Prijedlog podučavanja agilnih metoda razvoja softvera korištenjem alata Microsoft Project 2010

Autori: Aleksander Radovan, Veleučilište Velika Gorica

Tomislav Gligora, Veleučilište Velika Gorica

Davorin Valenčić, Veleučilište Velika Gorica

Sažetak

Agilne metode razvoja softvera nastale su kako bi poboljšale organizaciju projektnih timova, te pridonijele efikasnosti razvojnih inženjera s obzirom na klasične metode organizacije koje se temelje na „vodopadnom" (engl. waterfall) načinu razvoja softverskih rješenja.

Najpoznatija agilna metodologija je „Scrum" i temelji se na organiziranju projekta na osnovne vremenske periode koji se nazivaju „sprintevima". Na početku svakog „sprinta" svaki član razvojnog tima dobiva svoj skup zadataka koje mora riješiti do kraja „sprinta", kako bi razvojni projekt završio na vrijeme. Zadaci su najčešće organizirani tako da traju do maksimalno dva radna dana, pri čemu se složeniji zadaci raščlanjuju da više jednostavnih zadataka koji su međusobno zavisni. Svim aktivnostima tijekom cijelog projekta i pojedinog „sprinta" koordinira „Scrum master" koji odlučuje koji zadaci će se ispunjavati u pojedinom „sprintu" i tko će ih određivati.

Da bi se „Scrum" metodologija mogla maksimalno iskoristiti, potrebno je koristiti prikladnu programsku opremu prilagođenu za iskorištavanje njezinih prednosti. Alat tvrtke Microsoft „Microsoft Project 2010" nudi mogućnost proširenja dodatkom „Microsoft Project 2010 Scrum Solution Starter" koji sadrži sve ključne segmente za organizaciju projekata po „Scrum" metodologiji, te se konstantno razvija.

Zbog svoje široke prihvaćenosti alat „Microsoft Project 2010" je vrlo pogodan za korištenje u edukaciji novih naraštaja voditelja projekata kako bi koristili najnovije metodologije u razvoju softvera. Kroz definiranje detalja projekta kao što su „Product backlog", „Sprint backlog", te generiranje „Burndown" izvještaja, omogućeno je planiranje i praćenje napretka projekta od njegovih početaka pa sve do završetka planirane implementacije u zacrtanim vremenskim rokovima.

Uvod

Agilne metode razvoja softvera razlikuju se od ostalih tradicionalnih metoda po tome da uvode nove uloge stručnjaka u projektni tim i drugačije načine organizacije aktivnosti tijekom implementacije softverskog produkta[1].

Za razliku od nekoliko uloga i profila kod tradicionalnog načina upravljanja projektima (stručnjaci za marketing, upravljanje produktom i upravljanje projektom, koji zajedno

dijele odgovornost za produkt), agilne metode predviđaju samo jednu osobu za koja je zadužena za produkt i projekt: vlasnika produkta (engl. *Product owner*).

Kod tradicionalnih metoda stručnjaci za upravljanje produktom odvojeni su od razvojnog tima, dok agilne metode podrazumijevaju da su vlasnici produkta članovi „Scrum" tima te usko surađuju sa „Scrum" majstorom (engl. *Scrum Master*) i ostatkom tima.

Analiza tržišta, planiranje produkta i poslovna analiza se kod tradicionalnih metoda odrađuju prije samog početka implementacije produkta, dok se kod agilnih metoda odrađuje samo minimalna količina posla za kreiranje vizije o budućem produktu.

Definicija zahtjeva za funkcionalnostima se donose unaprijed i često se ne smiju mijenjati sve do kraja implementacije, a kod agilnih metoda se ta faza proteže kroz cijelu implementaciju. Agilne metode ne podrazumijevaju fazu definicije produkta i specificiranja zahtjeva. Lista zadataka projekta (engl. *Product backlog*), dokument koji sadrži popis svih aktivnosti i zadataka koje je potrebno obaviti u sklopu projekta, je vrlo dinamičan i znatno ovisi povratnim informacijama korisnika i klijenata.

