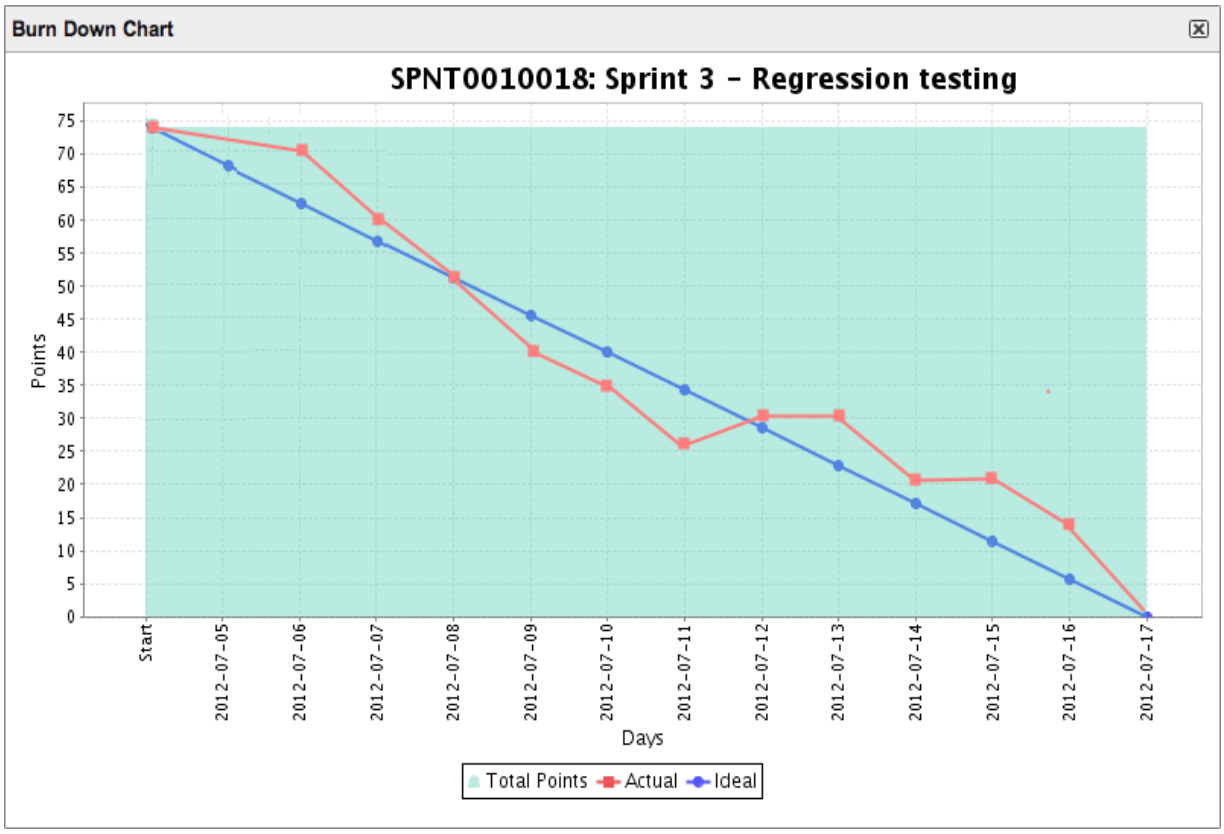
Generalno gledano jedan takav dijagram treba sadržavati sljedeće:

- X os za prikaz radnih dana odnosno vremena
- Y os za prikaz preostalog napora/resursa koje treba uložiti
- Idealnu crtu uloženih napora/resursa
- Stvarnu crtu uloženoga
- Početak mjerenja
- Završetak mjerenja

[15] Definition of Done – prijevod iz: Joachim Goll und Daniel Hommel, Mit Scrum zum gewünschten System, Springer Vieweg, Wiesbaden, Njemačka, 2015, str. 96
[16] Burn-Down chart Eng. – dijagram sagorjenog posla, odnosno obavljenog posla u nekom vremenu
[17] Sprint Burndown Chart Eng. – dijagram sagorijenog posla u toku jednog sprinta
[18] Release Burndown Chart – dijagram sagorijenog posla za verziju

Međutim različite kompanije koriste i drugačije principe označavanja Y osi. Može se koristiti:

- Posao koji je potreban za završiti / odnosno resursi
- Vrijeme potrebno za izvršavanje određenog posla
- Bodovi priče[19], čine ju količina posla kojeg treba obaviti, kompleksnost posla i rizik ili neizvjesnost istog tog posla



Slika 10: Burn-Down Dijagram. Izvor: https://docs.servicenow.com/bundle/kingston-it-business-management/page/product/sdlc-scrum/task/t_SprintStoryBurnDownCharts.html

Na slici 10. kao primjer je vidljiv jedan Sprint *Burn-Down* dijagram koji nam pokazuje na X osi datume i vrijeme koje je potrebno za obavljanje tog sprinta. Iz toga vidimo da sprint počinje 04.07.2012. a završava 17.07.2012. Na Y osi se nalaze tzv. Story Points koji u našem slučaju pokazuju omjer posla koji treba biti izvršen kroz sprint. Plava linija nam pokazuje idealnu crtu kojom brzinom i sa kolikim trudom je potrebno za izvršenje svih zadataka u roku koji je predviđen. Ova crta proizlazi iz matematičkih proračuna

[19] Točke priče (Eng. Story Points) – mjerna jedinica koja izražava procijenu potrebnog napora koji je potrebno uložiti da bi se u potpunosti implementirala neka funkcionalnost proizvoda ili bilo kojeg drugog posla

koji se temelje na procjenama. Crvena crta prikazuje nam stvarni napredak posla kroz vrijeme. Te uspoređujući ove dvije linije vidljivo je, koliko je stvarno efikasan razvojni tim u izvršenju posla. Same fluktuacije ovise o raznim parametrima i čimbenicima. Cilj ovog dijagrama je da nam približno (pošto se radi o procijeni) prikaže napredak prema završetku sprinta i da procjenu vjerojatnosti da će svi poslovi biti izvršeni unutar sprinta.

