Programsko orodje za podporo projektnemu delu študentov

Anže Časar in dr. Viljan Mahnič

Fakulteta za računalništvo
in informatiko Univerze v Ljubljani

Na Fakulteti za računalništvo in informatiko Univerze v Ljubljani že več let poučujemo metodo Scrum, ki je najbolj razširjena agilna metoda za razvoj programske opreme. Učenje poteka v obliki skupinskega dela na projektu, tako da se študenti spoznajo z metodo ob reševanju (skoraj) realnega problema. Za vodenje projektov smo razvili lastno orodje, ki študentom olajša izvajanje projekta, pedagoškemu osebju pa omogoča sprotno spremljanje učnega procesa in doseženih rezultatov. Orodje smo preizkusili v praksi, rezultati pa so zelo spodbudni.

Uvod

Ker je področje agilnega razvoja programske opreme sorazmerno novo in slabo raziskano – sistematičen pregled empiričnih študij s tega področja (Dybå in Dingsøyr, 2008) je odkril samo 33 študij, med katerimi pa je le ena obravnavala Scrum – predstavljajo zaključni študentski projekti priložnost za preizkus in empirično ovrednotenje novih metod razvoja, za kar v industriji običajno ni dovolj časa in denarja. Zato smo pri zasnovi predmeta Tehnologija programske opreme upoštevali tako priporočila organizacij IEEE in ACM (Lethbridge et al., 2006) kot priporočila za uporabo študentov pri raziskovalnem delu (Carver et al., 2005; Carver et al., 2010). S tako zasnovanim predmetom želimo zadovoljiti cilje vseh udeležencev: študentov, učnega osebja in raziskovalcev, posredno pa tudi podjetij za razvoj programske opreme. Študente želimo seznaniti z najnovejšimi metodami in jim posredovati znanja, ki so potrebna v praksi. Učnemu osebju želimo omogočiti uporabo sodobnih pedagoških pristopov, ki temeljijo na učenju skozi reševanje praktičnih problemov. Raziskovalcem pa tako zasnovan predmet omogoča, da z zbiranjem empiričnih podatkov postavljajo in preverjajo posamezne hipoteze. Doseganje navedenih ciljev sproti merimo z anketami med študenti (Mahnič, 2010), empirične podatke, zbrane med izvajanjem projektov, pa smo uporabili v okviru več študij s področja ocenjevanja in planiranja softverskih projektov (Mahnič, 2011; Mahnič in Hovelja, 2012). Za planiranje in vodenje študentskih projektov je potrebno ustrezno orodje, ki po eni strani pomaga študentom pri izvajanju projekta, po drugi strani pa omogoča beleženje podatkov, ki jih učno osebje in raziskovalci potrebujejo za spremljanje učnega procesa in predvidene raziskave. Sprva smo v ta namen uporabljali prosto dostopno različico orodja Agilo for Scrum, ki pa se je izkazala za precej okorno, zato smo se odločili za izdelavo lastnega orodja, ki ga bomo podrobneje opisali v nadaljevanju tega prispevka. Za boljše razumevanje bomo najprej predstavili, kako potekajo študentski projekti. Nato bomo opisali zasnovo orodja in njegovo funkcionalnost s stališča podpore, ki jo orodje nudi študentom med izvajanjem projekta. Nazadnje pa bomo predstavili še možnosti, ki so na voljo učnemu osebju.

