UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Janez Urevc

Agilni razvoj časopisnega portala po metodi Scrum

DIPLOMSKO DELO
NA UNIVERZITETNEM ŠTUDIJU

Mentor: prof. dr. Viljan Mahnič

Ljubljana, 2012

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01813/2012
Datum: 15.03.2012

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **JANEZ UREVC**

Naslov: **AGILNI RAZVOJ ČASOPISNEGA PORTALA PO METODI SCRUM**
AGILE DEVELOPMENT OF A NEWS PORTAL USING SCRUM

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Opišite uvajanje metode Scrum v postopek razvoja spletnega portala enega izmed najpomembnejših slovenskih dnevnih časopisov. Uvodoma predstavite metodo Scrum in razloge za njen izbor. Nato opišite priprave na uvedbo, potek projekta in pridobljene izkušnje. Posebno pozornost namenite analizi točnosti ocenjevanja uporabniških zgodb in primerjajte ocene zahtevnosti, pridobljene po metodi planning poker, z aritmetično srednjo vrednostjo ocen posameznih razvijalcev. Na koncu predstavite še mnenja uporabnikov o metodi Scrum na podlagi ankete med udeleženci projekta in drugimi uporabniki v Sloveniji in tujini.

Mentor:

prof. dr. Viljan Mahnič

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Namesto te strani **vstavite** original izdane teme diplomskega dela s podpisom mentorja in dekana ter žigom fakultete, ki ga diplomant dvigne v študentskem referatu, preden odda izdelek v vezavo!

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani Janez Urevc,

z vpisno številko 63040171,

sem avtor diplomskega dela z naslovom:

Agilni razvoj časopisnega portala po metodi Scrum

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom
  prof. dr. Viljana Mahniča
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 01.06.2012 Podpis avtorja:

Zahvala

Zahvaljujem se mentorju prof. dr. Viljanu Mahniču za nasvete, pomoč, podporo in potrpežljivost. Zahvala gre tudi Roku Štebetu, Taji Topolovec in vsem ostalim sodelavcem v oddelku spletnega razvoja.

Na koncu se zahvaljujem čisto vsem, ki so kakorkoli pripomogli k uspešnemu dokončanju moje študijske poti.

Mojim najdražjim

Kazalo

Seznam izrazov, specifičnih za metodo Scrum

- *Daily Scrum meeting* - vsakodnevni sestanek za pregled poteka del na projektu
- *Product Backlog* - seznam zahtev (množica uporabniških zgodb)
- *Product Owner* - produktni vodja
- *Release Plan* - plan izdaje
- *Scrum Team* - razvojna skupina
- *ScrumMaster* - skrbnik metodologije
- *Sprint* - iteracija
- *Sprint Backlog* - seznam nalog, potrebnih za realizacijo posameznih uporabniških zgodb
- *Sprint planning meeting* - sestanek za načrtovanje iteracije
- *Sprint retrospective meeting* - sestanek za oceno kakovosti razvojnega procesa
- *Sprint review meeting* - sestanek za predstavitev rezultatov iteracije
- *User story* - uporabniška zgodba
- *Velocity* - hitrost razvoja (število točk, ki jih razvojna skupina lahko realizira v eni iteraciji)

Povzetek

V diplomskem delu obravnavamo projekt uvedbe metode Scrum v oddelek spletnega razvoja medijske hiše. Uvedbo smo pilotno izvajali na projektu prenove večjega medijskega portala. V okviru diplomskega dela predstavimo razloge za uvedbo, opišemo njene podrobnosti, izpostavimo težave in njihove rešitve.

Posebno poglavje namenjamo analizi točnosti ocenjevanja uporabniških zgodb, saj gre pri tem za enega ključnih faktorjev uspešnega vodenja projekta. Predstavljamo tudi rezultate raziskave, ki smo jo o metodi Scrum izvedli med udeleženci projekta in ostalimi uporabniki metode.

Ključne besede:

Scrum, agilne metodologije, Planning Poker, ocenjevanje zahtevnosti, uporabniška zgodba, Drupal

Abstract

The thesis considers the Scrum implementation project in the web development department of the bigger Slovenian publishing house. The pilot implementation studied in this paper was carried out as part of a redesign project for a major media portal. The reasons for implementing Scrum are considered and the details of the implementation described, highlighting the problems encountered and their solutions.

A separate chapter is dedicated to the analysis of accuracy of evaluation of user stories, as these are one of the key factors for the successful management of a project. The results of a survey regarding the Scrum method carried out with the participants of the project and other users of the method are also presented.

Key words:

Scrum, agile methodologies, Planning Poker, level of difficulty evaluation, user story, Drupal

Poglavje 1

Uvod

V okviru diplomske naloge smo spremljali uvedbo agilne metode Scrum v delovni proces oddelka spletnega razvoja podjetja Delo d.d. Uvedba je bila pilotno izpeljana na projektu prenove in migracije medijskega portala *slovenskenovice.si*.

Obstoječ portal je bil postavljen na platformi, ki je bila pred leti razvita interno v podjetju. Zaradi lažjega razvoja in vzdrževanja je bila sprejeta odločitev, da se vsa spletna mesta podjetja Delo postopoma migrira na prosto kodno platformo Drupal.

Cilj projekta, ki ga opisujemo, je bilo prav to. Obstoječ spletni portal je bilo potrebno prenesti na novo platformo in ga hkrati obogatiti z novimi funkcionalnostmi. Pri tem je bilo potrebno paziti tako na uporabniško izkušnjo bralcev portala, kot tudi novinarjev, ki portal uporabljajo za objavo vsebin. Potrebno je bilo tudi slediti najnovejšim trendom in tehnologijam, ki se danes pojavljajo na internetu.

1.1 Okoliščine in razlogi za izbor teme

Tema za diplomsko nalogo je bila izbrana predvsem na podlagi dejstva, da se je načrtovanje projekta, ki ga opisujemo v nalogi, dogajalo ravno v času, ko je avtor naloge razmišljal tudi o temi svojega diplomskega dela. V prid temi je govorilo tudi dejstvo, da pred tem v razvojni skupini spletnega oddelka podjetja Delo ni bila uporabljana nobena metodologija. Takšen pristop je deloval pri manjših projektih, medtem ko za projekt večjega obsega takšen pristop ni bil primeren.

V prid temi diplomske naloge govori tudi dejstvo, da je šlo za večji projekt, ki je potekal skozi daljše časovno obdobje, govorilo v prid temi diplomske na-

loge. Daljši projekt je namreč lažje spremljati in analizirati, saj lahko sledimo dogajanju skozi več faz. Tako lahko ugotavljamo, kako se skozi čas spreminjajo razpoloženje, dojemanje metode, točnost ocenjevanja nalog ipd.

Projekt je bil primeren za vodenje po metodi Scrum tudi zaradi dejstva, da je šlo za spletni projekt, kjer so že po definiciji specifikacije velikokrat podane zelo ohlapno in se med samim razvojem pogosto spreminjajo. To od razvojne ekipe zahteva veliko mero agilnosti, za kar pa naj bi bil Scrum zelo primeren.

1.2 Razširjenost agilnih metodologij

Scrum spada v družino agilnih metod razvoja programske opreme, ki se v industriji vse bolj uveljavljajo. Agilne metodologije so se formalno pojavile leta 2001 [15], ko se pojavi *Manifest za agilni razvoj programske opreme*. Manifest predpostavlja, da tradicionalne inžinerske metode niso primerne za razvoj programske opreme, saj se ta preveč razlikuje od področij, ki imajo daljšo tradicijo.

Tradicionalne metode so veliko poudarka dajale pisanju obsežne dokumentacije in izdelavi načrtov, saj s tem poizkušajo vzpostavljati nadzor nad izvajanjem projekta. Takšno nefleksibilno okolje je primerno za bolj "konkretne" projekte, kot jih na primer zasledimo v arhitekturi, strojništvu, gradbeništvu, ipd. Razvoj programske opreme se od omenjenih področij razlikuje predvsem po tem, da je okolje, v katerem delujemo veliko bolj dinamično in spremenljivo. To se odraža tudi na programski opremi, saj se zahteve in okoliščine zelo pogosto spreminjajo. Na te spremembe se s tradicionalnimi metodami seveda ne moremo pravočasno odzvati.

Ker to lahko dosežemo s pomočjo agilnih metod, se slednje vse bolj uveljavljajo v celotnem spektru industrije razvoja programske opreme. Sprva je veljalo, da so agilne metode primerne predvsem za manjša podjetja oz. skupine, a se tudi to, kot bomo videli v nadaljevanju, že spreminja. Vse več velikih podjetij se danes odloča za uporabo agilnih metod, saj uspejo aktivno zadovoljevati tudi svoje potrebe.

Agilne metode se najbolj izkažejo pri projektih, kjer nimamo jasnih specifikacij projekta, kjer je zahtevana velika stopnja fleksibilnosti in čim hitrejše prilagajanje novim okoliščinam v okolju. Primer takšnih projektov so ravno spletni in mobilni projekti, saj so ravno ta področja industrije trenutno najaktualnejša in se posledično najhitreje razvijajo.

Kot smo omenili že v uvodu, smo portal razvijali na temelju platforme Drupal. Okoli projekta Drupal se je v zadnjih desetih letih oblikovala sorazmerno

velika skupnost posameznikov in podjetij, ki platformo skupaj razvijajo, uporabljajo in tržijo. V skupnosti, katere del smo tudi mi, so agilne metode skoraj standard, kar še dodatno dokazuje naše trditve. Ker se v skupnosti znanje deli zelo nesebično, smo imeli dostop tudi do izkušenj, ki so jih pri uvedbi Scruma imela ostala podjetja. Te informacije smo lahko s pridom izkoristili tudi pri svojem projektu.

1.3 Primernost projekta

Agilne metode oz. Scrum so bile za naš projekt primerne iz več razlogov. Šlo je za spletni projekt, ki je imel na začetku zelo ohlapno definirane zahteve. Pričakovati je bilo možno, da se bodo zahteve med potekom projekta pogosto spreminjale.

V prid Scrumu je govorilo tudi dejstvo, da smo sodelovali z naročnikom, ki je bil dovolj izobražen in se je bil pripravljen dovolj angažirati pri projektu. To je zelo pomembno, saj odsotnost dokumentacije in specifikacij večinoma nadomeščamo prav z intenzivno komunikacijo z naročnikom.

V nadaljevanju bomo najprej v grobem opisali agilne metodologije in metodo Scrum. V tretjem poglavju bomo bralcu predstavili projekt, njegove posebnosti, okoliščine in začetne zahteve naročnika. V četrtem poglavju bo sledil podroben opis poteka projekta, kjer bomo opisali kako smo projekt izvajali v praksi. Prav tako bomo omenili zanimive posebnosti, težave in njihove rešitve. V petem poglavju bo sledila analiza ocenjevanja zahtevnosti nalog, kjer bomo predstavili način ocenjevanja in preverili, kako uspešni so bili pri tem vidiku projekta. V predzadnjem poglavju bomo predstavili rezultate raziskave, ki smo jo izvedli med udeleženci projekta in jo nato razširili še na širšo množico uporabnikov metode Scrum. Diplomsko delo bomo zaokrožili z zaključkom, v katerem bomo povzeli vse ugotovitve in opažanja.

Poglavje 2

Opis agilnih metodologij

2.1 Agilne metodologije

Agilne metode za razvoj programske opreme so se začele pojavljati v devetdesetih letih prejšnjega stoletja, kot reakcija na tradicionalen discipliniran pristop, ki zahteva natančno vnaprejšnje načrtovanje in predpisuje točno določen postopek razvoja. V nasprotju s tem **Manifest za agilni razvoj programske opreme** [2] daje prednost posameznikom in interakciji med njimi pred postopki in orodji, delujoči programski opremi pred obsežno dokumentacijo, sodelovanju z naročnikom pred pogajanji o pogodbi in odzivanju na spremembe pred sledenjem načrtu. S tem postavlja v drugi plan nekatere vrednote, ki so bolj značilne za tradicionalne metode.

Kljub temu, da med razvijalci programske opreme in informacijskih sistemov ni enotnega mnenja o primernosti agilnih metod, vse več podatkov kaže na njihovo uspešnost in naraščajočo popularnost. Tako Ambler [1] navaja, da uporaba agilnih metod bistveno poveča produktivnost, kakovost in zadovoljstvo vseh udeležencev v razvojnem procesu. Forrester v svojem poročilu [19] ugotavlja, da je v letu 2009 uporabljalo agilne metode že 35% projektov, po Gartnerjevi napovedi [11] pa naj bi leta 2012 kar 80% projektov potekalo na agilen način. Čeprav prevladuje mnenje, da so agilne metode primerne predvsem za manjše projekte, lahko v literaturi zasledimo poročila o uvajanju teh metod tudi v največjih podjetjih s področja informacijske tehnologije, kot so npr. Microsoft [3], Yahoo [14], Nokia [7] ipd.

Scrum [13] je med vsemi agilnimi metodami najbolj razširjen, saj njegov delež po zadnjih podatkih za leto 2011 znaša 52%, v kombinaciji z ekstremnim programiranjem pa 66% [18]. Tudi v Sloveniji se krog uporabnikov te metode vedno bolj širi; skupina Skram.si v omrežju LinkedIn šteje preko 200 članov.

2.2 Metoda Scrum

Scrum izhaja iz premise, da je razvoj programske opreme preveč zapleten in nepredvidljiv proces, da bi ga lahko natančno planirali vnaprej. Namesto tega je treba vzpostaviti empiričen nadzor, ki omogoča sproten vpogled in prilagajanje trenutnemu stanju. To lahko dosežemo z iterativnim in inkrementalnim pristopom.

Metoda Scrum pozna kar nekaj specifičnih terminov, za katere še ne obstajajo splošno sprejeti prevodi v slovenščino. V tem prispevku bomo uporabljali prevode, ki jih predlaga [8]. Prevodi so navedeni tudi v razdelku "*Seznam uporabljenih kratic in simbolov*".

Več o metodi Scrum je možno prebrati v [13] in [17].

2.2.1 Vloge v metodi Scrum

Za vzpostavitev tega procesa Scrum predvideva tri vloge: produktni vodja *(angl. Product Owner)*, razvojna skupina *(angl. Scrum Team)* in skrbnik metodologije *(angl. ScrumMaster)*. Produktni vodja je predstavnik naročnika in zastopa vse, ki so zainteresirani za rezultate projekta. Njegova osnovna naloga je vzdrževanje seznama zahtev *(angl. Product Backlog)*, določanje njihove prioritete in grupiranje zahtev v posamezne izdaje *(angl. release)*. Seznam zahtev ni nikoli dokončen, ampak se lahko ves čas projekta dopolnjuje. Vsaka zahteva je praviloma opisana v obliki uporabniške zgodbe *(angl. user story)*, ki vsebuje besedilo zgodbe in seznam sprejemnih testov. Ker gre za zelo ohlapen opis zahteve, mora biti produktni vodja stalno na voljo razvijalcem za pojasnila glede podrobnosti, povezanih z realizacijo. Po vsaki iteraciji pa mora preveriti, ali je zgodba ustrezno realizirana ali ne. Ta vloga je ključnega pomena za uspešnost projekta, zato mora imeti produktni vodja jasno vizijo o tem, kaj je treba narediti, in skladno s tem usmerjati razvojni proces.