3. Praktični primjer

U nastavku slijedi praktični primjer *Scrum* metodologije na jednom timu Informatičara koji se bavi Mrežnom i Sistemskom administracijom informacijskog sustava u jednoj tvrtci. Opisi posla jednog sistemskog administratora vrlo je šarolik i opsežan, te ovisno o samoj veličini tvrtke/broju zaposlenih ovisi i broj administratora koji rade na tom radnom mjestu. Što su sustavi kompleksniji postaje ih i teže za održavati, te iz tog razloga svaki administrator teži prema automatizaciji raznih aktivnosti sa kojima se susreće u toku svoga radnog dana. Sama kompleksnost sustava traži i različite profile adminstracije, pa tako i samog administratora ovisno o njegovim vještinama. Danas su među administratorima najzastupljenije sljedeće podjele: Mrežni i Sistemski administratori, administratori baza podataka, Web administratori i Telekomunikacijski administratori.

Scrum Framework[20]

Do sada smo najveću pažnju posvetili *Scrum* metodologiji koja se koristi isključivo u svijetu razvoja informacijskih sustava odnosno Software-a. Međutim to nije jedino mjesto gdje se *Scrum* može implementirati. *Scrum* se može upotrijebiti u bilo kakvom okviru gdje ljudi pokušavaju uz produktivnost i kreativnost proizvesti kvalitetne proizvode. Adaptirajući se prema cilju ovisno o problemima koji im se u toku javljaju kao prepreke ili promjene okoline samog proizvoda na tržištu. *Scrum* kao takav ponajprije služi kao alat za razvoj kompleksnih proizvoda, čime isti postaje nepotreban za lakše i kratke proizvodne i razvojne procese. Sami procesi koji su definirani unutar *Scrum*-a se koriste do te razine koje je zapravo i potrebno, na koji način i kojoj količini

[20] Engl. Scrum Framework. Scrum radni okvir.

sve zavisi o potrebama. Sve to skupa omogućava veliku slobodu, a ta sloboda zavisi zapravo o sposobnostima samih zaposlenika. Zaposlenici mogu samostalno reagirati i sa tom slobodom konstruktivno raspolagati.

3.1. Radna okolina

Poslovanje svakog modernog poduzeća zasniva se na informacijskim tehnologijama pa samim time što je veće poduzeće i sam infomacijski sustav postaje sve kompleksniji. U današnje vrijeme vrlo je važno da su informacije i sami informacijski sustavi dostupni 24h dnevno, pa tako i 7 dana u tjednu.

Pad sustava može imati katastrofalne učinke na poslovanje jednog poduzeća, što se posebno odnosi na sustave koji su temeljeni na mrežnoj komunikaciji ili različitim web uslugama. Poslovanje poduzeća zbog toga jako zavisi o kvaliteti i izdržljivosti infrastrukture cijelog sustava. Neke od odgovornosti i aktivnosti Sistem Administratora:

- Praćenje IT trendova
- Praćenje hardverske opreme (mrežna oprema, serveri, pc-evi, konferencijska oprema itd.)
- Praćenje Software-ske podrške
- Projektiranje, instalacije, konfiguriranje i održavanje mrežne infrastrukture
- Nabava opreme (hardware općenito)
- Dizajn lokalne mreže (LAN) i bežićne mreže (WLAN)
- dizajn širokojasnih mreža (WAN)
- razumijevanje mrežnih tehnologija i protokola, OSI model
- poznavanje domenskog okruženja
- poznavanje Hardware-a serverske arhitekture
- poznavanje Windows i Linux serverskih operativnih sustava
- projektiranje i implementacija rješenja
- obuka korisnika
- podrška korisnicima
- upravljanje korisničkim računima (Active Directory i e-mail Administracija)
- ažuriranje postojećeg Software-a
- licenciranje
- Troubleshoting

- Optimizacija
- Enkripcija i dekripcija medija za pohranu podataka
- Backup (Pohrana podataka)
- Sigurnosni aspekti
- Denvni nadzori, održavanje i upravljanje
- Zaštita od zločudnih programa
- Dokumentiranje aktivnosti i pisanje procedura