Kod tradicionalnih metoda povratne informacije od korisnika stižu u kasnijim fazama razvoja i testiranja produkta nakon njegovog lansiranja, dok se korištenjem agilnih metoda rano izdaju nove verzije produkta. Nakon svakog perioda implementacije (engl. *sprint*), kao faze na koje je podijeljeno trajanje projekta, organiziraju se sastanci kako bi se od korisnika dobile povratne informacije o produktu.

„Scrum" kao ogledan primjer agilne metodologije temelji se na principu promjene kao jedine konstante u dinamičnim procesima kao što je razvoj softvera. U ovom radu opisan je prijedlog podučavanja „Scrum" metodologije korištenjem alata Microsoft Project 2010.

Scrum

„Scrum" je utemeljen na teoriji empirijske kontrole procesa, odnosno, tvrdnjama da znanje dolazi iz iskustva i odlučivanja temeljenih na poznatom. „Scrum" koristi iterativni inkrementalni pristup radi optimizacije predvidivosti i kontrole rizika[2].

Empirijska kontrola procesa temeljena je transparentnosti, kontroli (engl. *inspection*) i prilagodbi (engl. *adaptation*). Transparentnost se temelji na tome da ključni aspekti moraju biti vidljivi onima koji su odgovorni za krajnji proizvod. Korisnici „Scruma" često moraju kontrolirati artefakte (liste zadataka projekta i liste zadataka perioda implementacije) i napredak prema cilju radi uočavanja neželjenih odstupanja, na čemu se temelji kontrola, dok se prilagodba odnosi korekciju proizvoda ukoliko je to nužno radi minimiziranja odstupanja[2].

Kontrola i prilagodba

„Scrum" propisuje četiri formalne prilike za kontrolu i prilagodbu: sastanak planiranja perioda implementacije, dnevni „Scrum" sastanak, revizija perioda implementacije i retrospektiva perioda implementacije[2].

Period implementacije (engl. *sprint*) je srce „Scruma" i predstavlja vremenski ograničen period od jednog mjeseca ili kraći tijekom kojeg proizvede „završen", upotrebljiv i potencijalno isporučiv inkrement proizvoda. Inkrement je zbroj svih stavaka s liste zadataka projekta (engl. *Product backlog*) završenih tijekom perioda implementacije i svih prethodnih perioda implementacije. Dnevni „Scrum" sastanak je 15-minutni, vremenski ograničen događaj, koji služi da razvojni tim uskladi aktivnosti i donese plan za sljedeća 24 sata. Revizija perioda implementacije održava se na kraju samog perioda implementacije radi kontrole inkrementa i prilagođavanja liste zadataka projekta, ako je potrebno. Retrospektiva perioda implementacije je prilika za članove „Scrum" tima da kontroliraju sebe i donesu plan za unapređenja koja će se provesti tijekom sljedećeg perioda implementacije[2].

Članovi „Scrum" tima i artefakti „Scruma"

„Scrum" tim sastoji se od vlasnika proizvoda (engl. *Product Owner*), razvojnog tima (engl. *Development Team*) i „Scrum" majstora (engl. *Scrum Master*)[2].

Vlasnik proizvoda odgovoran je za maksimizaciju vrijednosti proizvoda i rada razvojnog tima, te upravljanje listom zadataka projekta. Lista zadataka projekta je artefakt „Scruma" koji sadrži sve što će možda biti potrebno za proizvod i jedini je izvor zahtjeva za bilo kakvim promjenama koje se rade na proizvodu[2].

Osim liste zadataka projekta u sklopu „Scruma" postoji i artefakt lista zadataka perioda implementacije, koji predstavlja skup stavaka koje su odabrane za period implementacije zajedno s planom realizacije inkrementa i cilja perioda implementacije. Za praćenje i procjenu napretka koriste se tzv. grafovi sagorijevanja (engl. *Burndown charts*), koji prikazuju procjenu napretka perioda implementacije.

Razvojni tim se sastoji od profesionalaca koji rade konkretan posao vezan uz potencijalno isporučiv inkrement proizvoda na kraju svakog perioda implementacije. Strukturirani su i ovlašteni da organiziraju i vode vlastiti posao. Optimalna veličina „Scrum" tima je od tri do devet članova, kako bi se izbjeglo premalo interakcije, nedostatak vještine, ali i previše utrošenog vremena na interakciju[2].

„Scrum" majstor je odgovoran da se „Scrum" razumije i da se koristi, što se postiže pridržavanjem teorije, prakse i pravila „Scruma".