Razvojna skupina je zadolžena za implementacijo zahtevane funkcionalnosti. Sestavljena mora biti tako, da pokriva vsa področja, ki so pomembna za izvedbo projekta. Skupina deluje po načelu samoorganizacije in sama določa, kako na najboljši način realizirati posamezne zahteve. Člani razvojne skupine so kolektivno odgovorni za uspeh posameznih iteracij in projekta kot celote.

Skrbnik metodologije ima navidez enako vlogo kot vodja projekta, vendar so njegove zadolžitve v resnici drugačne. Njegova naloga je skrbeti za nemoten potek projekta, tako da vsi, ki delajo na projektu, upoštevajo pravila metode Scrum in na ustrezen način izvajajo predpisane aktivnosti. Pri tem mora paziti, da se Scrum vklaplja v kulturo organizacije tako, da daje najboljše

rezultate. Skrbnik metodologije v veliki meri deluje kot varuh, ki ščiti razvojno skupino pred škodljivimi zunanjimi vplivi, odpravlja morebitne ovire in s tem zagotavlja optimalne pogoje za delo.

2.2.2 Potek razvojnega procesa po metodi Scrum

Na začetku projekta produktni vodja izdela seznam zahtev, ki vsebuje vse do takrat znane uporabniške zgodbe. Zgodbe razvrsti po prioriteti in jih razdeli v predvidene izdaje. Razvojna skupina oceni zahtevnost vsake zgodbe z ustreznim številom točk *(angl. story points)* in določi predvideno hitrost razvoja *(angl. velocity)*, tj. število točk, ki jih lahko realizira v eni iteraciji. V skladu s tem nato izdela plan izdaje *(angl. Release Plan)*, tako da v skladu s prioriteto razporedi zgodbe po posameznih iteracijah. Seštevek točk vseh zgodb v neki iteraciji ne sme preseči predvidene hitrosti razvoja.

Nadaljnji razvoj poteka v iteracijah *(angl. Sprint)*, ki v originalni verziji metode Scrum trajajo 30 dni, danes pa se največ uporabljajo iteracije, ki so dolge 2 ali 4 tedne. Vsaka iteracija se začne s sestankom za načrtovanje iteracije *(angl. Sprint planning meeting)*, na katerem se produktni vodja in razvojna skupina dogovorijo, katere zgodbe bodo realizirane v naslednji iteraciji.

Na podlagi tega razvojna skupina izdela seznam nalog *(angl. Sprint Backlog)*, ki vsebuje vse naloge, potrebne za realizacijo dogovorjene funkcionalnosti do konca iteracije. Med samo iteracijo se ta seznam še dopolnjuje, obsežnejše naloge pa se razdelijo na več manjših, tako da vsaka zahteva od 4 do 16 ur dela.

Člani razvojne skupine se vsak dan zberejo na 15-minutnem sestanku *(angl. Daily Scrum meeting)*, na katerem vsak izmed njih odgovori na 3 vprašanja: "Kaj si naredil od prejšnjega sestanka? Kaj boš delal do naslednjega sestanka? Ali imaš pri delu kakšne težave?" Namen tega sestanka je sinhronizirati delo vseh članov razvojne skupine in sproti identificirati morebitne probleme.

Na koncu vsake iteracije je predpisan sestanek za pregled rezultatov *(angl. Sprint review meeting)*, na katerem razvojna skupina predstavi rezultate svojega dela produktnemu vodji in vsem zainteresiranim uporabnikom. Metoda striktno zahteva, da razvijalci spoštujejo koncept "done", kar pomeni, da mora biti vsaka zgodba v celoti realizirana in dokumentirana, tako da jo je moč neposredno predati v produkcijo. Produktni vodja sme sprejeti samo tiste zgodbe, ki v celoti ustrezajo omenjenemu konceptu, in samo seštevek točk teh zgodb se lahko upošteva kot dejanska hitrost razvoja *(angl. actual velocity)* razvojne skupine.

Po tem sestanku (in pred pričetkom naslednje iteracije) skrbnik metodolo-

gije organizira še sestanek za oceno kakovosti razvojnega procesa *(angl. Sprint retrospective meeting)*, katerega namen je poiskati možnosti za izboljšave razvojnega procesa, tako da bi bil ta še bolj učinkovit v naslednjih iteracijah.

Poglavje 3

Predstavitev projekta

3.1 Projekt prenove spletnega portala slovenskenovice.si

Pri projektu je v osnovi šlo za prenovo, nadgradnjo in migracijo obstoječega spletnega mesta. Spletno mesto se je v okviru projekta iz lastnega sistema migriralo na platformo Drupal 7.

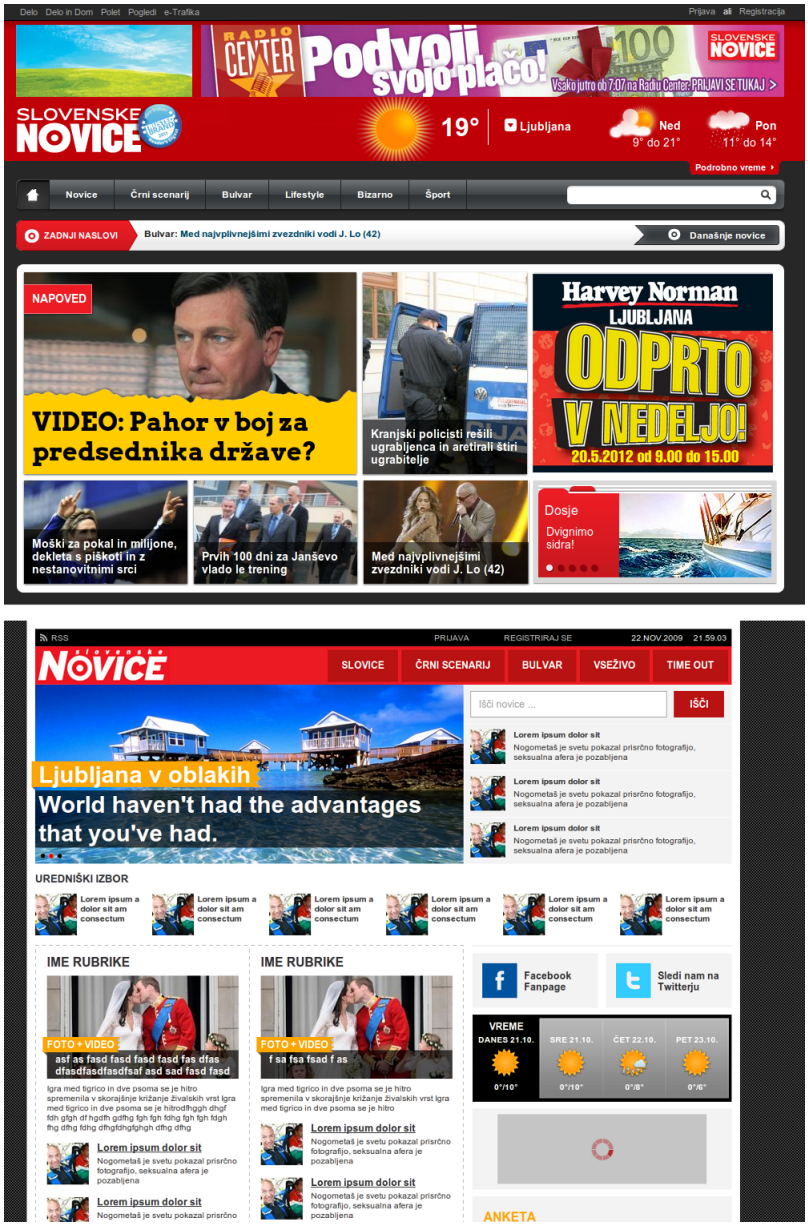
Naročnik, ki je bil v našem primeru interne narave, je imel na začetku projekta dokaj jasno definirane osnovne želje, medtem ko podrobnosti samih funkcionalnosti niso bile dorečene. Pričakovati je bilo možno, da se bodo podrobnosti kristalizirale tekom projekta, kar pa je seveda občasno povzročilo tudi spreminjanje bolj osnovnih zahtev.

3.2 Platforma Drupal

Drupal[1] je odprto-kodno ogrodje oz. sistem za gradnjo spletnih strani. Napisan je v jeziku PHP, ki poganja velik del spletnih strani in aplikacij. Gre za sistem, ki se razvija že več kot 10 let in se vse bolj uveljavlja predvsem pri večjih projektih. Na njem temeljijo tako večji medijski portali, kot tudi vladna spletna mesta več držav, velike spletne trgovine, ipd. Gre za zelo zmogljiv in fleksibilen sistem, ki pa je prav zaradi tega za razvijalce nekoliko bolj zahteven od konkurenčnih produktov.

Največja prednosti sistema je prav gotovo velika skupnost, ki se je izoblikovala okoli projekta. Gre za veliko množico posameznikom in podjetij, ki s

[1] Več informacij o projektu Drupal je na voljo na spletnem naslovu *http://drupal.org*.



Slika 3.1: Primerjava novega (zgoraj) in starega (spodaj) portala *slovenskenovice.si*.

skupnimi močmi razvijajo in seveda tudi uporabljajo Drupal. Ker veliko teh uporabnikov uporablja Drupal za velike spletne projekte, se je v skupnosti izoblikovala baza znanja, ki pokriva prav to področje. Pri velikih projektih namreč naletimo na nekatere probleme, ki jih pri manjših ne zasledimo (performančne omejitve, odzivnost, stabilnost, ...). Tudi to znanje je govorilo v prid Drupala, saj so nam izkušnje iz skupnosti dejansko zelo pomagale pri premagovanju nekaterih težav.

3.3 Zahteve naročnika

Naročnikove zahteve so bile usmerjene predvsem v konkurenčnost portala na tržišču in optimizacijo delovnega procesa na strani naročnika. Portal je moral biti v prvi meri dovolj ličen in privlačen za uporabnika, da bi se ta nanj vračal. Njegovo administracijsko okolje je moralo biti intuitivno, jasno in čim bolj jasno za urednika. Zaželeno je bilo, da se urednikovo delo z njegovo pomočjo čim bolj časovno optimizira. Zaradi tega smo želeli administracijsko okolje zastaviti tako, da je urednik za najbolj pogoste naloge potreboval čim manj klikov oz. časa.

Portal je moral biti tudi dobro optimiziran za spletne iskalnike, saj velika večina obiskovalcev novice išče prav preko tega vira. Da bi uporabnika, ki je iz tega vira prišel na portal, obdržali, jim je ta moral omogočati soustvarjanje vsebin. Iz tega razloga je bila zahtevana možnost komentiranja, oddaje čestitk in člankov, pošiljanje pripomb in predlogov ipd.

Poglavje 4

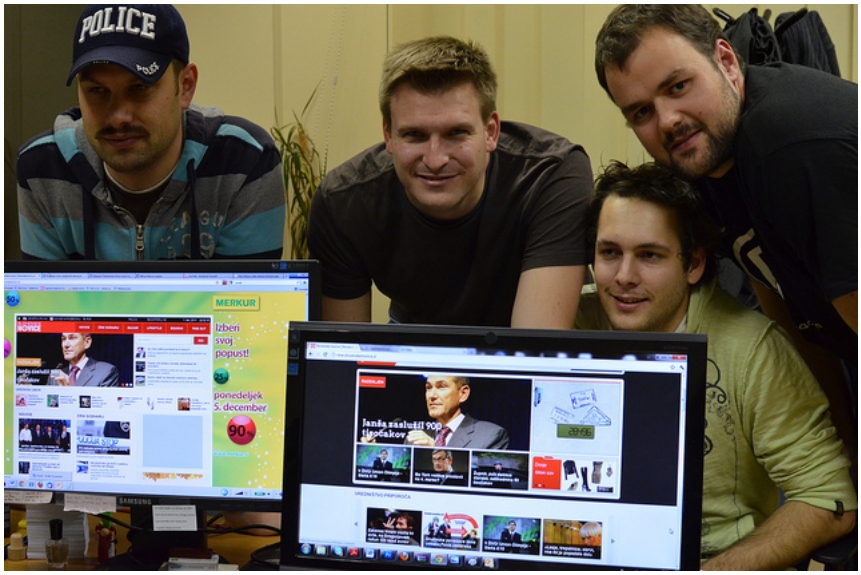
Potek projekta

4.1 Tehnični in administrativni vidiki uvedbe metode Scrum

V okviru priprav na uvedbo metode Scrum smo najprej pregledali izkušnje podobnih razvojnih skupin. Kot smo že omenili, portal, ki smo ga razvijali, temelji na odprtokodni platformi Drupal, ki ima veliko in povezano skupnost, v kateri je kar nekaj podjetij, ki uporabljajo metodo Scrum. Njihove izkušnje smo še prav posebej preučili, oprli pa smo se tudi na priporočila, ki so predstavljena v literaturi. Na podlagi tega smo posebno pozornost namenili razdelitvi vlog, definiciji koncepta "done", določitvi dolžine iteracije, izbiri ustreznega orodja za spremljanje poteka dela, ocenjevanju zahtevnosti posameznih zgodb in izobraževanju vseh sodelujočih.

4.1.1 Razdelitev vlog

V metodi Scrum nastopajo 3 vloge: produktni vodja (angl. Product Owner), skrbnik metodologije (angl. ScrumMaster) in razvojna skupine (angl. Team). Med njimi je ključnega pomena vloga produktnega vodje, ki zastopa interese vseh zainteresiranih uporabnikov. Produktni vodja posreduje vizijo o tem, kaj je treba narediti, in določa kriterije, po katerih se presojajo rezultati. Za nemoten potek dela je pomembno, da pravočasno odgovarja na vprašanja v zvezi s podrobnostmi uporabniških zgodb in sproti preverja ustreznost njihove realizacije. V našem primeru je uredništvo za produktnega vodjo določilo pomočnika odgovorne urednice, ki je v preteklosti že sodeloval pri projektih razvoja večjih spletnih mest, kar je bilo za uspeh projekta zelo pomembno.



Slika 4.1: Del udeležencev projekta nekaj minut pred objavo portala. Na zaslonih vidni stara in nova verzija portala. Od leve proti desni: Rok Štebe (skrbnik metodologije), Robert Schmitzer (produktni vodja), Tim Rijavec (član razvojne skupine) in Janez Urevc (član razvojne skupine, avtor diplomskega dela).