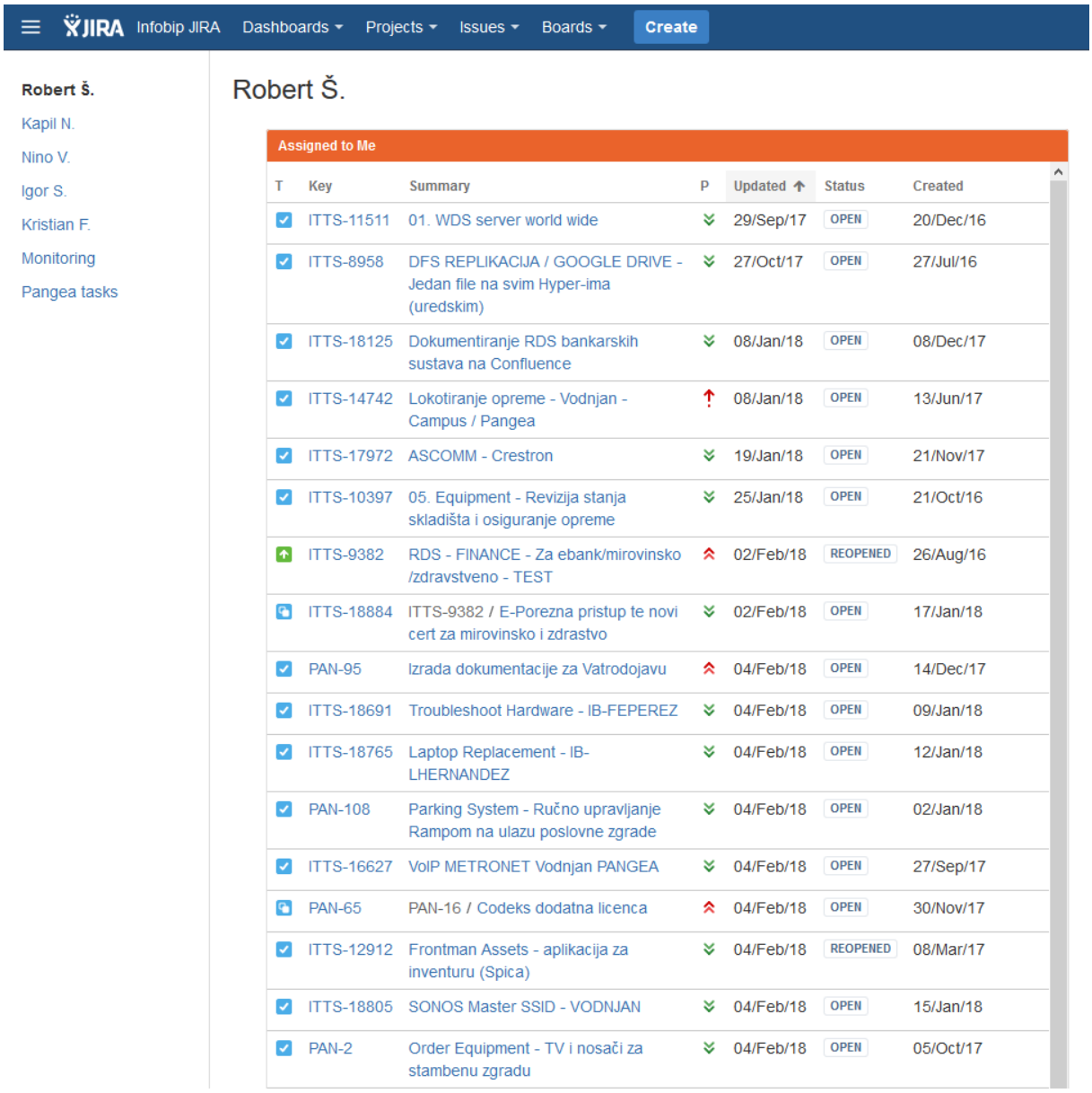
Sve ove aktivnosti i znanja u kombinaciji služe za nadgledanje i upravljanje informacijskim sustavom kako bi funkcionalnost sustava bila uvijek na najvišoj razini. Već iz ovog popisa je vidljivo da to ne može jedna osoba odrađivati te se iz tog razloga ovi poslovi raspodjeljuju na cijele timove Administratora gdje su zadaci jasno podijeljeni prema odgovornostima. Broj timova i ljudi zavisi o samoj kompleksnosti i veličini sustava.

3.2. Korišteni alat

JIRA je popularni alat tvrtke Atlassian sa kojim se demonstrira praktični primjer *Scrum* metodologije. Jira se koristi kroz web sučelje i služi kao alat za praćenje problema/zadataka (*Eng. task/issue*) nekog projekta.

Svaki djelatnik je član određenog projekta na JIRI. Na svakom projektu ujedno radi i tim ljudi. Svaki djelatnik ima svoju kontrolnu ploču na kojoj su mu vidljivi problemi odnosno zadaci kojima se on susreće i bavi, a ti zadaci su direktno vezani na projekt u kojem djelatnik sudjeluje.

Kako se u svakodnevnom radu djelatnici susreću novim izazovima i zadacima koje treba izvršiti potrebno je te zadatke i dokumentirati. Svaki od tih zadatak se dokumentira u JIRI. Zadatak je otvoren sve dok problem nije otklonjen. Neki zadaci su i puno opširniji te iziskuju puno napora i truda koko bi se izvršili do kraja. Zadatak je završen kada su ispunjeni svi definirani ciljevi, koji zavise o samoj kompleksnosti problema sa kojim se djelatnik susreće.