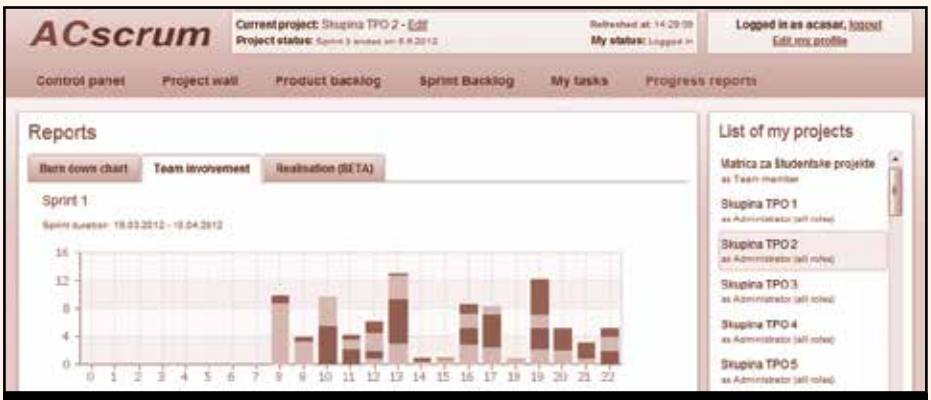
Potek študentskih projektov

Študenti morajo delati v skupinah in razviti (skoraj) realen projekt na podlagi uporabniških zahtev, ki jih pripravi poznavalec problemske domene, ki nastopa v vlogi produktnega vodje. V prvih treh tednih se na predavanjih seznanijo z metodo Scrum (Schwaber, 2004) in uporabo uporabniških zgodb (Cohn, 2004) za dokumentiranje zahtev in planiranje projekta. V tem času morajo oblikovati 4-članske skupine in pripraviti razvojno okolje, v katerem bodo razvili zahtevano aplikacijo. Vsaka skupina dobi seznam zahtev, ki vsebuje množico po prioriteti razvrščenih uporabniških zgodb. Skupina mora oceniti zahtevnost posameznih zgodb po metodi Planning Poker (Grenning, 2002), oceniti svojo predvideno hitrost in izdelati plan izdaje. Preostanek predmeta je razdeljen na tri iteracije, ki trajajo po 4 tedne. Vsaka iteracija se prične s sestankom za načrtovanje iteracije, na katerem se študenti s produktnim vodjo dogovorijo, katere uporabniške zgodbe bodo razvili v naslednji iteraciji, in izdelajo seznam nalog, ki so potrebne za realizacijo teh zgodb. Med iteracijo se vse skupine redno sestajajo na t. i. Daily Scrum sestankih ter vzdržujejo podatke o opravljenem in preostalem delu na posameznih nalogah. Ker imajo študenti tudi druge obveznosti, ti sestanki ne potekajo vsak dan (kot zahteva metoda Scrum), ampak dvakrat tedensko: enkrat v okviru rednega pedagoškega procesa, enkrat pa morajo sestanek organizirati sami. Na koncu vsake iteracije sta organizirana sestanka za pregled rezultatov in oceno kakovosti razvojnega procesa. Na sestanku za pregled rezultatov vsaka skupina predstavi uporabniške zgodbe, ki jih je realizirala v pravkar končani iteraciji. Na sestanku za oceno kakovosti razvojnega procesa pa vsaka skupina analizira dobre in slabe strani svojega dela ter se dogovori za izboljšave, ki jih bo vpeljala v naslednji iteraciji. Po treh iteracijah mora biti projekt končan in predan naročniku. V skladu z metodo Scrum so vse vloge natančno definirane. Študenti nastopajo kot člani razvojnih skupin, ki so kolektivno odgovorne za realizacijo dogovorjene funkcionalnosti. Učno osebje nastopa v vlogi skrbnika metodologije, ki zagotavlja, da delo vseh skupin poteka v skladu s pravili metode Scrum. To vlogo dodatno dodelimo še enemu študentu iz vsake skupine, ki mora skrbeti za redno izvajanje vsakodnevnih sestankov ter pravilen in pravočasen vnos vseh podatkov. Produktni vodja je lahko nekdo od učnega osebja (če je projekt definiran znotraj fakultete) ali pa nekdo iz industrije (če projekt definira podjetje za razvoj programske opreme, ki sodeluje s fakulteto). Njegova naloga je, da vzdržuje seznam uporabniških zgodb, daje pojasnila razvijalcem in na koncu vsake iteracije preveri, ali izdelane rešitve res ustrezajo vsem zahtevam.