Kandidata za vlogo skrbnika metodologije sta bila vodja razvoja in njegova pomočnica. Ker ima vsak od njiju veliko dodatnih zadolžitev (komunikacija in usklajevanje razvoja z ostalimi uredništvi v hiši na drugih projektih), nismo želeli tvegati in tako pomembne vloge prepustiti le enemu človeku. Na koncu smo se odločili, da vlogo skrbnika metodologije prevzameta oba. To naj bi jima omogočalo, da si pomagata razdeljevati obveznosti skladno z razpoložljivim časom. Naj poudarimo, da smo v tej točki kršili priporočila metode, saj ta vedno predvideva samo enega skrbnika. Posledice takšne odločitve bomo opisali v nadaljevanju dokumenta.

Tretjo vlogo predstavljajo razvijalci oz. razvojna skupina. Agilne metode v splošnem priporočajo, da jedro razvojne skupine sestavljajo nadpovprečno sposobni razvijalci, saj fleksibilnost razvojnega procesa zahteva več iznajdljivosti in samoiniciativosti. Še pomembneje je, da se vsi strinjajo z načinom dela, ki ga zahteva Scrum.

4.1.2 Določitev koncepta "done"

Koncept "done" zahteva, da je vsaka uporabniška zgodba, ki jo razvijemo v okviru neke iteracije, v celoti dokončana, tako da jo je možno neposredno uporabiti v produkcijskem okolju. Zato je potrebno že od samega začetka skrbeti za njihovo korektno preizkušanje in nadzor nad kakovostjo.

V našem primeru smo v okviru koncepta "done" določili seznam pogojev, ki jih je morala izpolnjevati vsaka zgodba, da je bila dokončno potrjena s strani produktnega vodje. Seznam obsega naslednje pogoje:

- **Rešitev mora uspešno prestati vse sprejeme teste.** Sprejemne teste pred pričetkom razvoja napiše produktni vodja. Iz njih lahko razvijalci razberejo zahteve, ki jih mora izpolnjevati rešitev. Morebitne nejasnosti rešujejo v neposrednem stiku s produktnim vodjo.

- **Koda je napisana v skladu s standardi kodiranja.** S temi standardi je določeno, kako mora biti koda strukturirana, kako se uporabljajo nevidni znaki, kako so strukturirani komentarji ipd. Ustreznost teh standardov preverjamo samodejno.

- **Koda je ustrezno komentirana.** V okviru te točke zahtevamo, da ima vsaka enota kode (razred, funkcija, datoteka ...) pripadajoče komentarje, ki so dovolj obširni in razumljivi. Zahtevnejše enote morajo imeti komentarje tudi znotraj svojega telesa.

- **Napisana mora biti dokumentacija**, kjer je razloženo delovanje rešitve in način namestitve.

- **Rešitev je pregledana s strani drugega razvijalca.** Kodo mora pregledati razvijalec, ki pri njenem razvoju ni sodeloval. Rešitev mora brez težav samostojno namestiti in enostavno razumeti njeno delovanje. Pregledovalec preveri tudi varnostne, performančne in integracijske vidike rešitve.

- **Rešitev je pregledana s strani skrbnika metodologije.** Ta pregled je v osnovi zelo podoben tistemu, ki ga opravi drug razvijalec, le da je ta bolj osredotočen na aplikativne vidike, kot sta intuitivnost uporabniškega vmesnika in kvaliteta uporabniške izkušnje.

- **Rešitev je pregledana in dokončno potrjena s strani produktnega vodje.** Končno odobritev neke rešitve poda produktni vodja. To ne pomeni, da produktni vodja rešitev preizkuša šele na koncu procesa.

Zaželeno je, da sodeluje pri preizkušanju skozi celoten proces razvoja, saj so morebitne pomanjkljivosti tako hitreje ugotovljene in odpravljene.

4.1.3 Dolžina iteracije

Dolžina iteracije *(angl. Sprint length)* vpliva na obseg režije in možnosti za vključevanje sprememb v zahtevah naročnika. Krajše iteracije pomenijo več režije zaradi sestankov, ki jih metoda predpisuje na začetku in koncu vsake iteracije. Glede na to, da je spremembe v zahtevah moč upoštevati samo na začetku vsake iteracije, pa krajše iteracije omogočajo večjo prilagodljivost. Metoda Scrum v svoji originalni verziji predvideva 30-dnevne iteracije, v praksi pa so v zadnjem času pogoste iteracije, ki trajajo dva tedna. V vsakem primeru pa velja pravilo, da morajo biti vse iteracije enako dolge. V našem primeru smo se odločili za tritedenske iteracije, saj se nam je zdelo, da je to najboljši kompromis med obsegom režije in prilagodljivostjo, ki je bila potrebna zaradi pričakovanih sprememb v zahtevah.

4.1.4 Izbira orodja za vodenje projekta

Za pregled nad potekom dela se pogosto uporablja tabla, na katero razvijalci lepijo lističe z zgodbami, kar pa v našem primeru zaradi delne distribuiranosti ekipe ni prišlo v poštev. Zato smo se odločili, da uporabimo enega od sistemov, ki temeljijo na spletnem vmesniku. Na tržišču je možno dobiti veliko takšnih orodij; bodisi komercialnih, bodisi brezplačnih ali prostokodnih. V našem primeru smo se odločili za plačljivo različico orodja "Agilo for Scrum", ki v primerjavi s prosto dostopno verzijo nudi še profesionalno podporo in nekaj dodatnih funkcionalnosti, ki uporabo orodja naredijo nekoliko bolj prijazno.

4.1.5 Ocenjevanje zahtevnosti zgodb

Metoda Scrum zahteva, da razvojna skupina oceni zahtevnost vsake uporabniške zgodbe. Ocena je izražena s številom točk *(angl. story points)* in skupaj s prioriteto posameznih zgodb, ki jo določi produktni vodja, predstavlja osnovo za izdelavo načrta izdaje *(angl. Release Plan)*, s katerim določimo dinamiko projekta in okvirno vsebino vsake iteracije.

V skladu s priporočili iz literature [4] smo se odločili, da bomo za ocenjevanje uporabili metodo "Planning poker" [6], kot možno število točk, ki jih dodelimo vsaki zgodbi, pa upoštevali samo vnaprej predpisane dopustne vrednosti 0.5, 1, 2, 3, 5, 8 in 13. Pri tem smo se dogovorili, da ena točka

predstavlja en delovni dan, ki obsega 6 ur efektivnega dela na projektu. Ta dogovor nam je poenostavil načrtovanje posameznih iteracij, saj je število točk predstavljalo tudi število potrebnih človek-dni.

Za načrtovanje vsake iteracije je potrebno določiti še hitrost razvoja (angl. *velocity*), i pove, koliko točk lahko razvojna skupina realizira v eni iteraciji. V vsako iteracijo lahko namreč uvrstimo samo toliko uporabniških zgodb, da je seštevek njihovih točk enak predvideni hitrosti. V našem primeru smo hitrost na začetku projekta ocenili na 32,5 točke. To številko smo dobili tako, da smo število delovnih dni zmnožili s številom razvijalcev in utežjo, ki je predstavljala izkušenost posameznega razvijalca. Začetno oceno smo v naslednjih iteracijah prilagajali glede na dejansko doseženo hitrost.

4.1.6 Izobraževanje

Da bi razvojni ekipi predstavili potek razvoja in s Scrumom seznanili tudi tiste člane ekipe, ki doslej po tej metodi še niso delali, smo pred začetkom projekta izvedli izobraževanje, kjer smo razložili principe, potek in namen metode. S tem smo želeli doseči, da bi se vsi sodelujoči zavedali nalog, ki jih metoda od njih pričakuje, in razumeli, zakaj so le-te potrebne. V izobraževanje smo vključili tudi produktnega vodjo, kjer smo natanko definirali njegovo vlogo, mu razložili njegove zadolžitve in dosegli to, da je pozitivno sprejel metodologijo ter predlagani način dela.

4.1.7 Delna distribuiranost razvojne ekipe

Zanimiv aspekt našega projekta je bilo tudi dejstvo, da razvojna skupina ni delovala na eni lokaciji. Medtem, ko je bila slaba polovica ekipe stacionirana na sedežu podjetja v Ljubljani, je ostali del deloval iz najrazličnejših lokacij po Sloveniji. Lahko bi tudi rekli, da je bila naša ekipa delno distribuirana. To seveda za sabo prinese nekatere posebnosti pri delovnem procesu.

Najbolj očitna sprememba je sigurno v izvedbi vsakodnevnih sestankov. Ker ni bilo možno, da bi tej sestanki potekali v živo, smo jih izvajali preko spletne audio konference. Takšen način izvajanja sestankov se je izkazal za povsem primernega, a smo se tudi strinjali, da pri tem pogrešamo nekoliko bolj osebno obliko komunikacije. Predpostavili smo, da bi se to izboljšalo, če bi uporabljali neko obliko video konference. Na trgu smo iskali neko rešitev, ki bi nam to omogočila, a žal nismo uspeli najti ničesar, kar bi za primeren denar zadovoljilo naše potrebe.

Kasneje smo ugotovili, da bi bilo možno postaviti lastno platformo za gostovanje videokonferenc, a se tudi za to nismo odločili, saj bi nam to vzelo preveč časa in zahtevalo preveč dela z vzdrževanjem.

Ostale sestanke je skupina izvajala na klasičen način. Vsake tri tedne so se namreč tisti razvijalci, ki so delovali dislocirano, pridružili svojim kolegom na sedežu podjetja. To se je izkazalo za zelo dobro, saj se je razvojna skupina občasno vseeno sestala in tako razvijala tudi nek bolj oseben odnos. Zanimivo bi bilo analizirati, kako so pri tem uspešne razvojne skupine, ki so distribuirane na večjih razdaljah in posledično nimajo možnosti osebnega srečanja vsakih nekaj tednov.

4.2 Potek projekta

4.2.1 Sestava razvojne skupine

Jedro razvojne skupine so sestavljali trije razvijalci, ki so na projektu delali od vsega začetka. Dva sta že imela predhodne izkušnje s platformo, na kateri smo portal razvijali, eden se je z njo srečal prvič. Prva dva razvijalca sta bila dislocirana, tretji pa je delal na sedežu podjetja v Ljubljani. Glede na to, da se je en član razvojne skupine prvič srečal z platformo Drupal, smo temu prilagodili tudi načrtovano hitrost za prvo iteracijo. Skladno z njegovim napredkom smo hitrost v naslednjih iteracijah ustrezno povečevali.

Tekom projekta se je razvojna skupina razširila. Tako se nam je v peti iteraciji pridružil zunanji sodelavec, katerega nalogi sta bili razrez oblikovnih predlog in izdelava teme portala. Temu razvijalcu ritem dela, ki ga zahteva Scrum, ni ustrezal, saj je delal tudi na drugih projektih. Vendar smo ga morali vseeno vključiti v načrtovanje iteracij, saj je bilo napredovanje razvojne skupine odvisno od rezultatov njegovega dela.

V isti iteraciji se je skupini priključil še en razvijalec interne razvojne ekipe. Tudi ta razvijalec še ni imel izkušenj z novo platformo. Poleg tega na projekt ni bil preusmerjen za poln delovni čas, kar je njegov napredek pri spoznavanju platforme še upočasnilo. Izkazalo se je, da dodaten razvijalec v takšni obliki ni prinesel nobenega povečanja hitrosti, ampak je bila le-ta po njegovi priključitvi celo nižja od predhodne. Razloge za to bi lahko iskali v času, ki so ga ostali razvijalci porabili za pomoč novemu članu ekipe. Posledično smo se po dveh iteracijah odločili, da tega razvijalca umaknemo iz projekta.

Iteracijo kasneje smo dobili še enega zunanjega sodelavca, ki je ravno tako deloval na dislocirani lokaciji. Za razliko od razvijalca, ki je bil zadolžen za razrez, je bil ta razvijalec projektu posvečen v celoti, kar se je izkazalo za zelo

dobro. Ker je imel veliko predhodnega znanja o tehnološki platformi, je bil tudi njegov učinek zelo velik. Po nekaj tednih, ki jih je potreboval za uvajanje, je postal tako produktiven kot razvijalci, ki so bili na projektu od samega začetka.

4.2.2 Potek tipične iteracije

Vsaka iteracija se je pričela s sestankom za načrtovanje iteracije *(angl. Sprint Planning Meeting)*, končala pa s sestankoma za predstavitev rezultatov (angl. Sprint Review Meeting) in oceno kakovosti razvojnega procesa *(angl. Sprint Retrospective Meeting)*. Vmes pa so vsak dan potekali redni 15-minutni dnevni sestanki *(angl. Daily Scrum Meetings)*.

Sestanke za načrtovanje iteracije smo vedno izvajali v četrtek prvega tedna iteracije. Za ta sestanek se je celotna ekipa zbrala na isti fizični lokaciji. Sestanek je potekal ves dan in je bil sestavljen iz dveh delov. V prvem delu smo s produktnim vodjo razčistili vse nejasnosti iz seznama vseh zahtev. Tako so razvijalci dobili dovolj informacij, da so lahko ocenili še neocenjene uporabniške zgodbe in tiste, ki se jim je ocena zaradi novih okoliščin spremenila. Nato smo na podlagi ocenjene hitrosti v iteracijo uvrstili ustrezno število najbolj prioritetnih uporabniških zgodb. V drugem delu sestanka smo te zgodbe razdelili na zaporedje nalog, ki so bile potrebne za njihovo realizacijo. Vsako nalogo smo ovrednotili s številom ur in določili razvijalca, ki bo odgovoren za njeno realizacijo. Končni rezultat sestanka je bil seznam nalog *(angl. Sprint Backlog)*, ki je predstavljal načrt dela v naslednji iteracji.

Sestanku za načrtovanje iteracije je sledilo 12 delovnih dni, namenjenih razvoju. Te dni so smo vsako jutro ob 9:00 izvedli redni dnevni sestanek, namenjen sprotnemu pregledu poteka dela. Dnevni sestanki so bili časovno omejeni na 15 minut in so zaradi distribuiranosti razvojne skupine potekali v obliki spletne konference. Na teh sestankih je vsak razvijalec povedal, kaj je delal prejšnji dan in kaj bo delal na ta dan. Prav tako je imel priložnost, da izrazi morebitne skrbi ali probleme, na katere je naletel. Največkrat je šlo za nerazumevanje zahtev posamezne naloge, zato se je s skrbnikom metodologije dogovoril za kasnejši sestanek s produktnim vodjo. Na dnevnem sestanku smo za vsako nalogo zabeležili, koliko ur dela smo že vložili in koliko ur dela je še ostalo do njene dokončne realizacije. To nam je omogočilo, da smo sproti sledili napredku in identificirali naloge, kjer bi lahko porabili več časa, kot je bilo sprva predvideno.

Nekajkrat se je zgodilo, da se je dnevni sestanek zavlekel. Največkrat je se je to zgodilo zaradi tega, ker smo v njegovem okviru začeli govoriti o stvareh,

ki na ta sestanek ne spadajo. Večinoma je šlo za razčiščevanje nejasnosti pri posameznih nalogah, ki pa bi jih moral ustrezni razvijalec reševati neposredno s produktnim vodjo.