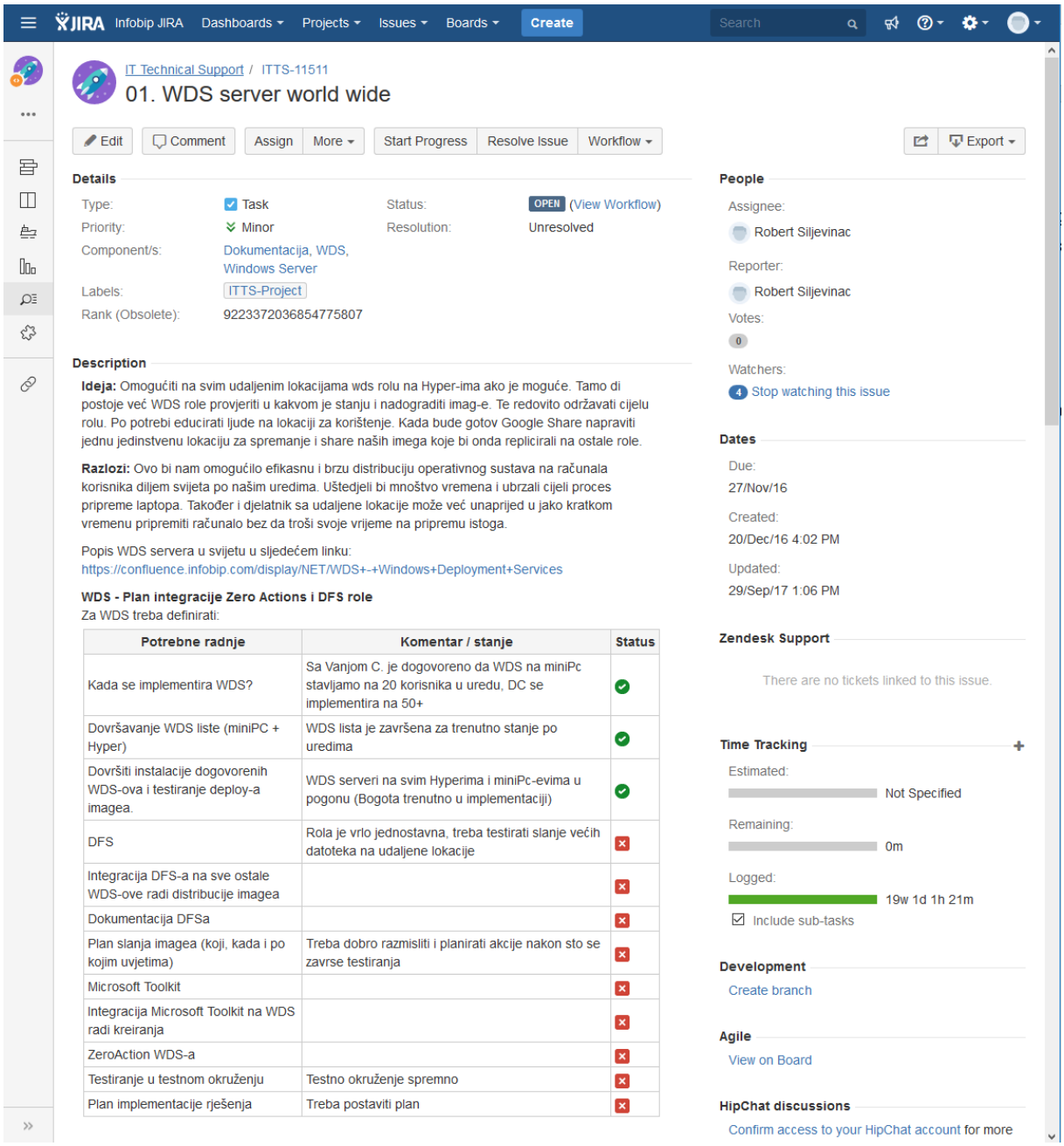


Slika 11: JIRA Kontrolna ploča. Izvor: autor.

Na slici 11. je vidljiv dio kontrolne ploče jednog djelatnika. Djelatnik ima popis svih otvorenih problema/zadataka (*task/issue*) koje mora još dovršiti ili su u procesu izrade. Svaki od tih zadataka ima:

- svoj ključ (*Key*) koji nam govori o kojem projektu je riječ i koji je redni broj tog zadatka
- sažetak (*Eng. Summary*) koji asocira odmah i o kakvom problemu/zadatku je riječ
- prioritet (P) označava koliko je bitan taj zadataka

- datum zadnje nadopune (*Eng. Updated*) kada je zadnji puta nešto rađeno na tom problemu odnosno zadatku
- status, govori nam dali je zadatak otvoren, završen, ponovno otvoren itd…
- datum kreiranja (*Eng. Created*) dan kada je taj zadatak/problem otkriven odnosno kada se počelo na tome raditi



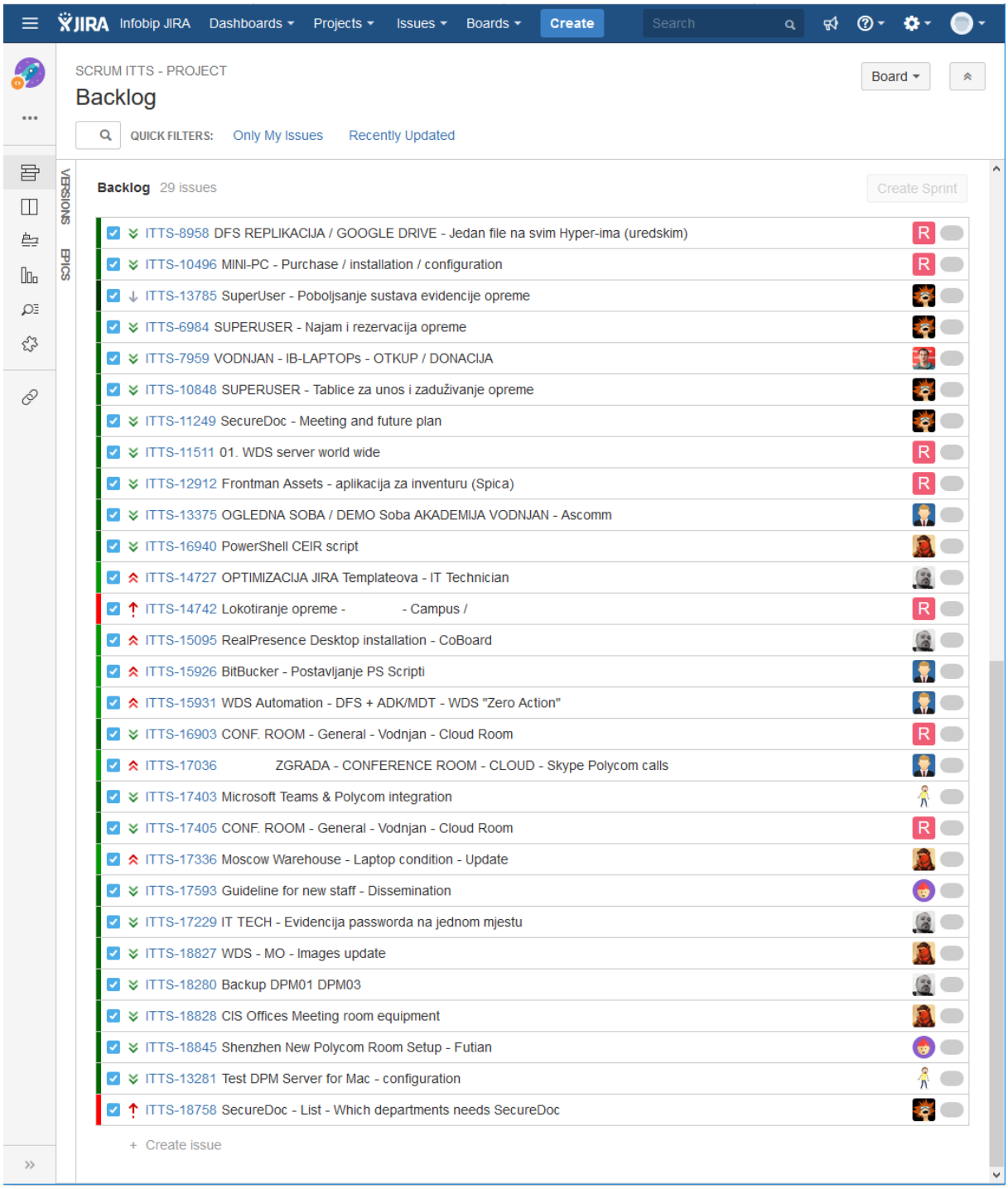
Slika 12: Prikaz zadatka/task u JIRA-i. Izvor: autor.