Zasnova orodja

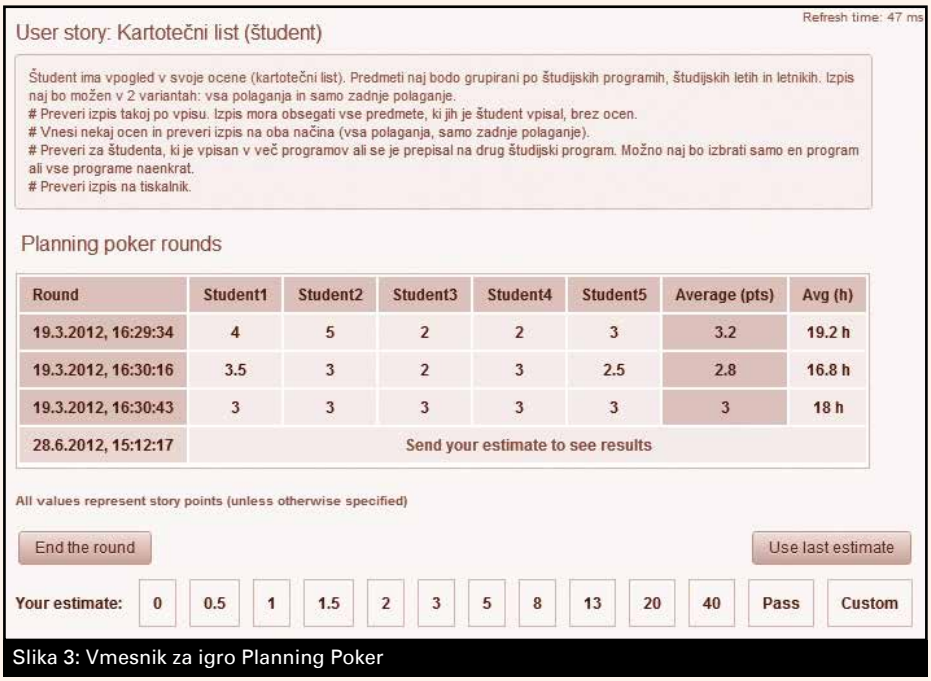
Orodje, ki smo ga razvili za podporo izvajanju opisanega projektnega dela, je zasnovano kot spletna aplikacija, ki vsem uporabnikom omogoča oddaljen dostop do podatkov. Uporabniški vmesnik je prikazan na sliki 1. Za realizacijo so bile izbrane odprtokodne tehnologije (PHP, MySQL, jQuery), kar je omogočilo prosto uporabo razvite rešitve brez dodatnih stroškov.

Z orodjem lahko vodimo več projektov istočasno, vsak uporabnik pa ima dostop samo do tistih projektov, pri katerih sodeluje. Uporabniške vloge v aplikaciji so enake, kot jih predvideva metoda Scrum, vsak uporabnik pa ima lahko v okviru istega projekta eno ali več vlog. Dodatno je bila uvedena še vloga administrator, katere nosilci imajo vpogled v vse projekte s pravicami vseh prej omenjenih vlog. Vlogo administratorja imajo pri našem



Slika 1: uporabniški vmesnik orodja

Slika 2: Primer tipične uporabniške zgodbe



Slika 3: Vmesnik za igro Planning Poker

predmetu člani učnega osebja, študenti pa imajo vlogo razvijalcev, pri čemer ima po en študent iz vsake skupine tudi pravice skrbnika metodologije. Pri izdelavi orodja smo upoštevali zahtevo, da mora le-to podpirati vse postopke in dokumente, ki jih zahteva metoda Scrum, zato je orodje uporabno za vodenje kateregakoli projekta, ki poteka po tej metodi. Poleg tega smo vpeljali tudi nekatere unikatne izboljšave, ki razvojni skupini še dodatno olajšajo delo, učnemu osebju in raziskovalcem pa omogočajo vpogled v delo vseh študentov in raznovrstne možnosti statistične obdelave podatkov.

Vzpostavitev projektov

Projekt se vzpostavi tako, da se v aplikacijo vnese podatke o vseh članih projektne skupine in se jim dodeli ustrezne vloge. Produktni vodja nato kreira začetni seznam zahtev, tako da vsako zahtevo opiše v obliki uporabniške zgodbe in ji določi prioriteto. Zgodbe imajo v naši aplikaciji obliko kartic, primer katere je prikazan na sliki 2. Po prioriteti so razdeljene v štiri kategorije: (1) zgodbe, ki so obvezen sestavni del naslednje izdaje, (2) zgodbe, ki jih tudi želimo imeti v naslednji izdaji, a lahko namesto njih začasno uporabimo kakšno drugo zasilno rešitev, (3) zgodbe, ki niso nujne, a izboljšajo končno rešitev, in (4) zgodbe, ki jih bomo uvrstili v naslednjo izdajo. S tem želimo študente usmeriti k čim prejšnji realizaciji bolj pomembnih zgodb, hkrati pa takšna realizacija tako po formi kot po vsebini ustreza zahtevam metode. Pred začetkom dela mora skrbnik metodologije v orodje vnesti še število in trajanje iteracij ter pričakovano hitrost razvojne skupine za vsako posamezno iteracijo. S tem določi časovni okvir za dokončanje trenutne izdaje produkta.