Priprema projekta za implementaciju po „Scrumu"

Svaki softverski projekt koji se želi implementirati po principima „Scruma" na početku se mora sastojati od faze sastavljanja liste zadataka projekta, koja će se trebati obaviti do kraja projekta. Prilikom podučavanja studenata je vrlo bitno držati se pravila da u listi zadataka projekta moraju biti zadaci koji moraju biti rješivi unutar jednog do dva radna dana, koji se kasnije mogu raščlaniti i na manje zadatke kad se prikupe dodatne informacije o detaljima samih zadataka. Na primjer, ako je potrebno implementirati jednostavnu web aplikaciju „Studomat" u programskom jeziku Java koristeći MVC (engl. *Model View Controller*) arhitekturu s programskim okvirom Hibernate, inicijalna inačica liste zadataka projekta može izgledati ovako:

Tablica 1. Popis zadataka u inicijalnoj listi zadataka projekta

R.broj	Zadaci unutar inicijalne liste zadataka projekta
1.	ER MODEL BAZE PODATAKA
2.	KREIRANJE TABLICA U BAZI PODATAKA
3.	KREIRANJE JAVA DOMENSKIH KLASA
4.	KREIRANJE HIBERNATE "MAPPING"-A ZA DOMENSKE KLASE
5.	JAVA "DAO" SLOJ ZA PRISTUPANJE BAZI PODATAKA
6.	JUNIT TESTOVI ZA TESTIRANJE "DAO" SLOJA
7.	JAVA "SERVISNI" SLOJ
8.	JUNIT TESTOVI ZA JAVA "SERVISNI" SLOJ

9.	DIZAJN TOKA EKRANA
10.	IMPLEMENTACIJA "CONTROLLER" KLASA
11.	JUNIT TESTOVI ZA METODE U "CONTROLLER" KLASAMA
12.	DIZAJN I IMPLEMENTACIJA EKRANA
13.	FUNKCIONALNO TESTIRANJE EKRANA
14.	PISANJE DOKUMENTACIJE ZA APLIKACIJU

U tablici 1. prikazan je popis svih zadataka koje je potrebno napraviti kako bi projekt završio. Za obavljanje svih zadataka potrebno je nekoliko profila stručnjaka pa će se zadaci morati podijeliti u nekoliko kategorija: „DATABASE", „DEVELOPMENT", „TEST", „DESIGN" i „DOCUMENTATION". Ako je na raspolaganju „Scrum" timu osim vlasnika produkta i „Scrum" majstora, po jedan administrator baze podataka, programer, tester, dizajner grafičkog sučelja i stručnjak za sastavljanje dokumentacije, studentima se može postaviti zadatak da sastave projektni plan i odrede vrijeme potrebno za dovršenje projekta.

Što se tiče procjena vremena potrebnih za dovršavanje pojedinog zadatka, u tablici 2. prikazan je popis zadataka s njihovim trajanjima. Za manji dio zadataka potrebna su dva radna dana, a za ostale zadatke po jedan radni dan. Ako je maksimalno trajanje jednog perioda implementacije deset radnih dana, od studenata se očekuje da će zaključiti da je za dovršenje projekta potrebno organizirati dva perioda implementacije.

Poštujući kronološki redoslijed aktivnosti i međuovisnosti zadataka, te uzevši u obzir različite profile stručnjaka za pojedine zadatke, moguće je sastaviti sljedeću tablicu:

Tablica 2.Tablica s detaljnijim podacima o zadacima

R.br.	Naziv zadatka	Traj.	Početak	Kraj	Ovis.	Profil
1.	ER MODEL BAZE PODATAKA	1 dan	29.05.12	29.05.12		DB Admin
2.	KREIRANJE TABLICA U BAZI PODATAKA	1 dan	30.05.12	30.05.12	1	DB Admin
3.	KREIRANJE JAVA DOMENSKIH KLASA	1 dan	31.05.12	31.05.12	2	Programer
4.	KREIRANJE HIBERNATE "MAPPING"-A ZA DOMENSKE KLASE	1 dan	01.06.12	01.06.12	3	Programer
5.	JAVA "DAO" SLOJ ZA PRISTUPANJE BAZI PODATAKA	2 dana	04.06.12	05.06.12	4	Programer
6.	JUNIT TESTOVI ZA TESTIRANJE "DAO" SLOJA	1 dan	06.06.12	06.06.12	5	Tester
7.	JAVA "SERVISNI" SLOJ	1 dan	06.06.12	06.06.12	5	Programer
8.	JUNIT TESTOVI ZA JAVA "SERVISNI" SLOJ	1 dan	08.06.12	08.06.12	7	Tester
9.	DIZAJN TOKA EKRANA	1 dan	12.06.12	12.06.12		Dizajner
10.	IMPLEMENTACIJA "CONTROLLER" KLASA	1 dan	13.06.12	13.06.12	9	Programer