Slika 12. prikazuje kako izgleda jedan od zadataka koji je definiran unutar samog alata.

Sam zadatak sastoji se od određenih detalja, opisa, popratnih informacija, komentara te specifičnih alata koji služe za uređivanje samog zadatka te dokumentiranje napretka.

3.3. Backlog Projekta

Projekt unutar JIRA-e ima svoj *Backlog* na kojem se nalaze specifični zadaci koje tim mora izvršiti.



Slika 13: JIRA Backlog. Izvor: autor.

Na samom Backlogu se nalaze specifični zadaci svih članova tima koji rade na istom projektu. Međutim ovdje nisu svakodnevni operativni zadaci. Na ovom popisu završavaju specifični zadaci koji su projektnog tipa, te samim time je potrebno više vremena i truda za izvršiti ih. Gdje je potrebno planiranje, testiranje, implementacija i dokumentiranje na kraju. S obzirom na veličinu određenih projektnih zadataka često se događa da se kreiraju pod-zadaci. Pod-zadatak je manji zadatak koji je potreban da se izvrši kako bi dobili određenu funkcionalnost glavnog zadatka.

Takvim pristupom segmentira se posao u više manjih komponenti koje je onda lakše rasporediti vremenski kroz više sprinteva. Stavke na *Backlog-u* se stalno mijenjaju i nadopunjavaju zbog same prirode ovog posla. Kako stalno raste potreba za nadogradnjom ili automatizacijom sustava uvijek iz nova postoje sve bolja i bolja rješenja koja se daju implementirati ili nadograditi.

3.4. Sprint u praktičnoj okolini

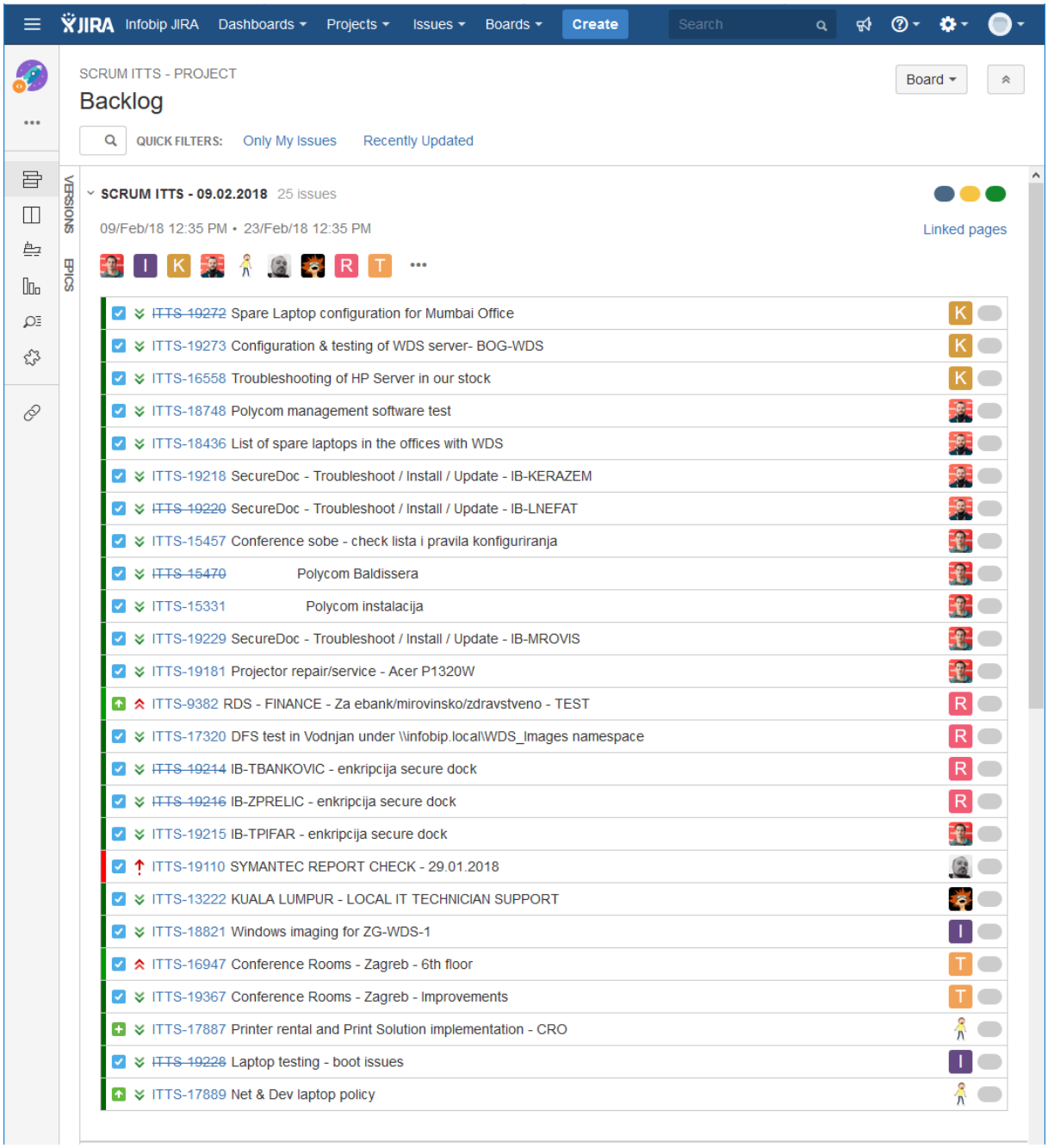
Tim koji radi na projektu ima dogovor da sprint traje 14 dana, što na neki način predstavlja optimalno vrijeme za rješavanje zadataka unutar samog sprinta. Treba uzeti u obzir da tim ljudi osim zadataka unutar sprinta ima i svoje redovite obaveze i operativne zadatke koje svakodnevno obavljaju. Cilj sprinta je da se što više projektnih zadataka riješi u toku sprinta. Projektni zadaci su tako koncipirani da ako budu uspješni dovode nove automatizacije koje će olakšati svakodnevni rad djelatnicima na istom projektu.