Načrtovanje iteracij

Na začetku vsake iteracije morajo člani razvojne skupine oceniti vse uporabniške zgodbe z uporabo metode Planning Poker. V našem orodju je izvedba ocenjevanja v celoti računalniško podprta, kar predstavlja unikatno prednost pred konkurenčnimi izdelki. Planning Poker, katerega vmesnik je prikazan na sliki 3, poteka v več iteracijah. Ko skrbnik metodologije odpre glasovanje za določeno zgodbo, jo lahko vsak član razvojne skupine oceni na svojem računalniku s klikom na gumb z ustreznim številom točk. Ena točka predstavlja neko vnaprej dogovorjeno časovno enoto; običajna (in tudi naša) izbira je en popoln delovni dan, oziroma 6 ur. Ko so vse ocene oddane, skrbnik metodologije na svojem računalniku zaključi igro in (če so ocene enotne) rezultat sprejme kot veljavno oceno zgodbe. V nasprotnem primeru pa odpre razpravo in od članov, ki sta dala najvišjo in najnižjo oceno, zahteva ustrezno utemeljitev. Ta postopek ponavlja, dokler se ocene ne poenotijo.

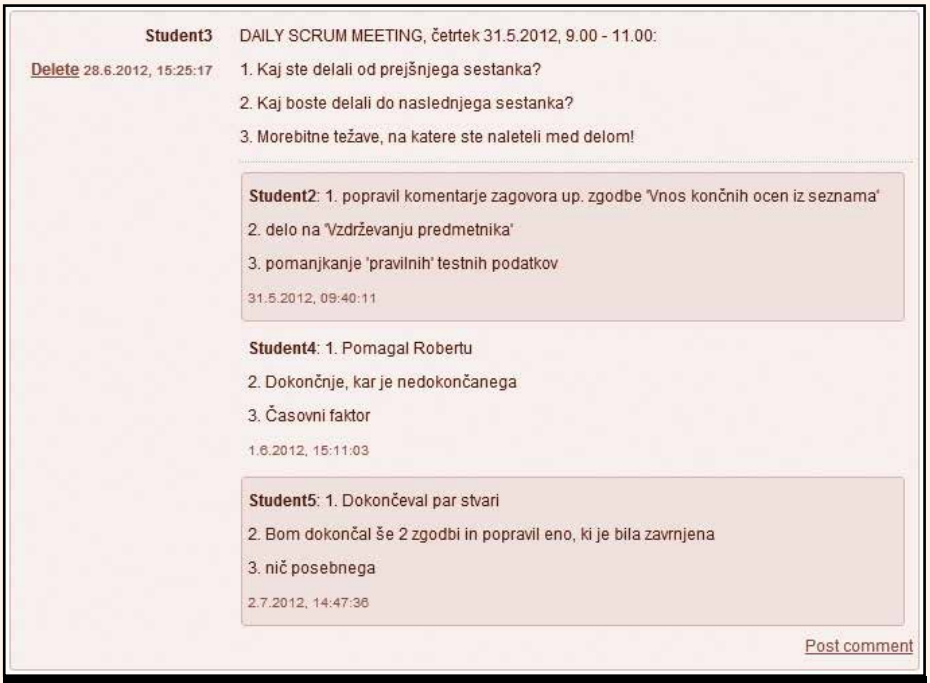
Vsaka skupina nato v aplikaciji izbere zgodbe, ki jih je namerava realizirati v okviru naslednje iteracije, pri čemer upošteva oceno hitrosti za naslednjo iteracijo. Program v vsakem trenutku postopka nudi informacijo o skupni vrednosti že izbranih zgodb (v točkah) in o predvideni hitrosti ter s tem pomaga skupini pri postopku izbire. Vse izbrane zgodbe skupaj oblikujejo t.i. Sprint Backlog, ki je samostojen razdelek v aplikaciji. Tu mora skupina vse zgodbe razdeliti na naloge, ki jim določi opis, časovno zahtevnost (v urah) in člane skupne, ki so odgovorni za njihovo realizacijo. Vsak član skupine mora

dogovorjene naloge še sprejeti v delo in si s tem zgraditi seznam svojega dela za tekočo iteracijo. S tem se planiranje iteracije zaključi in člani razvojne skupine lahko pričnejo z delom na nalogah.