Ko je razvijalec neko uporabniško zgodbo zaključil, jo je moral najprej dokumentirati. To je vključevalo tako opis funkcionalnosti in namestitve, kot tudi tehnične podrobnosti implementacije. V skladu z definicijo koncepta "done" je realizacijo zgodbe najprej preveril eden od razvijalcev, nato pa še skrbnik metodologije. Če so bile ugotovljene pomanjkljivosti, jih je moral avtor rešitve odpraviti, šele nato smo zgodbo posredovali v končni pregled produktnemu vodji.

Sestanka za predstavitev rezultatov in oceno kakovosti razvojnega procesa sta vedno potekala na isti dan in sicer v torek tretjega tedna iteracije, ko se je ekipa ponovno zbrala na isti lokaciji. Naslednji dan (sreda) smo namenili za vzdrževalne aktivnosti, ki se jim razvijalci niso uspeli posvečati med samo iteracijo, v četrtek pa smo začeli z načrtovanjem naslednje iteracije. Na sestanku za predstavitev rezultatov se je produktni vodja odločil, ali bo neko zgodbo sprejel, jo sprejel s pridržkom ali jo zavrnil. Razlika med sprejetimi in s pridržkom sprejetimi uporabniškimi zgodbami je bila v tem, da so slednje potrebovale le manjše popravke, zaradi katerih bi bilo nesmiselno, da bi jih razglasili za zavrnjene. Dejansko hitrost smo izračunali tako, da smo sešteli točke vseh sprejetih in s pridržkom sprejetih uporabniških zgodb.

Isti dan je sledil še sestanek za oceno razvojnega procesa. Izpeljali smo ga tako, da je vsak sodelujoči navedel nekaj praks, ki smo jih v pretekli iteraciji izvajali dobro, nekaj takšnih, ki bi jih lahko izvajali bolje, in nekaj takšnih, ki jih nismo izvajali, a bi bilo vredno razmisliti o njihovi uvedbi. Na podlagi individualnih predlogov smo nato izdelali skupni seznam koristnih izboljšav, iz katerega smo izbrali tiste točke, za katere se je zdelo realno, da jih lahko realiziramo v naslednji iteraciji.

4.2.3 Spremljanje hitrosti

Ves čas smo posebno pozornost namenjali spremljanju načrtovane in dosežene hitrosti. Projekt je na začetku kar presenetljivo lepo stekel. Prvih nekaj iteracij je skupina kvalitetno načrtovala in tudi zelo korektno izvedla.

V tabeli 4.1 je podana primerjava med planirano in dejansko doseženo hitrostjo po iteracijah. Podatki iz tabele kažejo na to, da smo bili v sredi projekta soočeni z določenimi izzivi, saj je viden opazen padec uspešnosti v 3., 4. in še posebej v 5. iteraciji. Kasneje so se stvari sorazmerno uredile, tako da smo proti koncu projekta spet dosegli planirano hitrost.

Iteracija	Planirana hitrost	Relizirana hitrost
1.	32,5 točke	30,5 točke
2.	34,5 točke	32,5 točke
3.	35 točke	26 točke
4.	40 točke	31,5 točke
5.	39,5 točke	7,5 točke
6.	42 točke	29 točke
7.	32 točke	32 točke

Tabela 4.1: Primerjava med planirano in dejansko hitrostjo

Najopaznejši padec hitrosti sovpada s časom, ko so se dogajale spremembe v razvojni skupini. Čeprav je to nedvomno eden od razlogov za slabše doseganje rezultatov, gotovo ni edini. Med ostalimi razlogi velja omeniti še preveč optimistično načrtovanje, manjšo skrb za metodo po določenem času, manjše nesporazume s produktnim vodjo ipd. Enega od razlogov so prav vsi vpleteni izpostavili pri kasnejših analizah. Mnenja so bili, da je velik vpliv na potek nekaterih iteracij imela slabša izvedba sestanka za planiranje iteracije. Izkazalo se je, da so bile tiste iteracije, ki so potekale slabše, tudi na samem začetku manj precizno načrtovane. Po drugi strani so tiste iteracije, ki smo jih planirali dovolj korektno, potekale skoraj brez težav.

4.2.4 Načrt izdaje in roki

Metoda Scrum priporoča, da pred začetkom projekta izdelamo načrt izdaje, iz katerega je razvidno, katere uporabniške zgodbe bo ta izdaja vsebovala in v kakšnem vrstnem redu bo potekala njihova realizacija po posameznih iteracijah. Na podlagi predvidenega števila iteracij lahko potem enostavno napovemo rok za dokočanje projekta. Da takšen dokument sestavimo, potrebujemo že na začetku razmeroma popoln seznam uporabniških zgodb in ocene njihove zahtevnosti. Z dodajanjem in spreminjanjem zgodb med samim potekom projekta se ta načrt lahko spreminja, vendar pa nam njegovo vzdrževanje omogoča, da vedno poznamo približni rok za dokončanje projekta oziroma (gledano z drugega zornega kota) obseg funkcionalnosti, ki jih lahko realiziramo do določenega roka. Izdelava in vzdrževanje takega načrta seveda zahteva nekaj časa in dodatnih analiz. Ravno slednje, v kombinaciji s pomanjkanjem časa, je bil največji razlog, da v našem primeru tega načrta nismo naredili.

To se je kasneje izkazalo za sorazmerno slabo odločitev, saj brez takšnega načrta objektivno nismo mogli napovedati roka izvedbe projekta, ampak so

bili zastavljeni roki bolj groba začetna ocena, ki je tekom projekta nismo spreminjali, kot pa dejanskih možnosti razvojne skupine. Ker načrta izdaje nismo imeli in smo posledično lahko spremljali le kratkoročni napredek od iteracije do iteracije, na koncu zadanih rokov nismo uspeli izpolniti. Posledično se je zgodilo, da se je načrtovan rok izdelave premaknil kar dvakrat. Najprej smo objavo projekta načrtovali za september, ta datum nato prestavili na konec oktobra in projekt dejansko splavili 1. decembra. Če bi na začetku izdelali načrt izdaje, bi lahko že veliko prej identificirali ta problem in bodisi zmanjšali nabor funkcionalnosti, nujnih za prvo izdajo, bodisi bi zamik produkcijskega roka lahko predvideli že na začetku projekta.

4.2.5 Sodelovanje s produktnim vodjo

Zdi se, da je bil produktni vodja na začetku projekta nekoliko presenečen nad obsegom dela, ki smo ga od njega pričakovali. Na začetku je bilo nekaj težav z odzivnostjo z njegove strani, kar pa bi lahko pripisali ravno temu. Vendar pa se je produktni vodja svoje vloge zelo hitro navadil in od takrat naprej je bilo sodelovanje z njim zelo dobro.

Popolnoma samostojno je urejal seznam vseh uporabniških zgodb, jim določal prioriteto in pisal sprejemne teste zanje. Sprejemni testi so se izkazali za zalo pomembne, saj so iz njih razvijalci zelo enostavno razbrali, kaj produktni vodja od njih pričakuje. V primeru dodatnih nejasnosti je bil produktni vodja vedno na voljo za vprašanja.

Poleg tega je bil produktni vodja prisoten na vseh sestankih za planiranje in predstavitev rezultatov iteracije. Skozi sodelovanje z njim se je razvil pozitiven odnos in medsebojno zaupanje, kar se je seveda izkazalo za zelo dobro.

Smo mnenja, da je vloga produktnega vodje zelo pomembna. Glede na to, da odnos med produktnim vodjo in razvojno skupino ni enak klasičnemu odnosu z naročnikom, je zelo pomembno, da se produktni vodja zaveda svoje partnerske vloge v projektu.

4.2.6 Rezultati

Sam projekt je naletel na precej pozitivne odzive. Portal je po prenovi začel beležiti vse večjo rast prometa. Naročnik je veliko zadovoljstvo izrazil tudi zaradi dejstva, da so uredniki na novem portalu svoje delo opravljali približno četrtino hitreje kot na starem.

Tudi javnost je nov portal sprejela zelo dobro. Pozitivne komentarje smo dobivali tako od stalnih obiskovalcev, kot tudi od uporabnikov, ki niso tipični

uporabniki portala.

Seveda projekt ni potekal idealno. Proti koncu projekta, ko se je začel približevati rok izdaje, se je metoda nekoliko razvodenela. Tudi konceptu "done" v tistem obdobju nismo več sledili v tolikšni meri kot na začetku projekta. Dogajalo se je tudi, da so se zahteve spreminjale že med samimi iteracijami. Vse to je v ekipo prinašalo nekaj napetosti, ki pa smo jih k sreči uspeli dokaj hitro in učinkovito reševati.

Glavni razlog za pojavljanje omenjenih pomanjkljivosti je najverjetneje v dejstvu, da nikoli nismo izdelali pravega načrta izvedbe. Zaradi tega nikoli nismo točno vedeli katere funkcionalnosti bomo morali razviti in posledično nismo znali oceniti koliko dela bo pravzaprav potrebnega za normalno realizacijo projekta.

4.2.7 Plan in dejanski dosežki

Zaradi dejstev, ki smo jih navedli v prejšnjem razdelku, se je datum predaje projekta prestavil za približno 2 meseca. Na koncu smo uspešno izdelali približno 80% funkcionalnosti, ki si jih je naročnik želel. To ni kritično vplivalo na uspešnost projekta, saj smo lahko zaradi modularne zasnove projekta, manjkajoče funkcionalnosti razvili kasneje.

Tisti del projekta, ki je bil realiziran se je obnesel zelo dobro. Tudi zamik predaje se ni izkazal za zelo negativnega, saj škode zaradi tega praktično ni bilo.

4.2.8 Pridobljene izkušnje

Med uvajanjem metode Scrum smo se predvsem veliko naučili. To še toliko bolj drži zaradi dejstva, da se je polovica ekipe tokrat prvič v realnem okolju srečala z metodo Scrum. V procesu smo naredili kar nekaj napak. To je iz vidika poteka projekta verjetno pomanjkljivost, a smo hkrati prav zato pridobili še nekaj dodatnih izkušenj, ki nam bodo prišle zelo prav pri bodočih projektih. Napake so, po našem mnenju, realnost vsakega projekta. Najpomembneje je, da se iz njih čim več naučimo. Naj izpostavimo tiste ugotovitve in izkušnje, ki se nam zdijo najbolj pomembne:

- Vloga produktnega vodje je ključnega pomena. Produktni vodja je zadolžen za urejanje seznama zahtev in stalno zagotavljanje informacij, ki jih razvijalci potrebujejo pri delu. Neodziven ali nekooperativen produktni vodja bo delovni proces zelo otežil. Iz prakse poznamo tudi primere,

ko v vlogi produktnega vodje nastopa eden od izkušenejših razvijalcev. Ta nato komunicira z naročnikom, ki se ne zaveda, da izvajalec deluje po metodi Scrum. Vendar pa se pri takšni organizaciji povečata verjetnost komunikacijskih šumov in čas, ki ga porabimo za režijo. Če je le možno, naj bo produktni vodja dejansko predstavnik naročnika, ki se zaveda svoje vloge in v svoji vlogi vidi smisel.

- Podpora vodstva je nujna za uvedbo katerekoli metode. Vodstvo ima ponavadi dovolj moči, da lahko uporabo neke metode podpre ali jo v celoti zavrne. Zelo težko je voditi strukturiran razvojni proces, če se celotno vodstvo tega ne zaveda in ne pozna vseh prednosti in omejitev neke metode. Pod izrazom "vodstvo" na tem mestu razumemo vse osebe v hierarhiji organizacije, ki imajo moč vplivati na proces. To seveda vključuje tako direktne nadrejene, kot tudi najvišje postavljene izvršne managerje.

- Pri uvajanju se čim bolj držimo priporočil metode. V vseh primerih, kjer smo zavestno sprejeli odločitve, ki niso bile v skladu z metodo, se je na koncu izkazalo, da te niso bile dobre. Priporočali bi, da se Scrum na začetku implementira po navodilih teorije. Morebitne prilagoditve lahko še vedno uvajamo kasneje, ko pridobimo nekoliko več izkušenj. Tako bo verjetnost, da bodo naše odločitve pravilne, veliko večja.

- Če želimo imeti vpogled v časovni potek projekta, moramo nujno izdelati načrt izdaje. To je še ena izkušnja, ki smo jo pridobili iz napake. Nemogoče je realno napovedati čas razvoja, če načrta izvedbe ne izdelamo. Tudi če ga izdelamo, se rok za izvedbo lahko podaljša v primeru pogostega spreminjanja zahtev.

- Ko se projekt začne prevešati v zadnji del, je lahko pritisk rokov zelo nevaren. Takrat se je potrebno upreti skušnjavi, ki nas žene v smer neupoštevanja metode. Tudi v najbolj kritičnih trenutkih je treba nadaljevati s postopkom, ki ga zahteva metoda, saj se odstopanje od dogovorjenega procesa zelo hitro odraža v napakah, slabi volji in pojavljanju konfliktnih situacij.

- Scrum se izkaže za zelo primernega v situacijah, ko so specifikacije zelo ohlapne ali se pogosto spreminjajo. To je navsezadnje tudi ena od prednosti, ki jo literatura v zvezi s Scrumom najpogosteje navaja. Poudariti je potrebno, da spremembe in nedefinirane specifikacije kljub vsemu vplivajo na povečanje porabljenega časa in denarja. Ravno iz tega razloga

je korektno planiranje na začetku projekta v primerih, ko je to mogoče, zelo zaželeno.

- Skrbnik metodologije naj bo samo eden. V drugem delu prispevka smo navedli razloge za delitev vloge skrbnika metodologije med dve osebi. Na koncu je prevladalo mnenje, da to ni bila najboljša odločitev, saj je večkrat prišlo do nesporazumov, ki so bili posledica šumov v koordinaciji med njima.

- Sestava razvojne skupine naj se med projektom ne spreminja. Dodajanje novih sodelavcev med projektom ima pogosto nasproten učinek od zaželenega. Namesto da bi se produktivnost povečala, se razvoj upočasni, roki za izdelavo pa podaljšajo. Posebno je treba biti pozoren pri usklajevanju razvijalcev, ki delajo na različnih področjih.

- Koncept "done" je zelo koristen za zagotavljanje kakovosti izdelka. Koncept smo na začetku projekta dobro definirali, njegovo upoštevanje pa je zagotavljalo stopnjo kakovosti, ki bi jo sicer zelo težko dosegali. Vseeno pa bi lahko postopek zagotavljanja kakovosti še izboljšali:

  - Testiranje izvedenih nalog v začetnih fazah projekta ni bilo izvedeno dovolj dobro, zato so bile nekatere pomanjkljivosti opažene šele proti koncu projekta.
  - Zaradi pomanjkanja časa smo proti koncu projekta začeli varčevati pri pregledih programske kode in se omejili predvsem na testiranje funkcionalnosti.