Kriteriji po kojima se biraju zadaci koji ulaze u sljedeći sprint:

- Zaostali zadaci s prethodnog sprinta
- Prioritetni projektni zadaci (zadaci visokih prioriteta)
- Prioritetni *pop-up* zadaci (zadaci koji su se samostalno pojavili kroz rad i imaju visoki prioritet)
- Zaostali zadaci koji nisu nadopunjeni više od 15 dana

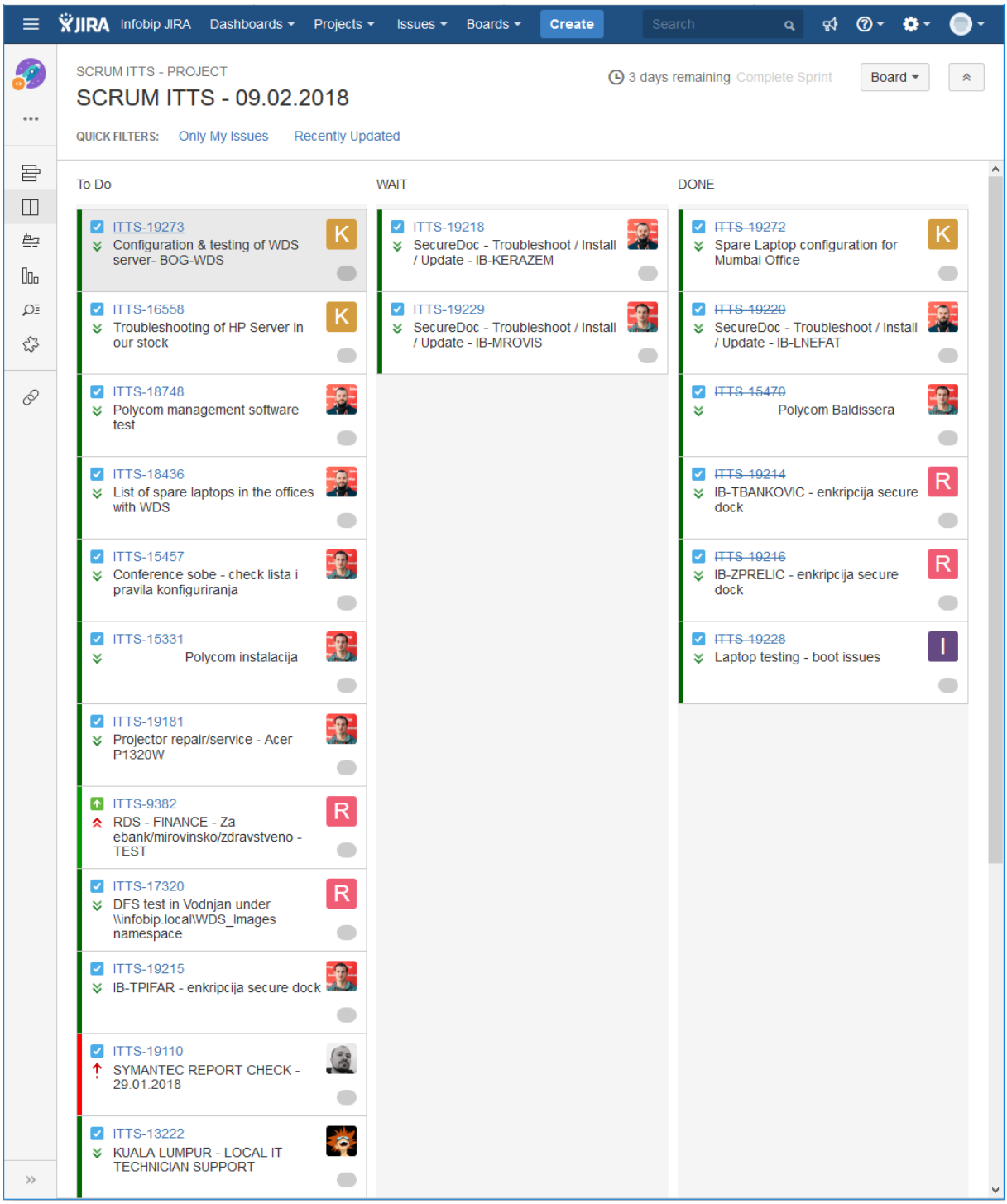
Orijentirajuće se prema ovim kriterijima tim kod planiranja sprinta zajedno s voditeljem tima dogovara koji zadaci će se uzimati u obzir te na koji način će se pristupati rješavanju problema. Svakom članu tima se dodjeljuju zadaci prema njihovim