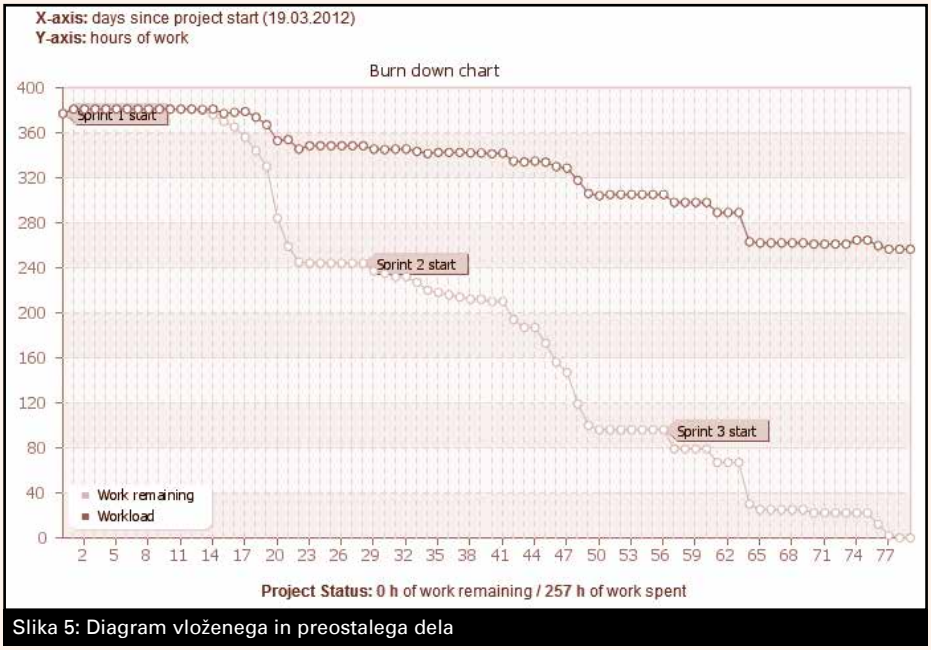
Vodenje podatkov o delu

Vsa orodja, ki podpirajo delo po metodi Scrum, običajno omogočajo vodenje vloženega in preostalega časa za vsako od nalog v Sprint Backlogu. Razvijalec mora običajno ta čas ročno vnašati v orodje, kar je lahko dokaj zamudno opravilo, poleg tega pa so vnesene vrednosti mnogokrat zgolj približne.

V našem orodju smo to opravilo poenostavili. Omogočili smo samodejno beleženje vloženega časa, kar je realizirano na način, da razvijalec sproti označuje, na katerih nalogah trenutno dela, aplikacija pa vloženi čas v ozadju ves čas samodejno sešteva. Seveda ima razvijalec še vedno možnost, da naknadno popravi samodejno zabeleženi čas, poleg tega pa mora ob koncu delovnega dne v skladu z metodo Scrum še ročno vnesti čas, ki mu je preostal do dokončanja vsake izmed nalog. Ta podatek namreč predstavlja bistveno informacijo za spremljanje napredka na projektu.



Slika 4: Primer objave rezultatov rednega vsakodnevnega sestanka na zidu za komunikacijo