11.	JUNIT TESTOVI ZA METODE U "CONTROLLER" KLASAMA	1 dan	14.06.12	14.06.12	10	Tester
12.	DIZAJN I IMPLEMENTACIJA EKRANA	2 dana	14.06.12	15.06.12	10	Programer
13.	FUNKCIONALNO TESTIRANJE EKRANA	2 dana	18.06.12	19.06.12	12	Tester
14.	PISANJE DOKUMENTACIJE ZA APLIKACIJU	2 dana	20.06.12	21.06.12	13	Osoba za pisanje dokumentacije

Detalji u tablici 2. prikazuju pojedini zadatak i njegovo trajanje, dan početka i završetka implementacije, ovisnost prema ostalim zadacima, te profil stručnjaka koji je zadužen za njihovu implementaciju. Pretpostavljeno je da implementacija počinje 29. svibnja, a završava 21. lipnja 2012. godine. Prvih osam zadataka potrebno je izdvojiti u prvi period implementacije, a preostale zadatke potrebno je organizirati u drugi period implementacije.

Bitno je primijetiti da se zadaci numerirani brojevima 6 i 7, te 11 i 12 mogu izvršavati paralelno, jer su za njihovu implementaciju zaduženi različiti profili stručnjaka.

Svaki period implementacije se osim navedenih implementacijskih zadataka mora sastojati i od zadataka planiranja perioda implementacije, revizije perioda implementacije i retrospektive perioda implementacije, koje je potrebno uključiti u projektni plan na početku, odnosno, na kraju svakog perioda implementacije.

Korištenje alata Microsoft Project 2010 za „Scrum" projekt

Instaliranje dodatka „Scrum Solution Starter"

Microsoft Project 2010 u osnovnoj inačici nema izravnu podršku za organiziranje projekata po agilnoj metodi „Scrum", ali preuzimanjem dodatka „Scrum Solution Starter" s Microsoftovih mrežnih stranica (http://archive.msdn.microsoft.com/P2010Scrum), te instalacijom istog moguće je u potpunosti prilagoditi alat toj namjeni.

Instaliranjem dodatka unutar alata pojavljuje se izbornik „Scrum" koji nudi osnovne opcije kreiranja artefakata liste zadataka projekta, liste zadataka perioda implementacije, kreiranje novih perioda implementacije, prikazivanje ključnih datuma u projektu i generiranje grafova sagorijevanja za praćenje napretka projekta.

Korištenje dodatka „Scrum Solution Starter"

Opcija „Product backlog" unutar izbornika „Scrum" omogućava unošenje podataka iz tablice 2. u kontekstu kreiranja istoimenog artefakta koji će sadržavati sve zadatke koje je potrebno ispuniti kako bi projekt bio završen. Na slici 1. prikazan je dio ekrana koji prikazuje kako izgleda popis zadataka s nekim od detalja koje je moguće definirati u sklopu „Product backlog" opcije:

	Category	Deliverable	Sprint	Task Status	Item Type	Add New Column
3	SCRUM TEAM	PLANIRANJE PRVOG SPRINTA	1	Not Started	Product Backlog	
4	DATABASE	ER MODEL BAZE PODATAKA	1	Not Started	Product Backlog	
5	DATABASE	KREIRANJE TABLICE U BAZI PODATAKA	1	Not Started	Product Backlog	
6	DEVELOPMENT	KREIRANJE JAVA DOMENSKIH KLASA	1	Not Started	Product Backlog	
7	DEVELOPMENT	KREIRANJE HIBERNATE "MAPPING"-A ZA DOMENSKE KLASE	1	Not Started	Product Backlog	
8	DEVELOPMENT	JAVA "DAO" SLOJ ZA PRISTUPANJE BAZI PODATAKA	1	Not Started	Product Backlog	
9	TESTING	JUNIT TESTOVI ZA TESTIRANJE "DAO" SLOJA	1	Not Started	Product Backlog	
10	DEVELOPMENT	JAVA "SERVISNI" SLOJ	1	Not Started	Product Backlog	
11	TESTING	JUNIT TESTOVI ZA JAVA "SERVISNI" SLOJ	1	Not Started	Product Backlog	
12	SCRUM TEAM	REVIZIJA PRVOG SPRINTA	1	Not Started	Product Backlog	
13	SCRUM TEAM	RETROSPEKTIVA PRVOG SPRINTA	1	Not Started	Product Backlog	
14	SCRUM TEAM	PLANIRANJE DRUGOG SPRINTA	2	Not Started	Product Backlog	
15	DESIGN	DIZAJN TOKA EKRANA	2	Not Started	Product Backlog	
16	DEVELOPMENT	IMPLEMENTACIJA "CONTROLLER" KLASA	2	Not Started	Product Backlog	
17	TESTING	JUNIT TESTOVI ZA METODE U "CONTROLLER" KLASAMA	2	Not Started	Product Backlog	
18	DEVELOPMENT	DIZAJN I IMPLEMENTACIJA EKRANA	2	Not Started	Product Backlog	
19	TESTING	FUNKCIONALNO TESTIRANJE EKRANA	2	Not Started	Product Backlog	
20	DOCUMENTATION	PISANJE DOKUMENTACIJE ZA APLIKACIJU	2	Not Started	Product Backlog	
21	SCRUM TEAM	REVIZIJA DRUGOG SPRINTA	2	Not Started	Product Backlog	
22	SCRUM TEAM	RETROSPEKTIVA DRUGOG SPRINTA	2	Not Started	Product Backlog	

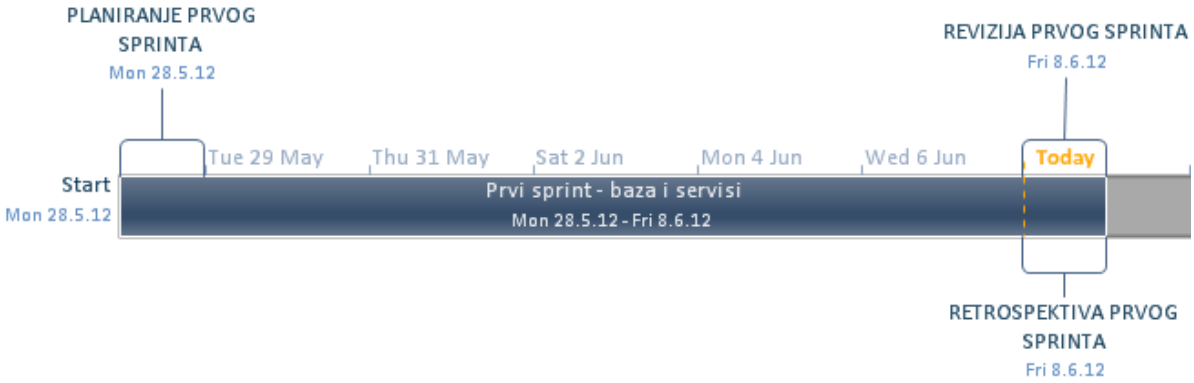
Slika 1. Izgled ekrana za kreiranje „Product backlog" artifakta „Scrum"-a

Promjenom opcije „Item Type" na „Sprint Backlog" moguće je prebacivati pojedine zadatke u jedan od perioda implementacije koji se mogu definirati pomoću „Add New Sprint" opcije u glavnom izborniku „Scrum". Opcija „Add New Column" omogućava dodavanje novih detalja o zadacima.

Odabirom opcije „Key Dates and Milestones" moguće je definirati ključne datume u fazi implementacije projekta, odnosno periode implementacije (slika 2.), a unutar njih su smješteni svi ostali događaji koji su navedeni na slici 1. Grafički „Timeline" prikaz događaja u prvom periodu implementacije zajedno sa „Scrum" događajima prikazan je na slici 3.