vještinama i odgovornostima. Osim zadataka djelatnici moraju i definirati početak i kraj sprinta. Nakon planiranja svi zadaci se pojavljuju na *Sprint Backlog*-u.



Slika 14: JIRA Sprint Backlog

Sprint Backlog je lista koja pokazuje sve zadatke raspoređene prema dijelatnicima koji ulaze u određeni sprint. Na samom vrhu su vidljivi dogovoreni početak i kraj sprinta. Za bolji pregled na status pojedinih zadataka unutar sprinta koristi se specifičan pregled sprinta:



Slika 15: JIRA status taskova u Sprintu. Izvor: autor.

Polje „*TO DO*“[21] označava koje zadatke treba još izvršiti. U polju „*WAIT*“[22] završavaju zadaci koji zbog nekog vanjskog utjecaja na koje članovi tima ne mogu utjecati. Polje

[21] To do, Eng. – za napraviti
[22] Wait, Eng. – na čekanju

„*DONE*“[23] predstavlja zadatke koji su u potpunosti izvršeni i kompletirani. Cilj je da što više zadataka iz „*TO DO*“ polja završe u „*DONE*“ polju. Što ih je više to je i sam sprint bio učinkoviti ji te time predstavlja uspješno obavljen posao od strane samog tima.

U toku sprinta tim svaki dan održava *daily meeting*[24] u trajanju od 10-tak minuta. Na dnevnom sastanku se u kratko prolazi kroz problematike prethodnog dana i predviđa se posao koji će biti izvršen u toku tekućeg dana. Ovo je prilika da svatko od članova iznese problem ako postoji te da se međusobno savjetuje. Takvim pristupom međusobno si pomažu u samom radu. Osim toga pojedini članovi tima znaju biti i na udaljenim lokacijama diljem svijeta, te iz tog razloga ovaj sastanak ima još veće značenje. Sama distanca zbog lokacije razdvaja ljude, pa se ovim pristupom održava zajedništvo i pripadnost timu.

Osim dnevnog sastanka postoji i *weekly meeting*[25] koji traje trideset minuta gdje se malo dublje ulazi u problematiku i samo stanje zadataka u sprintu. Na tom sastanku se provjeravaju prioriteti i stanje. Ako su se pojavile nove nepredviđene situacije koje iziskuju neke radnje, prema tome se pristupa s obzirom na važnost same situacije. Tu se odlučuje dali se to svrstava u svakodnevnu aktivnost, trenutni sprint ili se svrsta na Backlog te se čeka početak novog sprinta.

Retrospektiva sprinta se vrši na mjesečnoj razini te obuhvaća dva sprinta. Takvim pristupom dobivamo pogled na veći vremenski period. Na retrospektivi se raspravlja o pogreškama koje su učinjene u toku sprinteva. Sugeriraju se promjene kako bi se određene stvari bolje planirale u budućnosti.

Ovakvim pristupom kao rezultat se pokaže jedan samoorganizirajući tim ljudi koji prema *Scrum* procesu pristupa rješavanju problema i zadataka koji im se javljaju u domeni Sistemske administracije.

[23] Done, Eng. - Gotovo
[24] Daily Meeting, Eng. – Dnevni sastanka
[25] Weekly Meeting, Eng. – Tjedni sastanak

4. Zaključak

Scrum metodologija u današnjem modernom dobu predstavlja vrlo učinkovit pristup razvoju kompleksnih proizvoda. Samim time što u kratkim vremenskim rokovima nastaju funkcionalni inkrementi nekog proizvoda pa ih se ujedno može i isporučiti krajnjim korisnicima. Pošto je razvoj koncipiran na manje iteracije razvoja dijelova proizvoda, cijeli proces se sa lakoćom unutar kratkih vremenskih perioda može prilagoditi potrebnom stanju na tržištu. Sam pristup ima najveći fokus prema članovima tima koji uz veliku fleksibilnost u dinamičnoj okolini djeluju na sam proizvod.

Osim kod razvoja proizvoda ovaj se proces može integrirati i u kompleksne radne okoline, gdje prevladava jako velika i brza dinamika na svako dnevnoj razini. Korištenjem *Scrum*-a dobiva se mogućnost dobro organizirati jednu ili više skupina ljudi koji rade na određenom projektu kako bi povećali efikasnost njihovog rada i bolje organizirali cijeli pristup radu.

Velika prednosti metodologije leži u samom procesu gradnje koji se brzo prilagođava okolini i promjenama, te se samoorganizirajući prilagodi nastalim promjenama kako bi se sve problematike otklonile odnosno nadogradile.