Slika 5: Diagram vloženega in preostalega dela

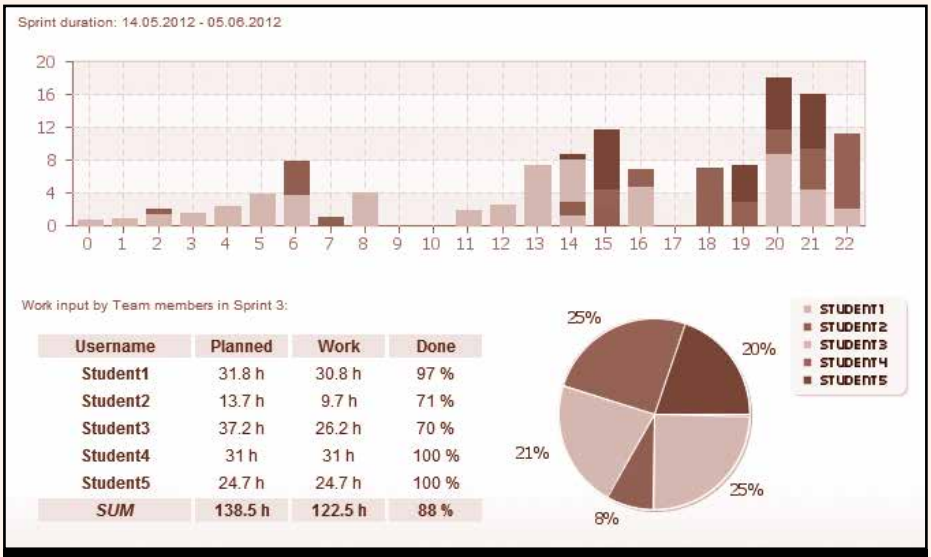
V orodje smo vgradili tudi komunikacijski vmesnik, ki je po funkcionalnosti podoben zidu priljubljenega družabnega omrežja Facebook. Namenjen je medsebojni komunikaciji skupine med delom na projektu, poleg tega pa ga lahko za objavo obvestil uporablja tudi učno osebje. Vsaki projektni skupini je na razpolago lasten komunikacijski zid, kjer lahko objavljajo, komentirajo in pregledujejo le objave, ki se tičejo njihovega projekta. Prav tako lahko skupine na zid objavljajo tudi zabeležke rednih vsakodnevnih sestankov, kot je prikazano na sliki 4. Na ta način tudi učno osebje pridobi vpogled v sprotno dogajanje na projektu, razvijalci pa lahko kadarkoli pregledujejo zaveze preteklih sestankov in preverijo njihovo uresničevanje.

Zaključek iteracije

Ob zaključku vsake iteracije produktni vodja določi, katere zgodbe so bile pravilno realizirane. Le povsem pravilno realizirane zgodbe namreč lahko sestavljajo končni izdelek, medtem ko morajo biti vse ostale zgodbe zavrnjene. Produktni vodja v takšnih primerih v aplikacijo vnese podroben opis napak, razvojna ekipa pa mora te zgodbe v naslednji iteraciji popraviti. Pri tem se za zavrnjene zgodbe v skladu z zasnovo orodja ohrani vsa zgodovina opravljenega dela, preteklih časovnih ocen in tudi vse zgodbi pripadajoče naloge, katerim je moč naknadno dodati tudi nove. Na ta način lahko skupina isto zgodbo prenaša iz iteracije v iteracijo, vse dokler njena realizacija ni popolna. Po zaključku vseh predvidenih iteracij je pri našem predmetu na sporedu še končna predstavitev izdelkov. Vsak projekt mora predstavljati celovito rešitev, v katero so integrirane vse uporabniške zgodbe. Ta predstavitev je tudi eden od kriterijev za določitev končnih ocen članov posameznih skupin. Ostali ocenjevalni kriteriji vključujejo še: realizacijo celotne skupine, realizacijo posameznikov, rednost opravljanja Daily Scrum sestankov in kakovost planiranja posameznih iteracij. Vse te podatke je moč pridobiti s pomočjo aplikacije, kar postopek ocenjevanja bistveno poenostavi.

Uporaba orodja s stališča učnega osebja

Učno osebje ima od aplikacije koristi predvsem s stališča sprotnega vpogleda v delo študentov in dostopa do statističnih podatkov o delu. Ti podatki po eni strani omogočajo lažje in bolj objektivno končno ocenjevanje, po drugi strani pa so uporabni tudi v namen



Username	Planned	Work	Done
Student1	31.8 h	30.8 h	97 %
Student2	13.7 h	9.7 h	71 %
Student3	37.2 h	26.2 h	70 %
Student4	31 h	31 h	100 %
Student5	24.7 h	24.7 h	100 %
SUM	138.5 h	122.5 h	88 %