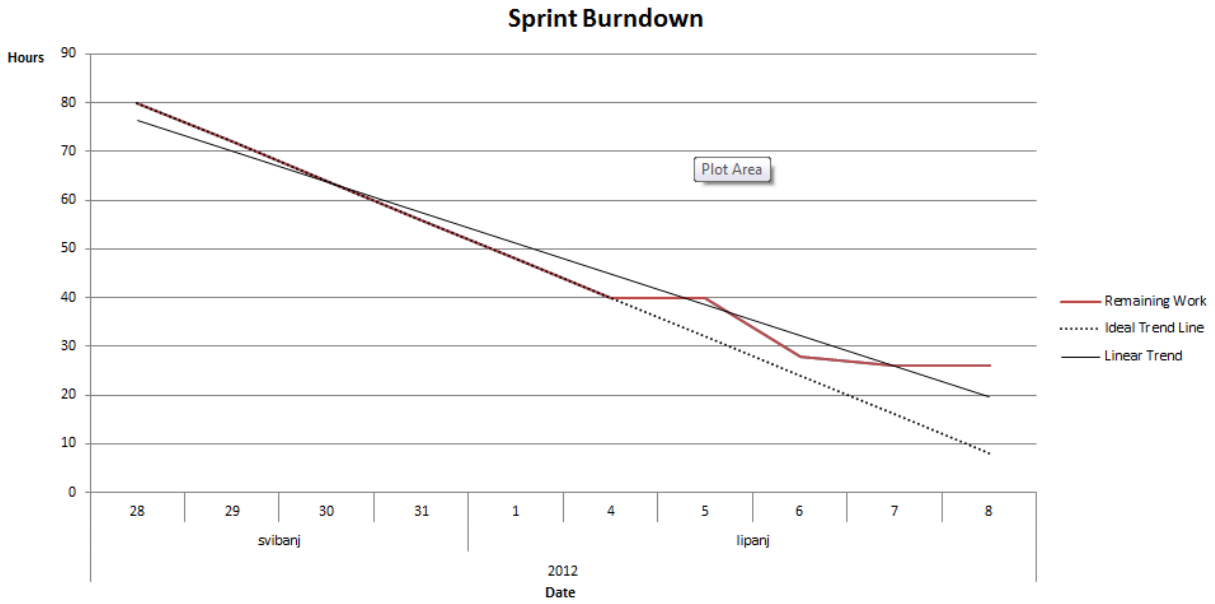
	Name	Sprint	Start	Finish	Duration	Add New Column
1	Prvi sprint - baza i servisi	1	Mon 28.5.12	Fri 8.6.12	10 days	
2	Drugi sprint - ekrani	2	Mon 11.6.12	Fri 22.6.12	10 days	

Slika 2. Dio ekrana koji prikazuje ključne datume



Slika 3. Grafički prikaz ključnih događaja u sklopu prvog perioda implementacije

Tijekom svakog perioda implementacije moguće je pratiti napredak izvršavanja zadataka pomoću grafova sagorijevanja. Taj graf omogućava uvid u odnos završenih zadataka s obzirom na planirani i idealni trend rješavanja zadataka. Primjer grafa sagorijevanja prikazan je na slici 4. Crtkana linija na slici 4. Označava idealni trend rješavanja zadataka (broja radnih sati za njihovo izvršavanje), puna crna linija označava linearan trend rješavanja zadataka s obzirom na stvarno rješavanje zadataka, a puna crvena linija označava preostali broj sati koji je potrebno odraditi do kraja prvog perioda implementacije.



Slika 4. Primjer grafa sagorijevanja

Zaključak

Microsoft Project 2010 je jedan od najčešće korištenih alata koje koriste voditelji projekata prilikom planiranja, a s dodatkom „Scrum Solution Starter" u potpunosti je prilagođen provođenju agilnih metodologija organiziranja projekata. Kako mnoge obrazovne institucije raspolažu besplatnim licencama za Microsoftove proizvode, vrlo je pogodan za korištenje u obrazovanju novih naraštaja voditelja projekata koji nastoje pratiti i koristiti najnovije svjetske trendove i metodologije vođenja projekata.

Popis literature

1. Pichler, R. *Agile Product Management with Scrum*, Boston, 2010.
2. Križ, Z. Vodič za Scrum, Zagreb, 2011.