- Udeležencem projekta moramo že na samem začetku zelo dobro razložiti metodo samo, predvsem pa njen namen in smisel. Udeleženci projekta naj imajo možnost izraziti tudi pomisleke, ki jih je seveda potrebno dobro predebatirati in poizkušati udeležencem razložiti zakaj so ti neutemeljeni oz. skupaj najti rešitev zanje. Samo s tem bomo dosegli sprejemanje metode pri vseh udeležencih, kar pa bo zelo pomembno za njen potek. Samo udeleženci, ki bodo metodo sprejeli in se z njo zares strinjali, jo bodo tudi zares uporabljali.

- Testiranje naj bo sprotno ves čas. Bolje, da se z neko uporabniško zgodbo ukvarjamo nekoliko več časa, ko poteka njen razvoj, kot da šele kasneje ugotovimo njene pomanjkljivosti oz. celo popolno neprimernost implementacije. Pri tem aspektu ponovno zelo pomembno vlogo igra produk-

tni vodja, ki je med drugim zadolžen za testiranje in potrjevanje primernost uporabniških zgodb. Kvalitetno testiranje mu bo seveda vzelo kar nekaj časa, zato mu to razložimo že na začetku projekta. S tem bomo zagotovili, da bo produktni vodja na to nalogo pripravljen.

- Preden neko uporabniško zgodbo uvrstimo v iteracijo, naj bodo izpolnjeni vsi zunanji predpogoji zanjo. Kot zunanje predpogoje na tem mestu razumemo tiste, na katere sami nimamo vpliva, saj jih izpolnjujejo zunanja podjetja oz. partnerji, ki v metodo niso vključeni. Če bomo v iteracijo sprejemali zgodbe, ki še nimajo izpolnjenih teh pogojev, se bo njen razvoj zelo verjetno zakompliciral ali zavlekel. Pogosto se namreč zgodi, da se določene podrobnosti pokažejo šele takrat, ko imamo o zgodbi popolno sliko. Če se tega priporočila ne bomo držali, lahko na takšni zgodbi pričakujemo spreminjanje zahtev in posledično podaljševanje časa njene implementacije.

Poglavje 5

Analiza točnosti ocenjevanja

Eden temeljnih delov vsake metode razvoja programske opreme je vsekakor ocenjevanje zahtevnosti uporabniških zgodb oz. nalog. Ocene zahtevnosti so predpogoj za načrtovanje razvoja, saj ga brez njih praktično ne moremo načrtovati. Šele na podlagi ocen lahko vidimo, koliko časa oz. denarja bomo porabili za razvoj neke enote programske opreme, kar pa je seveda navadno, predvsem za naročnika, ključna informacija.

Navadno je največji izziv pri ocenjevanju dejstvo, da razvijalci tipično ocenjujejo zelo optimistično. To še posebej velja za manj izkušene razvijalce. Optimistično ocenjevanje lahko vodi v zamujanje in posledično povečevanje stroškov projekta. To lahko poslabša odnose med naročnikom in izvajalcem, kar v projekt prinaša slabo voljo in zmanjševanje stopnje medsebojnega zaupanja.

Stroka predlaga več različnih metod ocenjevanja, ki naj bi do določene mere omejile probleme, ki smo jih navedli. Mi smo v ta namen uporabljali metodo *Planning poker*, ki jo literatura priporoča v primeru uporabe agilnih metodologij [5].

5.1 Planning poker

Planning poker je skupinska metoda za ocenjevanje zahtevnosti nalog, ki temelji na doseganju konsenza v skupini. Večinoma se uporablja prav pri ocenjevanju zahtevnosti uporabniških zgodb pri projektih razvoja programske opreme. Še posebej je popularna prav med uporabniki agilnih metod razvoja. Formalno je bila prvič opisana leta 2002 [16], njeno širšo uporabo pa je spodbudil Mike Cohn [4].

Pri ocenjevanju po tej metodi skupina uporablja karte (od tod tudi njeno

ime). Na kartah so zapisane predefinirane vrednosti točk (angl. *story points*), ki označujejo zahtevnost neke zgodbe. Točke ponavadi zaradi lažje reference razumemo kot ekvivalent neki količini časa. Najpogosteje je ena točka enaka enemu delovnemu dnevu enega razvijalca, seveda pa si to referenco lahko vsaka skupina določi po svoje.

Lestvica ocen naj bi bila progresivna (razlike med večjimi vrednostmi so večje od razlik med manjšimi vrednostmi). To naj bi odražalo dejstvo, da je obsežnejše zgodbe težje pravilno oceniti od preprostejših. V literaturi je večinoma priporočena lestvica, ki temelji na fibonaccijevem zaporedju (0, 0.5, 1, 2, 3, 5, 8, 13, 20, 40, 100). Takšno lestvico smo uporabljali tudi mi.

Ocenjevanje posamezne zgodbe po metodi Planning poker poteka v več krogih. Trajanje vsakega kroga je tipično omejeno na nekaj minut. Pred prvim krogom sem med ocenjevalci razvije debata, v kateri se v grobem definira način implementacije zgodbe, ki se trenutno ocenjuje. Zelo pomembno je, da v tej fazi noben izmed udeležencev ne poda prav nobene ocene o zahtevnosti. S tem želimo preprečiti, da bi vplivnejši posamezniki v skupini vplivali na ostale in tako hote ali nehote dosegli povečanje ali zmanjšanje ocen ostalih ocenjevalcev. To bi lahko povzročila že izjava v stilu "Ta zgodba je zelo enostavna" ali "Zgodba se mi zdi zelo zakomplicirana.".

Ko se čas za debato zaključi (debata v prvem krogu naj ne bi bila daljša od 5 minut), vsak ocenjevalec neodvisno od ostalih izbere karto z oceno, za katero misli da je za ocenjevano zgodbo najbolj primerna. Svoje ocene (karte) vsi ocenjevalci pokažejo istočasno.

V naslednjem krogu se k obrazložitvi svoje ocene najprej povabi tista dva ocenjevalca, ki sta v prejšnjem krogu podala najnižjo in najvišjo oceno. Njuni argumentaciji ponovno sledi odprta debata, ki pa naj tokrat ne bi trajala več kot 3 minute. Po preteku treh minut se ocenjevanje ponovi. Krogi razprave in ocenjevanja se ponavljajo toliko časa, dokler skupina ne doseže konsenza.

Med ocenjevalce naj bi bili vključeni samo razvijalci, saj s tem nekako prevzemajo odgovornost za svoje ocene. Smatra se tudi, da so prav razvijalci najbolj verodostojen vir za te ocene, saj imajo z razvojem največ izkušenj. Seveda tak sistem zahteva veliko mero zaupanja med razvijalci, projektnimi vodji in naročnikom.

Ko so ocenjene vse zgodbe, se v naslednjo iteracijo sprejemajo glede na njihovo prioriteto in predvideno hitrost razvojne skupine. Ker naj bi vsaka razvojna skupina čez čas pridobila verodostojno oceno o svoji dejanski hitrosti (produktivnosti), bomo točno vedeli koliko točk lahko skupina realizira v eni iteraciji. Zgodbe bomo zato sprejemali v iteracijo, dokler to število ne bo doseženo.

5.2 Statistična analiza podanih ocen

Med potekom našega projekta smo med ocenjevanji pred 3., 4., 5., 6. in 7. iteracijo zajemali tudi podatke, ki smo jih nato uporabili za statistično obdelavo. Posamezne uporabniške zgodbe je ocenjevalo 5 članov razvojne skupine in skrbnik metodologije. Med ocenjevanjem smo dosledno, za vsak krog ocenjevanja posebej, beležili ocene vseh šestih ocenjevalcev in končno, skupno oceno.

Na ta način smo poizkušali predvsem raziskati, če držijo navedbe iz literature [10], da metoda Planning poker prispeva k zmanjševanju pretiranega optimizma pri ocenjevanju, ki je med razvijalci zelo pogosto prisoten. Seveda smo želeli tudi ugotoviti, ali je ocena po metodi Planning poker v splošnem bližja dejansko porabljenemu času od povprečne vrednosti ocen v prvem krogu ocenjevanja. Če temu ne bi bilo tako, bi to seveda pomenilo, da metoda nima smisla, saj bi lahko z preprostim izračunom zgodbe ocenili veliko hitreje.

5.2.1 Zajem in priprava podatkov

Ocenjevanje je tipično poteklo na sestankih za načrtovanje iteracije. Na vsakem sestanku smo ocenili tiste neocenjene zgodbe, za katere so bili izpolnjeni vsi predpogoji za njen razvoj v prihajajoči iteraciji. Včasih se je tudi zgodilo, da smo neko zgodbo, ki je bila že ocenjena, ocenili ponovno. To se je tipično zgodilo zaradi spremenjenih specifikacij ali okoliščin, ki so vplivale na njeno zahtevnost. V takšnem primeru smo pri analizi upoštevali le zadnjo oceno.

Ker se je ekipa med potekom projekta spreminjala, na nekaterih ocenjevanjih pa so določeni razvijalci manjkali tudi zaradi drugih razlogov (bolezen, dopust, ...), so bili podatki deloma nepopolni. V takšnem primeru smo pri izračunu povprečne vrednosti upoštevali samo ocene prisotnih ocenjevalcev.

Ko je bila neka zgodba zaključena, smo na podlagi zabeleženih ur izračunali dejansko zahtevnost zgodbe. Zahtevnost smo ocenjevali v t.i. *točkah*, ki so pomenile približen ekvivalent šestim uram efektivnega dela. Seštevek ur, ki so bile porabljene na neki zgodbi, smo delili s 6 in tako dobili dejansko zahtevnost zgodbe, izraženo v točkah. Ko smo imeli ocenjeno in dejansko zahtevnost izraženo v isti enoti, smo ju lahko primerjali med seboj.

Pri nekaterih zgodbah se je žal zgodilo, da porabljenega časa ni bilo možno verodostojno izračunati. Največ takšnih zgodb je bilo iz iteracij, ki so bile bližje koncu projekta. V tistem času se je namreč že zelo čutil pritisk roka izvedbe, zato beleženje porabljenega časa ni bilo tako kvalitetno kot bi moralo biti. Takšne zgodbe smo iz svoje analize izključili. Na koncu nam je ostalo 49

zgodb, nad katerimi smo izvedli analizo.

5.2.2 Analiza podatkov

Analizirati smo želeli odnos med končno oceno vsake zgodbe in povprečjem individualnih ocen v prvem krogu. Metoda Planning poker naj bi namreč pomagala dosegati boljše ocene prav zaradi dejstva, da ocenjevalci ocenjujejo skupinsko. Debata, ki se razvije med posameznimi krogi ocenjevanja, naj bi zmanjšala optimizem ocenjevalcev, kar pa naj bi pripeljalo do bolj točnih ocen. Če to drži, bi morale biti končne ocene bolj točne od povprečja individualnih ocen v prvem krogu.

Obe oceni smo analizirali s petimi različnimi merami. Najpreprostejša mera je absolutna napaka:

$$AE = |A - E|$$

Ker nam absolutna napaka pove velikost napake brez upoštevanja velikosti ocene, je lahko v nekaterih primerih nekoliko zavajujoča. V tem pogledu je nekoliko boljša relativna napaka, ki je podana z enačbo:

$$RE = \frac{|A - E|}{A}$$

Spremenljivki na desni strani enačbe predstavljata dejansko (A) in ocenjeno (E) zahtevnost neke zgodbe. Relativna napaka nam pove samo magnitudo napake. Če nas zanima tudi smer, je bolj primerna mera REbias:

$$REbias = \frac{A - E}{A}$$

Poleg AE, RE in REbias smo izračunali tudi BRE (*Balanced measure of relative error*) in BREbias, ki ju uporabljajo tudi v [9] in [10].

$$BRE = \frac{|A - E|}{min(A, E)}$$

$$BREbias = \frac{A - E}{min(A, E)}$$

Kot je vidno iz enačb, BRE ocenjuje le velikost napake, medtem ko BREbias ocenjuje tudi smer napak (optimistična oz. pesimistična ocena). Prednost ocen BRE in BREbias je v tem, da enakovredno obravnava tako pesimisitčne kot optimistične ocene.

Izračunali smo tudi povprečne vrednosti mer za celotno množico zgodb.

$$\overline{AE} = avg(AE_i);\ i = 1, 2, ..., 49$$
$$\overline{RE} = avg(RE_i);\ i = 1, 2, ..., 49$$
$$\overline{REbias} = avg(RE_i);\ i = 1, 2, ..., 49$$
$$\overline{BRE} = avg(BRE_i);\ i = 1, 2, ..., 49$$
$$\overline{BREbias} = avg(BREbias_i);\ i = 1, 2, ..., 49$$

5.2.3 Predstavitev rezultatov

Rezultati analize zajetih podatkov se nahajajo v tabelah 5.1, 5.2, 5.3, 5.4 in 5.5.

Prva zanimivost, ki jo lahko razberemo iz tabel, je vsekakor odnos med oceno, pridobljeno po metodi Planning poker in povprečno vrednostjo ocen prvega kroga. Čeprav to ne drži za vse iteracije, so v splošnem ocene, pridobljene po metodi Planning poker bolj optimistične od povprečnih ocen prvega kroga. Opazimo lahko tudi, da sta v splošnem obe preveč optimistični (tabela 5.6). To je seveda sorazmerno presenetljivo, saj je v literaturi [10] večinoma možno zaslediti tezo, da metoda Planning poker pomaga pri omejevanju optimizma pri ocenah razvijalcev. V našem primeru se je izkazalo ravno nasprotno. Videti je, da so povprečne vrednosti prvega kroga iz tega vidika boljše. Naša raziskava ni prva, ki je pokazala takšne rezultate, zato se lahko upravičeno sprašujemo o smotrnosti uporabe te metode. Po drugi strani je seveda tudi res, da je rezultat lahko odvisen tudi od sestave skupine, izkušenosti njenih članov, prevladujočega vpliva določenih posameznikov itd.

Zanimiva je tudi primerjava uspešnosti obeh ocen v primerjavi z dejansko porabljenim časom. Če v tabeli 5.6 za obe oceni primerjamo mere AE, RE in BRE vidimo, da je napaka po velikosti manjša pri oceni, ki smo jo dobili po metodi Planning poker. Če upoštevamo vrednosti mer, ki nam povejo tudi smer napake (BREbias in REbias), se bo ponovno izkazalo, da so ocene, pridobljene po metodi Planning poker, nekoliko bolj optimistične od povprečja posameznih ocen v prvem krogu. To še enkrat potrjuje ugotovitve, ki smo jih navedli že v prejšnjem odstavku.