Slika 6: Diagram razporeditve dela znotraj ene iteracije

Realisation by Stories and Team Members

#	Story name	Student4 (Mojca Četrta)	Student2 (Jože Drugi)	Student5 (Marija Peta)	Student1 (Janez Prvi)	Student3 (Lojze Tretji)
1	Odjava izpita (študent)	2 pt	-	-	-	-
2	Prijava na izpit (študent)	6 pt	-	-	-	-
3	Vpisni list (referentka)	-	-	-	-	5 pt
4	Izbirni predmeti (referentka)	-	-	2 pt	-	-
5	Iskanje študenta	-	-	2 pt	-	-
6	Seznam za [illegible]	-	-	-	-	-
15	Prijava v sistem	-	-	-	3 pt	-
16	Osebni podatki študenta	-	-	-	-	4 pt
17	Sprememba izpitnega roka	-	-	-	4 pt	-
18	Brisanje izpitnega roka	-	-	-	3 pt	-
SUM (pt)		14 pt	12.36 pt	12 pt	22.64 pt	9 pt
SUM (%)		20 %	17.66 %	17.14 %	32.34 %	12.86 %

Realisation by Priority

Priority	All points	Realized points	% Realized	% SUM
Must have	85 pt	59 pt	69.41 %	69.41 %
Should have	16 pt	11 pt	68.75 %	69.31 %
Could have	6 pt	0 pt	0 %	65.42 %

Slika 7: Realizacija po posameznih članih skupine

raznovrstnih analiz in raziskav. Naša aplikacija omogoča vodenje velike količine podatkov, izmed katerih je nekaj tudi takšnih, ki so specifični za študijski proces, četudi jih metoda Scrum ne zahteva eksplicitno. V skladu z metodologijo je namreč potrebno za vsako nalogo voditi čas, ki je še preostal do njene realizacije, za boljši pregled dela študentov pa je zelo koristno voditi tudi čas, ki je bil vložen v posamezno nalogo. Ta čas lahko še nadalje kategoriziramo na:

- vložen čas celotne skupine,
- vložen čas posameznega študenta,
- realizacijo (število točk) celotne skupine,
- realizacijo (število točk) posameznika.

Aplikacija omogoča pregledovanje vloženega časa po vseh štirih kriterijih. Vloženi čas celotne skupine je razviden iz diagrama na sliki 5, ki prikazuje razmerje med vloženim in preostalim delom skozi čas. Nižja krivulja na diagramu prikazuje preostalo delo v posameznem trenutku procesa, višja pa skupno količino dela skozi čas (vsoto preostalega in že vloženega dela). Ta krivulja ni vedno konstantna, ker se časovne ocene zgodb in nalog skozi čas spreminjajo, kar seveda vpliva na količino skupnega dela. Pod diagramom je podan tudi kumulativni podatek o vloženem in preostalem delu na projektu v urah. Za namen spremljanja vloženega dela po posameznih članih skupine orodje ponuja diagram, kjer je za vsak dan trajanja projekta za vsakega člana grafično razviden čas njegovega dela. Primer je prikazan na sliki 6. S pomočjo te preglednice lahko učno osebje spremlja, ali prihaja pri delegiranju dela na projektu do kakšnih neenakomernosti, prav tako pa lahko tudi vsak član skupine primerja količino svojega dela z delom svojih kolegov. Iz omenjenega diagrama je razvidna tudi sprotnost vnašanja delovnih ur v aplikacijo, kar je eden bistvenih pogojev za uspešen potek procesa in jo je v sorodnih orodjih za vodenje projektov po metodi Scrum običajno težko spremljati.

Morda še najpomembnejši podatek, ki priča tako o uspešnosti skupine kot celote, kot tudi o uspešnosti posameznih članov v njej, je realizacija skupine. Naša aplikacija ponuja pregledovanje realizacije po več kriterijih: po zgodbah, po prioriteti in po članih skupine. V statistiko so zajete le tiste zgodbe, ki so bile potrjene s strani produktnega vodje, rezultati pa so zbrani v preglednici, katere primer je prikazan na sliki 7.