U praktičnom primjeru Sistemske Administracije ovakav pristup radu se pokazao jako pozitivnim sa velikim učinkom u kratkom vremenu. Članovi tima osim što se bave svakodnevnim aktivnostima održavanja i nadgledanja kompleksnog informacijskog sustava ovim pristupom dobili su povećani fokus na projektne zadatke koji ako budu izvršeni, njima omogućavaju i lakše održavanje istog sustava. Dobili smo veliki rast produktivnosti, prilagodljivosti, efikasnosti i automatizaciju.

Uz dokumentaciju koja se vodi kroz cijeli proces osigurana je transparentnost u svim ciklusima razvoja ili pristupanju problemima odnosno zadacima. Pa tako dobivamo i širu sliku samog projekta što je vrlo korisno u predviđanju mogućih problema.

Fokus je također postavljen na timski rad koji čini jednu od osnova metodologije. Timski duh je vrlo važan faktor koji čini tim jačim od prostog skupa pojedinaca. Dobrim usmjeravanjem tima prema cilju uz konstantno prilagođavanje kroz retrospektive dobivamo visoku produktivnost.

Nedostatak koji je zamijećen u praktičnom primjeru javlja se u situacijama vanjskog utjecaja. Sistem Administratori nažalost nemaju privilegiju da ih netko kao „*Scrum Master*“ štiti od vanjskih utjecaja dok se nalaze u sprintu kao što to imaju razvojni timovei prilikom razvoja programskih rješenja. Sistem administratori su odgovorni za funkcionalnost informacijskog sustava 0/24 h svakim danom, te je ujedno to jedan od glavnih faktora koji remeti koncept i stvara određenu nestabilnost u procesu.

Međutim *Scrum* metodologija se pokazala vrlo učinkovitom u ovakvoj okolini. Bez obzira na svakodnevnu količinu posla zadržavanjem fokusa na zadatke u sprintu, tim administratora svakim završetkom sprinta prezentira novu automatizaciju ili nadogrdanju informacijskog sustava.

5. Literatura

1. Satpathy Tridibesh, Scrum body of knowledge, SCRUMstudy™, Phoenix, Arizona, 2003
2. Joachim Goll und Daniel Hommel, Mit Scrum zum gewünschten System, Springer Vieweg, Wiesbaden, Njemačka, 2015
3. Dan Rawsthorne and Doug Shimp, Exploring Scrum: The Fundamentals Second Edition, 2003
4. Scrum History, http://www.scrumguides.org/download.html , 15.12.2017.
5. Scrum Guidse, http://www.scrumguides.org/scrum-guide.html , 16.12.2017.
6. Agile Manifesto, http://www.agilemanifesto.org/ , 18.12.2017.
7. Scrum Allianze, https://www.scrumalliance.org/ , 20.12.2017.
8. Scrum Team, http://www.agile42.com/en/agile-info-center/scrum-roles/ , 24.12.2017.
9. Scrum Team, http://www.scrum-institute.org/Scrum_Roles_The_Scrum_Team.php , 24.12.2017.
10. Scrum općenito, https://www.atlassian.com/agile , 25.12.2017.
11. Scrum Development Team, https://www.scrum.org/resources/what-is-a-scrum-development-team , 1.1.2018.

6. Popis slika

Slika 1: Najznačajniji povijesni Scrum događaji 3
Slika 2: Scrum Proces 8
Slika 3: Tri osnovne uloge u Scrum-u 10
Slika 4: Osobnosti vlasnika proizvoda 12
Slika 5: Scrum Master 13
Slika 6: Scrum događaji 16
Slika 7: Planiranje Sprinta 19
Slika 8: Pregled Sprinta 22
Slika 9: Retrospektiva Sprinta 23
Slika 10: Burn-Down Dijagram 30
Slika 11: JIRA Kontrolna ploča 34
Slika 12: Prikaz zadatka/task u JIRA-i 35
Slika 13: JIRA Backlog 36
Slika 14: JIRA Sprint Backlog 38
Slika 15: JIRA status taskova u sprintu 39

7. Popis tablica

Tablica 1: Primjer Product Backlog-a 26
Tablica 2: Sprint Backlog 27

8. Sažetak

U današnje vrijeme *Scrum* metodologija predstavlja moderan pristup razvoja, ne samo informacijskih programskih rješenja već bilo kakvih kompleksnih proizvoda. Koristeći pravila, uloge i tehnike dokumentiranja koju metodologija zahtjeva, te pravilnim korištenjem svega što *Scrum* nudi svaka se radna okolina može preinačiti u jednu samoorganizirajuću skupinu odnosno tim koji u vrlo kratkom roku pokazuje iznimnu produktivnost. Srž metodologije leži u timskom radu koji se prilagođava okolini stvarajući što veću učinkovitost, smanjujući nedostakte i rizike.

Scrum Metodologija se pokazala vrlo učinkovitom u izvršavanju devnih zadataka i dugoročnih projekata unutar radne okoline Sistem administratora i što je najbitnije od svega značajno je povećala produktivnost njihovog rada.

Ključne riječi: *Scrum*, Agilne metode, *Scrum* Tim, *Scrum* događaji, *Scrum* Artefakti

9. Summary

Today *Scrum* methodology represents a modern way in developing new products, not only in software development, but in any type of production. Using *Scrum* rules, roles and documentation techniques which this methodology requires, every working environment can replace a group of individuals with a well performing self-organised team in a short period of time. The key of this methodology is the teamwork which adapts to a variety of incoming requests delivering high quality products in the most efficient way and in the same time reducing deficiencies and risks.

The *Scrum* Methodology proved to be very effective in perfomring daily tasks and long term projects in a team of System Administrators and, which is most important, significantly increased the productivity of their work.

Keywords: Scrum, Agile methods, *Scrum* Team, *Scrum* events, *Scrum* Artefacts