				Povprečne vrednosti					Planning poker				
	PpE	AvgE	A	AE	RE	REbias	BRE	BREbias	AE	RE	REbias	BRE	BREbias
1	3	4,20	2,5	1,70	0,68	-0,68	0,68	-0,68	0,50	0,20	-0,20	0,20	-0,20
2	3	5,80	3	2,80	0,93	-0,93	0,93	-0,93	0,00	0,00	0,00	0,00	0,00
3	2	2,00	3,5	1,50	0,43	0,43	0,75	0,75	1,50	0,43	0,43	0,75	0,75
4	3	3,60	1,8	1,80	1,00	-1,00	1,00	-1,00	1,20	0,67	-0,67	0,67	-0,67
5	5	8,00	7	1,00	0,14	-0,14	0,14	-0,14	2,00	0,29	0,29	0,40	0,40
6	1	0,83	1	0,17	0,17	0,17	0,20	0,20	0,00	0,00	0,00	0,00	0,00
7	0,5	0,83	0,5	0,33	0,67	-0,67	0,67	-0,67	0,00	0,00	0,00	0,00	0,00
8	1	1,00	0,8	0,20	0,25	-0,25	0,25	-0,25	0,20	0,25	-0,25	0,25	-0,25
9	2	2,50	4	1,50	0,38	0,38	0,60	0,60	2,00	0,50	0,50	1,00	1,00
10	5	5,00	13	8,00	0,62	0,62	1,60	1,60	8,00	0,62	0,62	1,60	1,60
11	3	2,25	10	7,75	0,78	0,78	3,44	3,44	7,00	0,70	0,70	2,33	2,33
Povp.	2,59	3,27	4,28	2,43	0,55	-0,12	0,93	0,27	2,04	0,33	0,13	0,65	0,45

Tabela 5.1: Podatki po posameznih zgodbah za 3. iteracijo.

				Povprečne vrednosti					Planning poker				
	PpE	AvgE	A	AE	RE	REbias	BRE	BREbias	AE	RE	REbias	BRE	BREbias
1	3	3,25	2,5	0,75	0,30	-0,30	0,30	-0,30	0,50	0,20	-0,20	0,20	-0,20
2	2	2,25	2	0,25	0,12	-0,12	0,12	-0,12	0,00	0,00	0,00	0,00	0,00
3	2	2,25	4	1,75	0,44	0,44	0,78	0,78	2,00	0,50	0,50	1,00	1,00
4	2	3,00	2,5	0,50	0,20	-0,20	0,20	-0,20	0,50	0,20	0,20	0,25	0,25
5	3	4,33	5,5	1,17	0,21	0,21	0,27	0,27	2,50	0,45	0,45	0,83	0,83
6	2	1,75	1,5	0,25	0,17	-0,17	0,17	-0,17	0,50	0,33	-0,33	0,33	-0,33
7	3	5,50	1,5	4,00	2,67	-2,67	2,67	-2,67	1,50	1,00	-1,00	1,00	-1,00
8	3	4,50	2	2,50	1,25	-1,25	1,25	-1,25	1,00	0,50	-0,50	0,50	-0,50
Povp.	2,5	3,35	2,69	1,40	0,67	-0,51	0,72	-0,46	1,06	0,40	-0,11	0,51	0,01

Tabela 5.2: Podatki po posameznih zgodbah za 4. iteracijo.

				Povprečne vrednosti					Planning poker				
	PpE	AvgE	A	AE	RE	REbias	BRE	BREbias	AE	RE	REbias	BRE	BREbias
1	8	6,17	12,5	6,33	0,51	0,51	1,03	1,03	4,50	0,36	0,36	0,56	0,56
2	2	2,00	2,5	0,50	0,20	0,20	0,25	0,25	0,50	0,20	0,20	0,25	0,25
3	5	5,67	5	0,67	0,13	-0,13	0,13	-0,13	0,00	0,00	0,00	0,00	0,00
4	5	4,00	12	8,00	0,67	0,67	2,00	2,00	7,00	0,58	0,58	1,40	1,40
5	3	3,80	4,5	0,70	0,16	0,16	0,18	0,18	1,50	0,33	0,33	0,50	0,50
6	2	2,67	4,5	1,83	0,41	0,41	0,69	0,69	2,50	0,56	0,56	1,25	1,25
7	13	8,17	12	3,83	0,32	0,32	0,47	0,47	1,00	0,08	-0,08	0,08	-0,08
8	8	11,80	7	4,80	0,69	-0,69	0,69	-0,69	1,00	0,14	-0,14	0,14	-0,14
9	1	1,20	1	0,20	0,20	-0,20	0,20	-0,20	0,00	0,00	0,00	0,00	0,00
10	1	1,10	1	0,10	0,10	-0,10	0,10	-0,10	0,00	0,00	0,00	0,00	0,00
Povp.	4,80	4,66	6,20	2,70	0,34	0,11	0,57	0,34	1,80	0,23	0,18	0,42	0,37

Tabela 5.3: Podatki po posameznih zgodbah za 5. iteracijo.

				Povprečne vrednosti					Planning poker				
	PpE	AvgE	A	AE	RE	REbias	BRE	BREbias	AE	RE	REbias	BRE	BREbias
1	3	3,80	3,5	0,30	0,09	-0,09	0,09	-0,09	0,50	0,14	0,14	0,17	0,17
2	5	4,00	2	2,00	1,00	-1,00	1,00	-1,00	3,00	1,50	-1,50	1,50	-1,50
3	1	2,20	1	1,20	1,20	-1,20	1,20	-1,20	0,00	0,00	0,00	0,00	0,00
4	3	2,60	3	0,40	0,13	0,13	0,15	0,15	0,00	0,00	0,00	0,00	0,00
5	3	3,40	3	0,40	0,13	-0,13	0,13	-0,13	0,00	0,00	0,00	0,00	0,00
6	2	1,80	2	0,20	0,10	0,10	0,11	0,11	0,00	0,00	0,00	0,00	0,00
7	1	1,40	2,5	1,10	0,44	0,44	0,79	0,79	1,50	0,60	0,60	1,50	1,50
8	3	4,33	6	1,67	0,28	0,28	0,38	0,38	3,00	0,50	0,50	1,00	1,00
9	3	3,40	4	0,60	0,15	0,15	0,18	0,18	1,00	0,25	0,25	0,33	0,33
10	5	5,50	3	2,50	0,83	-0,83	0,83	-0,83	2,00	0,67	-0,67	0,67	-0,67
11	1	0,83	1	0,17	0,17	0,17	0,20	0,20	0,00	0,00	0,00	0,00	0,00
12	1	0,67	1	0,33	0,33	0,33	0,50	0,50	0,00	0,00	0,00	0,00	0,00
13	2	2,00	2	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
14	5	5,20	3,5	1,70	0,49	-0,49	0,33	-0,33	1,50	0,43	-0,43	0,30	-0,30
15	2	1,67	5	3,33	0,67	0,67	2,00	2,00	3,00	0,60	0,60	1,50	1,50
16	3	4,20	5	0,80	0,16	0,16	0,19	0,19	2,00	0,40	0,40	0,67	0,67
Povp.	2,69	2,94	2,93	1,04	0,39	-0,08	0,51	0,06	1,09	0,32	-0,01	0,48	0,17

Tabela 5.4: Podatki po posameznih zgodbah za 6. iteracijo.

				Povprečne vrednosti					Planning poker				
	PpE	AvgE	A	AE	RE	REbias	BRE	BREbias	AE	RE	REbias	BRE	BREbias
1	5	3,80	8	4,20	0,53	0,53	1,11	1,11	3,00	0,38	0,38	0,60	0,60
2	5	5,50	6	0,50	0,08	0,08	0,09	0,09	1,00	0,17	0,17	0,20	0,20
3	3	3,00	5	2,00	0,40	0,40	0,67	0,67	2,00	0,40	0,40	0,67	0,67
4	1	1,17	1	0,17	0,17	-0,17	0,17	-0,17	0,00	0,00	0,00	0,00	0,00
Povp.	3,50	3,37	5	1,72	0,29	0,21	0,51	0,42	1,50	0,24	0,24	0,37	0,37

Tabela 5.5: Podatki po posameznih zgodbah za 7. iteracijo.

				Povprečne vrednosti					Planning poker				
It.	PpE	AvgE	A	AE	RE	REbias	BRE	BREbias	AE	RE	REbias	BRE	BREbias
3	2,59	3,27	4,28	2,43	0,55	-0,12	0,93	0,27	2,04	0,33	0,13	0,65	0,45
4	2,5	3,35	2,69	1,40	0,67	-0,51	0,72	-0,46	1,06	0,40	-0,11	0,51	0,01
5	4,80	4,66	6,20	2,70	0,34	0,11	0,57	0,34	1,80	0,23	0,18	0,42	0,37
6	2,69	2,94	2,93	1,04	0,39	-0,08	0,51	0,06	1,09	0,32	-0,01	0,48	0,17
7	3,50	3,37	5	1,72	0,29	0,21	0,51	0,42	1,50	0,24	0,24	0,37	0,37
Vse	3,13	3,47	4,05	1,81	0,45	-0,10	0,65	0,11	1,48	0,31	0,06	0,50	0,26

Tabela 5.6: Povprečne vrednosti po posameznih iteracijah in povprečne vrednosti za vse uporabniške zgodbe skupaj.

5.2.4 Analiza veljavnosti

Podatke o ocenjevanju posameznih zgodb smo zajemali zelo vestno in natančno, zato si upamo trditi, da pri tem ni prišlo do napak. Nekoliko več težav smo imeli zaradi dejstva, da dejansko porabljen čas ni bil zabeležen za vse uporabniške zgodbe. To je bil tudi razlog, da smo množico zgodb, na katerih smo izvajali analizo, zmanjšali približno za polovico (iz približno 100 na 49). Na ta način smo želeli zagotoviti verodostojnost vhodnih podatkov in s tem celotne študije.

Poglavje 6

Raziskava prednosti metode Scrum in zadovoljstva udeležencev projekta

Vzporedno s projektom uvedbe metode Scrum v delovni proces podjetja Delo, smo med ostalimi uporabniki metode Scrum izvedli tudi raziskavo o njihovem videnju najpomembnejših faktorjev in dobrih praks, ki so ključni za uspešnost projektov. Ker smo podobno raziskavo izvajali tudi med vključenimi v naš projekt, smo želeli rezultate primerjati med seboj. Na ta način smo želeli ugotoviti, do kakšne mere se odzivi posameznikov na našem projektu skladajo z odzivi drugih uporabnikov metode Scrum.

Da bi ugotovili, kako uporabniki metode Scrum ocenjujejo njene prednosti in kakšen pomen pripisujejo posameznim tipičnim praksam, ki jih Scrum predpisuje, sta anketo (poleg uvodnega dela) sestavljala dva vsebinska sklopa vprašanj. Uvodni del je služil za segmentacijo anketirancev in je obsegal vprašanja o njihovi vlogi v projektih, ki so vodeni po metodi Scrum (ScrumMaster/Product Owner/Član razvojne skupine), in izkušnjah (koliko časa že uporabljajo Scrum in na koliko Scrum projektih so že sodelovali). Namen prvega vsebinskega sklopa vprašanj je bil preveriti, v kolikšni meri se anketiranci strinjajo s trditvami o prednostih metode Scrum, ki jih zasledimo v literaturi. V okviru drugega sklopa pa smo želeli ugotoviti, katere tipične prakse, ki jih Scrum predpisuje, po njihovem mnenju največ prispevajo k uspešnosti projektov.

Za izpolnjevanje ankete smo poleg sodelavcev na projektu prenove spletnega poratala slovenskenovice.si zaprosili še člane slovenske skupine Skram.si, člane skupnosti razvijalcev Drupala in člane skupine Scrum Practitioners, ki

tudi deluje v omrežju LinkedIn. Dobili smo 75 odgovorov, od tega 47 iz Slovenije in 28 iz tujine.

Med anketiranci je bilo 29 razvijalcev, 13 produktnih vodij in 33 skrbnikov metodologije. Tudi njihova izkušenost je bila zelo raznolika. Dva anketiranca metode nista uporabljala še nikoli, 35 jo je uporabljalo največ pol leta, 8 največ eno leto, 11 do dve leti in 19 več kot dve leti. Prav tako 2 anketiranca metode nista uporabila še na nobenem, 48 na največ dveh, 12 na največ petih in 13 na več kot petih projektih.

Namen tega poglavja je skozi rezultate ankete izpostaviti glavne prednosti metode Scrum in izluščiti tiste tipične prakse, ki po mnenju njenih uporabnikov najbolj vplivajo na uspešnost projektov, vodenih po tej metodi. Oblikovati želimo množico nasvetov, ki bodo koristili predvsem tistim uporabnikom Scruma, ki se z njim šele spoznavajo. S tem jim želimo olajšati njegovo uvedbo. Upamo, da bo to pripomoglo k večji uspešnosti projektov in povečanju zadovoljstva ob uvajanju agilnih metod.

V nadaljevanju bomo najprej podrobneje opisali odgovore na oba vsebinska sklopa vprašanj in nato primerjali rezultate, ki smo jih dobili na našem projektu s tistimi, ki so jih prispevali ostali uporabniki metode.

6.1 Ocena prednosti metode Scrum

V prvem sklopu vprašanj smo anketirance spraševali, v kolikšni meri po njihovem mnenju držijo trditve o metodi Scrum, ki so bile objavljene v literaturi. Za osnovo smo vzeli prednosti, ki jih navajata Rising in Janoff v [12]. Anketiranci so veljavnost posameznih trditev ocenjevali s pomočjo 5-stopenjske Likertove lestvice, pri čemer je ocena 1 (najslabša ocena) pomenila, da je trditev v celoti napačna, ocena 5 (najboljša ocena) pa, da trditev v celoti drži (tj., da se anketiranec z njo v celoti strinja). Spraševali smo o naslednjih trditvah:

- **Izdelek (projekt) lahko razdelimo na zaporedje obvladljivih delov.** Ta trditev izhaja iz iterativne in inkrementalne narave metode Scrum, ki zagotavlja, da smo v določenem trenutku osredotočeni na zahteve trenutne iteracije, ki (zaradi doslednega upoštevanja prioritete posameznih zgodb) prinašajo naročniku največjo korist. To zahteva od produktnega vodje (razvijalci pa mu morajo pri tem pomagati), da načrtuje razvoj v obliki večjega števila izdaj, ki jih je moč postopoma in v čim krajših časovnih intervalih predajati v produkcijo. Ključno vprašanje je, ali je takšna razdelitev možna pri vseh projektih, saj se lahko zgodi, da so vse komponente nekega izdelka preveč prepletene med seboj.