Ker je vsaka zgodba ocenjena z nekim številom točk, ki so bile določene s soglasjem vseh članov skupine, lahko te ocene smatramo kot dobro osnovo za statistično obdelavo realizirane količine dela in za objektivno določitev prispevkov posameznikov k delu celotne skupine. Podatek je namreč precej bolj reprezentativen kot npr. količina vloženih ur, saj lahko nek član, ki v programiranju ni tako vešč, za realizacijo iste zgodbe porabi več časa, kot bi zanjo porabil nek bolj izkušen član skupine. Na podlagi preglednice je moč utežiti tudi pričakovano količino realizacije posameznih skupin, če se med seboj razlikujejo po številu članov. Tako lahko od 4-članskih skupin zahtevamo zgolj realizacijo obveznih zgodb z najvišjo prioriteto, od 5-članskih pa tudi realizacijo ostalih zgodb, ki jih želimo v naslednji izdaji, ne glede na to, da zanje obstaja nadomestna zasilna rešitev. Največje skupine morajo realizirati vse zgodbe, tudi tiste, ki niso nujne, ampak samo izboljšajo delovanje aplikacije. Tovrstno utežitev smo pri ocenjevanju rezultatov uporabili tudi mi.

Zaključek

Naš primer dokazuje, da je za izvajanje študija pri predmetih, ki temeljijo na skupinskem

projektnem delu, potrebno zagotoviti ustrezno računalniško podporo. S tem dosežemo več ciljev: študentom olajšamo delo na projektu, učnemu osebju pa omogočimo vpogled v napredek študentov, dobimo objektivno podlago za določitev končnih ocen in pridobimo statistične podatke, ki jih lahko uporabimo v raziskovalne namene. Naše orodje je zadostilo vsem tem ciljem, saj so bili z njim zadovoljni vsi udeleženci, kar smo ob zaključku predmeta potrdili tudi z anketo, ki smo jo izvedli med študenti. Ker se naše orodje osredotoča na funkcionalnosti, ki so potrebne predvsem za pedagoški proces, smo z njegovo pomočjo pridobili precej boljšo sliko o delu posameznih skupin. Zato smo jih lahko pravočasno opozarjali na morebitne napake in ovire v procesu, pridobljeni podatki o delu pa so posledično bolj realni in predstavljajo dobro osnovo za nadaljnje analize. Zaradi pozitivnih izkušenj v prihodnje načrtujemo nadaljnji razvoj orodja tudi na podlagi odziva, ki smo ga prejeli od študentov. Z njegovo uporabo bomo vsekakor nadaljevali tudi v prihodnjih letih, kasneje pa ga bomo morda ponudili tudi preostalim inštitucijam na trgu. •

Literatura

- Carver, J., Jaccheri, L., Morasca, S. , in Shull, F., (2005): »Issues in using students in empirical studies in software engineering education,« Proc. 9th Int. Software Metrics Symp., Sydney, Australia, str. 239–249.
- Carver, J. C., Jaccheri, L., Morasca, S., in Shull, F., (2010): »A checklist for integrating student empirical studies with research and teaching goals,« Empirical Software Eng., let. 15, str. 35–59.
- Cohn, M., (2004): User Stories Applied For Agile Software Developement, Addison – Wesley.
- Dybå, T. in Dingsøyr, T., (2008): »Empirical studies of agile software development: A systematic review,« Inf. and Software Technology, let. 50, št. 9-10, str. 833–859
- Grenning, J., (2002): »Planning poker or how to avoid analysis paralysis while release planning,« http://renaissancesoftware.net/files/articles/PlanningPoker-v1.1.pdf
- Lethbridge, T. C., LeBlanc Jr., R. J., Kelley Sobel, A. E. in Díaz-Herrera, J. L., SE2004, (2006): »Recommendations for Undergraduate Software Engineering Curricula,« IEEE Softw., let. 23, št. 6, str. 19–25.
- Mahnič, V. (2010): »Teaching Scrum through team-project work: students' perceptions and teacher's observations,« Int. J. of Eng. Educ., let. 26, št. 1, str. 96–110.
- Mahnič, V., (2011): »A case study on agile estimating and planning using Scrum,« Electronics and Electrical Engineering, št. 5(111), str. 123-128.
- Mahnič, V., (2012): »A capstone course on agile software development using Scrum,« IEEE Transactions on Education, let. 55, št. 1, str. 99-106.
- Mahnič, V., Hovelja, T., (2012): »On using planning poker for estimating user stories,« J. Syst. Software, let. 85, str. 2086-2095.
- Murphy, T., Duggan, J., Norton, D., Prentice, B., Plummer, D., Landry, S., (2009): »Predicts 2010: Agile and Cloud Impact Application Development Directions,« Gartner, http://www.gartner.com/DisplayDocument?id=1244514
- Schwaber, K., (2004): Agile Project Management with Scrum, Redmond, WA, Microsoft Press.