- **Projekt napreduje tudi, ko zahteve niso stabilne.** Seznam zahtev se lahko dopolnjuje z novimi uporabniškimi zgodbami ves čas, dokler projekt ni končan. Pri tem je pomembno, da produktni vodja vzdržuje prioriteto posameznih zgodb in s tem zagotavlja, da so zgodbe vedno razvrščene glede na poslovno vrednost, ki jo prinašajo naročniku. To poenostavi načrtovanje iteracij (vsakokrat izberemo za realizacijo tiste zgodbe, ki so trenutno na vrhu seznama), obenem pa zagotavlja, da naročnik vedno dobiva rešitev, ki je zanj v tistem trenutku najbolj koristna. Pri tem je treba poudariti, da se produktni vodja ne sme vmešavati v delo razvojne skupine po tistem, ko je vsebina iteracije že dogovorjena. Med samo iteracijo mora imeti razvojna skupina mir, da se lahko v celoti posveti realizaciji dogovorjene funkcionalnosti. Na začetku naslednje iteracije pa spet sledi dogovarjanje o realizaciji najpomembnejših uporabniških zgodb iz seznama zahtev, ki se je medtem lahko spremenil.

- **Vsi udeleženci imajo popoln vpogled v potek dela.** To je zagotovljeno z rednimi vsakodnevnimi sestanki, na katerih se člani razvojne skupine medsebojno informirajo o poteku dela in morebitnih težavah, na katere naletijo. Uporabniki pa imajo možnost, da na koncu vsake iteracije pregledajo realizirano funkcionalnost in podajo svoje pripombe.

- **Izboljša se komunikacija med člani razvojne skupine.** Ta trditev se neposredno navezuje na prejšnjo. Naj poudarimo, da komunikacijo izboljšuje tudi ti. "face-to-face" pristop. Glede na to, da se vse podrobnosti projekta razčiščujejo v komunikaciji med produktnim vodjo in člani razvojne skupine, se komunikacija v primerjavi z metodami, kjer razvijalci delajo na podlagi pisno podanih zahtev, vsekakor poveča.

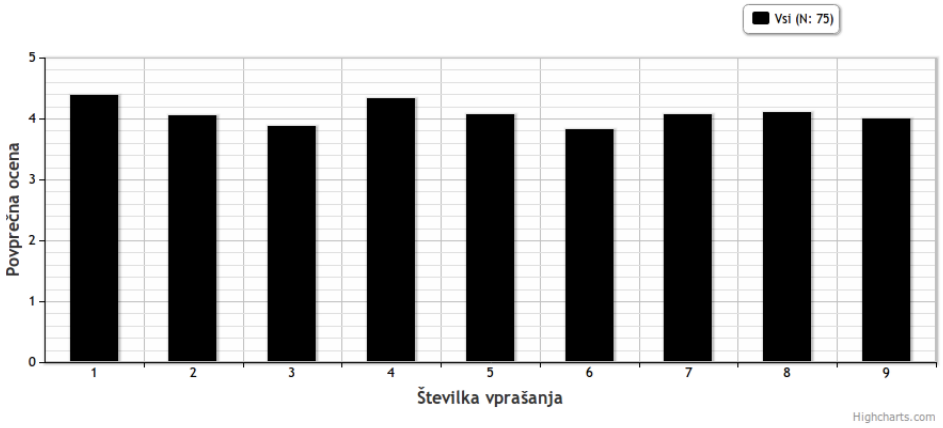
- **Člani razvojne skupine so zaslužni za uspeh v času razvoja in ob zaključku projekta.** Metoda omogoča članom razvojne skupine, da sami ocenjujejo zahtevnost posameznih zgodb in predvideno hitrost razvoja, zato praviloma niso (ali vsaj ne smejo biti) izpostavljeni pritiskom zaradi nerazumno postavljenih rokov in nesprejemljivih zahtev produktnega vodje. Pred tem jih ščiti tudi skrbnik metodologije. Obenem delujejo po načelu samoorganizacije in sami izbirajo način, kako bodo realizirali posamezne zahteve. Zato so pri svojem delu bolj sproščeni. Vse to prispeva k večji odgovornosti in zavzetosti razvojne skupine, da tisto, kar je obljubila na začetku vsake iteracije, tudi uresniči. S tem pa se poveča tudi zavedanje o kolektivnih zaslugah za uspešen potek projekta.

- **Naročnik dobiva novo funkcionalnost v dogovorjenih časovnih intervalih.** To je zagotovljeno z iterativnim in inkrementalnim postopkom razvoja. Vsaka iteracija se konča s sestankom, na katerem razvojna skupina predstavi rezultate svojega dela. Rezultat vsake iteracije mora biti nova funkcionalnost, ki je neposredno uporabna. Dolžina iteracij je vnaprej dogovorjena, tako da naročnik točno ve, v kakšnih intervalih bo dobival predvidene rezultate. Na podlagi sprotnega vzdrževanja plana izdaje pa lahko vedno oceni, kdaj bo izdaja v celoti končana, oziroma kolikšen obseg funkcionalnosti bo lahko reliziran do določenega časovnega roka.

- **Naročnik (uporabniki) lahko sproti preverja(jo), kako izdelana programska oprema v resnici deluje.** Koncept "done" zahteva, da je rezultat vsake iteracije delujoča programska koda, ki je v celoti preverjena in integrirana v trenutno rešitev. Zato lahko naročnik po vsaki iteraciji preveri, kako bo načrtovani sistem v resnici deloval. Pogosto se dogaja, da so uporabniki šele takrat, ko v živo preizkusijo neko rešitev, sposobni pravilno formulirati svoje zahteve. Sprotno preverjanje izdelane programske opreme tako zagotavlja, da uporabnik tako rešitev, kot jo v resnici potrebuje. Po drugi strani pa odpravlja obsežne integracijske teste na koncu projekta.

- **Metoda prispeva k izgradnji boljših odnosov med naročnikom (uporabniki) in razvijalci: poveča se medsebojno zaupanje in presek skupnega znanja.** Ker metoda predvideva sodelovanje predstavnikov naročnika (produktnega vodje) ves čas trajanja projekta, ne pa samo na začetku in na koncu, kot je to praviloma pri tradicionalnem pristopu. To pa vpliva na boljše medsebojno (s)poznavanje vseh sodelujočih, kar se odraža v izboljšanju odnosov in povečanju zaupanja. Razumeti je potrebno, da v projektih razvoja programske opreme pogosto sodelujejo tudi strokovnjaki z ne-tehničnih področij. Pogosta težava v tako heterogenih skupinah se izkaže prav v velikih interesnih razlikah in majhnem preseku skupnega znanja. Dobra in sprotna komunikacija te probleme odpravlja in povečuje presek skupnega znanja.

- **Metoda prispeva k izgradnji pozitivnega vzdušja, v katerem vsi pričakujejo uspeh projekta.** Ta točka je zelo povezana s predhodno. V okolju, kjer vladajo pozitivni odnosi in medsebojno zaupanje, bo tudi vladalo tudi pozitivno vzdušje, to pa se bo seveda odražalo tudi v pozitivnih pričakovanjih.



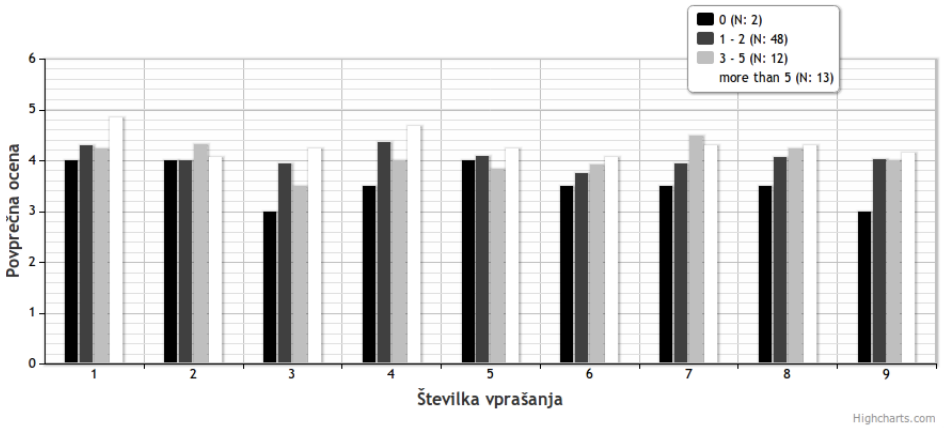
Slika 6.1: Povprečne vrednosti odgovorov na prvi sklop vprašanj.

Povprečne ocene, izračunane na podlagi odgovorov vseh 75 anketirancev, so prikazane na sliki 6.1. Iz slike je razvidno, da se uporabniki metode Scrum strinjajo z vsemi trditvami, saj se vse ocene nahajajo na intervalu med 3,8 in 4,4. Tudi standardni odklon, ki smo ga izračunali, je sorazmerno nizek in je za vse trditve manjši od 1,0.

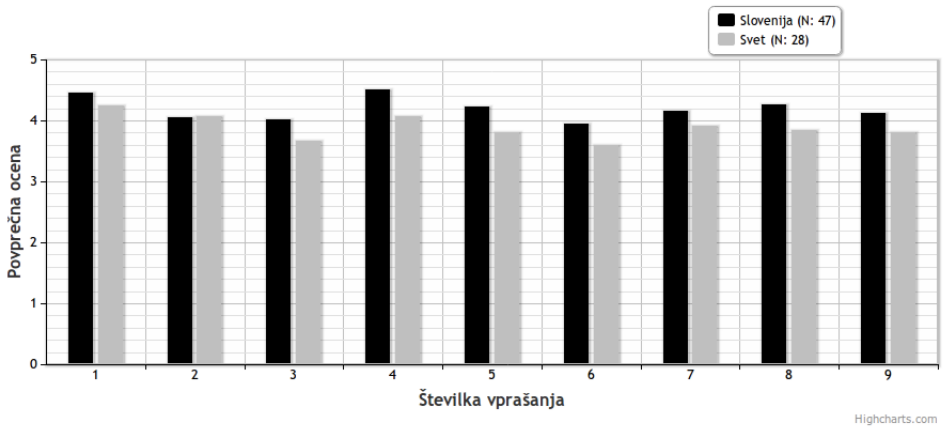
Anketiranci so se najbolj strinjali s trditvama št. 1 (projekt lahko razdelimo na množico obvladljivih delov) in št. 4 (izboljša se komunikacija med člani razvojne skupine). Po drugi strani sta se za najmanj sprejeti trditvi izkazali št. 3 (vsi udeleženci imajo popoln vpogled v potek dela) in št. 6 (naročnik dobiva novo funkcionalnost v dogovorjenih časovnih intervalih).

Odgovore smo dodatno analizirali glede na izkušnje anketirancev (slika 6.2), tako da smo anketirance razdelili na 4 skupine, upoštevajoč število projektov, vodenih po metodi Scrum, pri katerih so že sodelovali (0 projektov, 1-2 projekta, 3-5 projektov in več kot 5 projektov). Pokazalo se je, da se s trditvami opazno bolj strinjajo tisti, ki imajo s Scrumom več izkušenj. Še posebej pa je zanimivo, da se pri večini trditev povprečne ocene povečujejo sorazmerno s številom projektov, pri katerih je anketiranec do sedaj sodeloval. Na podlagi te ugotovitve bi lahko trdili, da se z večanjem izkušenj veča tudi zaupanje v metodo Scrum. Videti je, da so novi uporabniki najprej nekoliko bolj previdni, a se z izkušnjami začnejo bolj zavedati njenih prednosti.

Zanimiva je tudi primerjava odgovorov slovenskih in tujih strokovnjakov, ki jo prikazuje slika 6.2. Prav z vsemi trditvami se slovenski uporabniki metode Scrum strinjajo nekoliko bolj kot njihovi kolegi iz tujine.



Slika 6.2: Povprečne vrednosti odgovorov na prvi sklop vprašanj v odvisnosti z izkušenostjo anketiranca.



Slika 6.3: Povprečne vrednosti odgovorov na prvi sklop vprašanj v odvisnosti z lokacijo anketiranca.

6.2 Faktorji, ki vplivajo na uspeh projekta

V drugem sklopu vprašanj nas je zanimalo, katere tipične prakse, ki jih predpisuje metoda Scrum, najbolj vplivajo na uspeh projekta. Anketiranci so z ocenami od 1 do 5 ocenjevali naslednje faktorje:

- **Jasno zastavljen seznam vseh zahtev.** Seznam zahtev služi kot osnova za ocenjevanje zahtevnosti posameznih zgodb, izdelavo plana izdaje in načrtovanje posameznih iteracij. Sprejemni testi, ki jih mora vsebovati vsaka uporabniška zgodba, omogočajo preverjanje, ali je zgodba realizirana v skladu s potrebami uporabnikov.
- **Sprotna komunikacija s produktnim vodjo.** Ker predstavljajo uporabniške zgodbe samo ohlapen opis vsake zahteve, je za razjasnitev vseh podrobnosti potrebna sprotna komunikacija s produktnim vodjo. Če je ta komunikacija slaba ali ne poteka sproti in se nejasnosti ne razrešijo, lahko pride do situacije, ko razvojna skupina realizira nekaj, kar ne zadovolji pričakovanj naročnika.
- **Dober skrbnik metodologije.** Skrbnik metodologije vsakodnevno bdi nad uporabo metodologije in potekom projekta. Zagotavljati mora take pogoje za delo, da je razvojna skupina čim bolj produktivna. Dober skrbnik metodologije bo zelo zgodaj identificiral odstopanja od začrtanega plana in nastajajoče težave. Z usklajenim delovanjem vseh sodelujočih bo zagotovil, da bodo takšne situacije kar najhitreje razrešene.
- **Dobra komunikacija med člani razvojne skupine.** Dobra komunikacija med člani razvojne skupine se odraža tudi v dobrih in pozitivnih odnosih, kjer vsi člani delujejo kooperativno. Omogoča tudi sproten, celovit in transparenten vpogled v potek dela na projektu ter medsebojno pomoč v primeru težav.
- **Pravilne ocene zahtevnosti posameznih zgodb.** Pravilne ocene zahtevnosti so predpogoj za pravilno načrtovanje plana izdaje in posameznih iteracij. Na podlagi teh ocen in ocenjene hitrosti razvrščamo zgodbe v posamezne iteracije. Če ocene niso vsaj približno pravilne, se nam lahko zgodi, da v neko iteracijo uvrstimo preveč ali premalo zgodb.
- **Začetna razporeditev zgodb po iteracijah - plan izdaje.** Plan izdaje omogoča načrtovanje rokov za izvedbo in časovnih okvirov, v katerih bomo sposobni posamezne dele projekta predati naročniku. Plan izdaje daje celovit vpogled v časovne okvire projekta kot celote.

- **Pravilna ocena hitrosti na začetku vsake iteracije.** Hitrost nam pove kakšno količino dela je razvojna skupina sposobna opraviti v eni iteraciji. Na začetku jo ocenimo glede na različne kriterije (velikost razvojne skupine, izkušenost, ...) kasneje pa jo prilagajamo glede na dejansko doseženo število točk. Ta faktor je zelo povezan s tistim, ki smo ga omenili pod točko 5. Samo kombinacija dobrih ocen zahtevnosti in dobre ocene hitrosti, nam bo omogočila korektno načrtovanje posameznih iteracij.

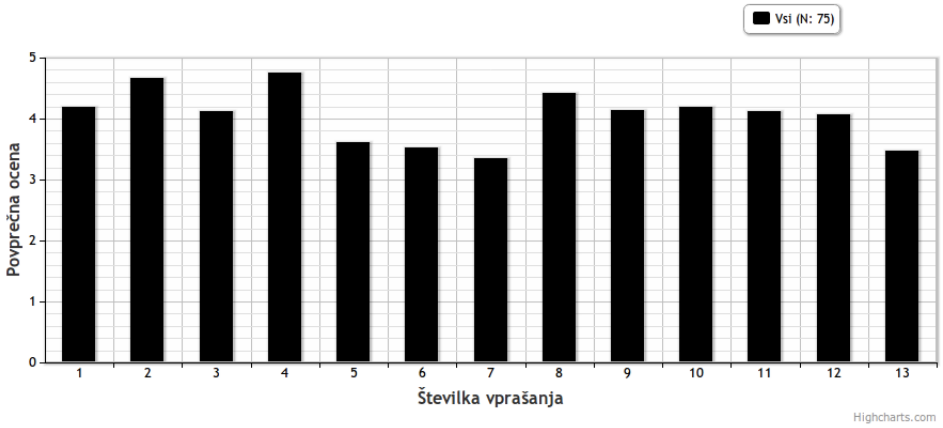
- **Dosledno izvajanje sestankov za načrtovanje iteracije.** Rezultat tega sestanka je seznam vseh uporabniških zgodb, ki jih moramo v iteraciji realizirati, in dekompozicija teh zgodb na posamezne naloge. Ker nam ta predstavlja plan dela za posamezno iteracijo, bo nekorektno izvajanje tega sestanka lahko pripeljalo do slabo vodene in/ali izpeljane iteracije.

- **Dosledno izvajanje vsakodnevnih sestankov.** Vsakodnevni sestanki omogočajo sproten vpogled v stanje projekta. Omoogčajo, da morebitne težave odkrijemo takoj, ko se pojavijo, in jih sproti odpravimo. Paziti pa je treba, da ne postanejo predolgi, saj s tem zmanjšamo čas, ki ga lahko izkoristimo za razvoj. Ravno zato so vsakodnevni sestanki časovno omejeni.

- **Dosledno upoštevanje koncepta "done".** Scrum zahteva, da je na koncu iteracije vsaka funkcionalnost realizirana tako, da jo je moč neposredno predati v uporabo. S tem odpade obsežno testiranje na koncu projekta, ampak imamo vedno na razpolago delujočo verzijo izdelka. Dosledno upoštevanje koncepta "done" omogoča, da naročnik sproti preizkuša novo funkcionalnost in na podlagi tega precizneje oblikuje svoje zahteve.

- **Dosledno izvajanje sestankov za predstavitev rezultatov iteracije.** V okviru teh sestankov produktnemu vodji in zainteresiranim uporabnikom predstavimo rezultate vsake iteracije. Služijo tako za pregled in potrjevanje realizirane funkcionalnosti kot za razpravo o morebitnih pomanjkljivostih in nadaljnjih usmeritvah. S tem sproti preprečujemo morebitne odklone in zagotavljamo, da izdelana funkcionalnost res ustreza potrebam naročnika.

- **Dosledno izvajanje sestankov za uvajanje izboljšav.** Ti sestanki služijo za stalno izboljševanje razvojnega procesa. Razvijalcem dajejo



Slika 6.4: Povprečne vrednosti odgovorov na drugi sklop vprašanj.

možnost, da na podlagi analize pozitivnih in negativnih izkušenj iz prejšnje iteracije definirajo izboljšave, ki jih bodo uvedli v naslednji iteraciji. Seveda je potrebno poudariti, da so ti sestanki koristni samo, če se dogovorjeni ukrepi potem tudi izvajajo.

- **Dobro orodje za spremljanje poteka del na projektu.** Čeprav Scrum ne zahteva obsežne dokumentacije, je koristno imeti orodje, ki nam olajša spremljanje poteka del na projektu. Tovrstna orodja omogočajo vzdrževanje seznama zahtev, seznamov nalog za posamezne iteracije, beleženje podatkov o vloženem in preostalem delu, grafične prikaze napredka ipd.

Iz rezultatov, ki so prikazani na sliki 6.4, je razvidno, da anketiranci pripisujejo največji pomen dobri komunikaciji med člani razvojne skupine (vprašanje 4) in dobri komunikaciji s produktnim vodjo (vprašanje 2). To je tudi razumljivo, saj za Scrum (tako kot za vse agilne metode) velja, da stalna in kvalitetna komunikacija nadomešča pisanje obsežne in zahtevne dokumentacije.

Po drugi strani pa vidimo, da so bili najnižje ocenjeni tiste prakse, ki se nanašajo na ocenjevanje hitrosti (vprašanje 7), načrtovanje izdaje (vprašanje 6) in ocenjevanje zahtevnosti posameznih zgodb (vprašanje 5). V določeni meri je to presenetljivo, še posebej zato, ker ocena hitrosti in ocene posameznih zgodb predstavljajo osnovo za planiranje posameznih iteracij, plan izdaje pa nudi pregled nad projektom kot celoto. Po drugi strani pa je takšen rezultat razumljiv, saj Scrum večinoma uporabljamo v agilnih okoljih, kjer je

v ospredju prilagajanje spremembam v zahtevah, medtem ko sta natančnost planiranja in sledenje planu drugotnega pomena. Sorazmerno nizko je bila ocenjena tudi potreba po ustreznem orodju za spremljanje poteka del, kar pa potrjuje, da je metoda Scrum razmeroma enostavna za uporabo, mnogi uporabniki pa namesto računalniške podpore raje uporabljajo fizične kartice in tablo, na kateri s prestavljanjem kartic prikazujejo, kako napreduje delo na projektu.

6.3 Ugotovitve in opažanja

Naša raziskava potrjuje, da se uporabniki metode Scrum v celoti strinjajo s trditvami o prednostih te metode, ki jih navaja literatura, še zlasti s tem, da Scrum omogoča razbitje projekta na zaporedje manjših obvladljivih delov in izboljša komunikacijo med člani razvojne skupine. Prav komunikacija pa je po mnenju anketirancev najpomembnejši faktor, ki vpliva na uspešnost po Scrumu vodenih projektov. To vključuje tako komunikacijo med razvojno skupino in produktnim vodjo, kot tudi komunikacijo znotraj razvojne skupine in s skrbnikom metodologije.

Druga, nekoliko presenetljiva, ugotovitev govori o tem, da uporabniki metode Scrum ne pripisujejo tako velike vloge ocenjevanju in načrtovanju časovnega poteka projekta, kot bi to lahko pričakovali. To lahko razložimo z naravo projektov, pri katerih se uporablja Scrum. Pogosto gre za spletne in start-up projekte, kjer sledimo principu "release early, release often". Pri tem principu velikokrat pravzaprav ne moremo govoriti o roku izvedbe, saj se izdelek neprestano izboljšuje in dodeluje, izdaje pa se dogajajo pogosto in inkrementalno. To bi lahko potrdil tudi komentar enega od anketirancev: *"Težko je reči, da naročnik dobi dogovorjeno funkcionalnost v roku - ta je namreč fleksibilna in dopolnjujoča in se sproti prilagaja situaciji v projektu."*

Poglavje 7

Zaključek

Pilotni projekt uvedbe metode Scrum v delovni proces spletnega oddelka podjetja Delo je bil vsekakor zelo zanimiva in poučna, predvsem pa dobra izkušnja. V splošnem je bil odziv na projekt z vseh strani zelo pozitiven. Naročnik je s predanim izdelkom zadovoljen, saj so bili doseženi vsi cilji, ki so bili definirani na začetku projekta. Dokazov za to je več. Prvi je vsekakor bistveno večje število obiskov prenovljene spletne strani v primerjavi s prejšnjo. Tudi odzivi konkurence, strokovne in splošne javnosti, so bili več kot pozitivni. Pohvale so prihajale tudi s strani uporabnikov, ki niso tipični bralci Slovenskih novic, kar je glede na sorazmerno specifično publiko portala zelo velik dosežek. Naj omenimo še podatek, da je nov portal le nekaj dni po pričetku obratovanja odigral zelo pomembno in pozitivno vlogo pri spletnem pokrivanju predčasnih državnozborskih volitev. Glede na promet in obremenitev, ki ga takšen dogodek prinaša, je to vsekakor hvale vreden dosežek.

V celoti gledano je bila še največja pomanjkljivost projekta prav zamik datuma predaje. Glede na pozitivne odzive, ki smo jih navedli v prejšnjem odstavku, lahko trdimo, da zaradi tega nismo utrpeli nobene škode. Ker smo se želeli iz projekta naučiti čim več, smo ta vidik vseeno bolj podrobno analizirali, Menimo, da bi se lahko temu izognili oziroma to pomanjkljivost identificirali veliko prej, če bi obstajal načrt izdaje. Žal pri konkretnem projektu, zaradi dejstev, ki smo jih že navedli v članku, tega ni bilo mogoče izdelati.

Ena glavnih izkušenj, ki bi jo radi delili z bodočimi uporabniki Scruma, je povezana z upoštevanjem priporočil, ki jih omenja literatura. Odločitve, ki smo jih zavestno sprejeli v nasprotju s temi priporočili, se večinoma niso izkazale za dobre. Zato bi vsem bodočim uporabnikom Scruma svetovali, da se na začetku uvedbe čim bolj držijo priporočil stroke. Upamo, da bo ta prispevek čim več bodočim in obstoječim uporabnikom Scruma in ostalih agilnih metod pomagal

pri doseganju še boljših rezultatov.

Zelo pomembno je tudi uvajanje udeležencev projekta. Zelo pomembno se nam zdi, da prav vsem udeležencem predstavimo metodo že na samem začetku in se z njimi pogovorimo do te mere, da so vsi dvomi odpravljeni. Zelo pomembno je namreč, da se udeleženci strinjajo z njeno uporabo. Če temu ne bo tako, bodo pri projektu najverjetneje trpeli. Od sodelavca, ki pri projektu ne bo sodeloval s srcem, seveda ne bomo imeli veliko, zato bomo tudi sami še bolj nezadovoljni z njim in njegovim prispevkom h projektu.

Seveda si bomo v okviru metodologije, najverjetneje v obliki nekih dogovorov, postavili neka pravila, v okviru katerih bomo delovali. Zelo pomembno je, da se teh pravil držimo čez celotno trajanje projekta. Če se ne bomo držali pravil, ki smo si jih sami določili, se bomo še težje držali tistih, ki nam jih "postavlja" metoda sama. Njihovo neupoštevanje pa lahko kaj hitro pripelje do razpada metode.

Verjetno najpomembnejša ugotovitev, ki bi jo radi izpostavili je povezana z vlogo produktnega vodje. Njegova vloga je zelo pomembna. Verjetno se na začetku uvedbe, kljub pogostemu poudarjanju tega aspekta v literaturi, tega ne bomo dovolj zavedali. Zelo težko bomo razvoj pravilno vodili po metodi Scrum, če bomo imeli naročnika oz. produktnega vodjo, kateremu se bo takšna angažiranost zdela nepotrebna. Metodologija namreč predpostavlja zelo velik angažma produktnega vodje, za kar pa bo ta seveda porabil tudi veliko časa in energije. Pomembno je, da si ta čas rezervira že na samem začetku projekta.

V našem primeru se je Scrum izkazal za zelo učinkovito metodo, saj nam je omogočal doseganje napredka in agilen razvoj skozi celoten projekt. Menimo, da se je zaradi uporabe metode zelo izboljšala komunikacija in medsebojno zaupanje med udeleženci projekta. Z vidika naslednjih projektov je to zelo pomemben dosežek, saj je to osnovna podlaga za doseganje odličnih rezultatov.

Literatura

[1] S. W. Ambler, "Has Agile Peaked? Let's look at the numbers", *Dr. Dobb's Journal*, May 2008, Dostopno na: http://www.ddj.com/architect/207600615?pgno=1.

[2] M. Beedle, A. van Bennekum, A. Cockburn, et al., "Manifesto for agile software development", Dostopno na: http://agilemanifesto.org/

[3] A. Begel, N. Nagappan, "Usage and preceptions of agile software development in an industrial context: an exploratory study", *First International Symposium on Empirical Software Engineering and Measurement*, Madrid, September 2007, str. 255 - 264.

[4] M. Cohn, "User stories applied", Adison-Wesley, 2004

[5] M. Cohn, "Agile Estimating an Planning", Prentice Hall Professional Techincal Reference, 2006

[6] J. Grenning, "Planning poker or how to avoid analysis paralysis while release planning", 2002, Dostopno na: http://www.renaissancesoftware.net/files/articles/PlanningPoker-v1.1.pdf.

[7] M. Laanti, O. Salo, P. Abrahamsson, "Agile methods rapidly replacing traditional methods at Nokia: A survey of opinions on agile transformation", *Information and Software Technology*, 2011, št. 3, str. 276-290.

[8] V. Mahnič, "Problemi in rešitve pri uvajanju metode Scrum v proces razvoja programske opreme", *Zbornik referatov Dnevi slovenske informatike*, Portorož, april 2011

[9] Mahnič, V., Hovelja, T., On using planning poker for estimating user stories. J. Syst. Software (2012), http://dx.doi.org/10.1016/j.jss.2012.04.005

[10] K. Molokken-Ostvold, N.C. Haugen, H.C. Benestad, "Using planning poker for combining expert estimates in software projects", *Journal of Systems and Software 81*, 2008, str. 2106-2117.

[11] T. Murphy, J. Duggan, D. Norton, et al., "Predicts 2010: Agile and Cloud Impact Application Development Directions", Gartner, Decemeber 2009, Dostopno na: http://www.gartner.com/DisplayDocument?id=1244514

[12] L. Rising, N. S. Janiff, "The Scrum software development process for small teams", *IEEE Software, 2000, letnik 17, št. 4*, str. 26 - 32.

[13] K. Schwaber, "Agile Project Management with Scrum", Microsoft Press, 2004.

[14] K. Scotland, A. Boutin, "Integrating Scrum with the Process Framework at Yahoo! Europe", *Proceedings of the Agile 2008 Conference*, Toronto, Canada, 2008, str. 191-195.

[15] Več avtorjev, "Agile software development", Wikipedia, Dostopno na: http://en.wikipedia.org/wiki/Agile_software_development

[16] Več avtorjev, "Planning poker", Dostopno na: http://en.wikipedia.org/wiki/Planning_poker

[17] Več avtorjev, "Scrum", Dostopno na: http://en.wikipedia.org/wiki/Scrum_(Development)

[18] VersionOne, "State of Agile Survey 2011", Dostopno na: http://bit.ly/state-of-agile

[19] D. West, T. Grant, "Agile Development: Mainstream Adoption Has Changed Agility", Forrester research, 2010, Dostopno na: http://bit.ly/forrester-agile