Izboljšanje testiranja programske opreme – študija primera

Sašo Greblo
CGS plus, d. o. o., Brnčičeva ulica 13, 1000 Ljubljana
saso.greblo@cgsplus.si

Izvleček

Članek obravnava izboljšanje procesa testiranja programske opreme v življenjskem ciklu programske opreme v majhni družbi, ki se ukvarja z razvojem programske opreme za projektiranje nizkih gradenj. Programska oprema je kompleksna, podpira več jezikov, več platform CAD (Computer Aided Design), različne operacijske sisteme in lokalne standarde, ki so predpisani v posameznih državah. Analiziran je proces razvoja programske opreme s poudarkom na testiranju. Predlogi sprememb se nanašajo na: a) testiranje funkcij, b) testiranje prevedenih programskih modulov ter c) testiranje namestitvenih procedur. Predlog dopolnitve procesa testiranja funkcij vpeljuje strokovni pregled programske kode. Predlog dopolnitve testiranja prevedenih programskih modulov ter testiranja namestitvenih procedur predvideva sistematizacijo postopkov in dokumentov kot osnovni korak v pripravo načrtov testiranja skladno z ISO/IEC/IEEE 29119 ter regresijsko testiranje. Predlog dopolnitve testiranja namestitvenih procedur dodatno predvideva sistematizacijo testiranja namestitvenih procedur v tri nivoje in avtomatizacijo testiranja. Ocenjeno je, da se bo investicija v izdelavo načrtov testiranja, avtomatizacijo testiranja ter vpeljavo dokumentiranja testiranja povrnila v manj kot dveh letih.

Ključne besede: testiranje programske opreme, avtomatsko testiranje programske opreme, optimizacija testiranja programske opreme, (ROI) donosnost naložbe.

Abstract

Improving software testing – case study

The article addresses the improvement of the software testing process in a small company engaged in the development of civil engineering design software. The software is complex, localised into a number of languages, supports multiple CAD platforms, different operating systems as well as local standards. The software development process focusing in particular on testing has been analysed. Prepared were proposals for the improvement of the testing process related to: a) unit testing, b) testing of localised software modules, and c) testing of installation procedures. The proposal for the update of unit testing introduces peer review. The proposal for the update of testing of localised software modules and testing of installation procedures envisages the systematisation of procedures and documents as a prerequisite for the preparation of testing plans in accordance with ISO/IEC/IEEE 29119 as well as regression testing. In addition, the proposal for the update of testing of installation procedures provides for the systematisation of testing of installation procedures into three levels along with the automation of software testing. It is estimated that the investment in the production of test plans, test automation and the introduction of testing documentation will be returned in less than two years.

Keywords: software testing, automated software testing, optimization of software testing, return on investment (ROI).

1 UVOD

Prodajni modeli programske opreme se spreminjajo. Spletna prodaja je vse pomembnejša; pri spletni prodaji programske opreme pa je vse bolj pomembna kakovost programske opreme v primerjavi s klasičnimi prodajnimi kanali, kot so npr. direktna prodaja ali prodaja preko partnerske mreže, pri katerih predprodajne, prodajne in poprodajne aktivnosti potekajo na osebni ravni. Neposredni osebni pristop je v primeru spletne prodaje izgubljen (Greblo, 2016).

V obravnavanem primeru je treba pri razvoju programske opreme za načrtovanje nizkih gradenj upoštevati kar nekaj specifičnih značilnosti, ki so povezane tudi z zahtevami uporabnikov in prilagoditvami programske opreme lokalnim okoljem. Programska oprema je prilagojena delovanju različnih platform CAD proizvajalcev Autodesk in Bricsys. V programsko opremo je vgrajenih 35 tehničnih standardov, ki so predpisani v 17 državah, prevajajo pa jo v petnajst jezikov.

Ohranjanje stika s sodobnimi trendi načrtovanja zahteva konstanten razvoj programske opreme. Ekipa tehnične podpore v podjetju je zaznala povečan obseg dela pri odpravljanju težav, ki so povezane s kakovostjo programske opreme. Odpravljanje zaznanih težav s kakovostjo programske opreme mora biti prioriteta vsakega razvijalca. Podjetje za uspešen razvoj potrebuje zadovoljne obstoječe in nove uporabnike. Težave s kakovostjo programske opreme povzročajo nezadovoljstvo obstoječih uporabnikov in oteženo pridobivanje novih. Analiza procesa razvoja programske opreme je nakazala težave v procesu testiranja. Sredstva, ki jih ima na voljo razvojna ekipa, ostajajo enaka, kljub temu da količina programske kode neprestano narašča. Testiranje ni popolno in podjetje ga mora izboljšati. Velik delež testov opravljajo ad hoc. Dejstvo je, da zaradi omejenih človeških in strojnih virov nikoli ne bo dovolj sredstev za testiranje, zato mora podjetje rezerve iskati drugje.

Postavili smo si cilj, da bomo raziskali možnosti izboljšanja testiranja programske opreme v podjetju. Raziskali smo standarde na področju razvoja in testiranja programske opreme, metodologije razvoja ter postopke in metode testiranja programske opreme.

Na podlagi rezultatov analize procesa razvoja programske opreme s poudarkom na testiranju je pripravljen predlog optimizacije procesa testiranja. Predlog predvideva organizacijske spremembe in implementacijo avtomatskega testiranja v proces testiranja namestitvenih procedur. Predvidena je priprava projekta optimizacije testiranja in izvedba v več fazah. Vodstvo je predlog sprejelo in projekt optimizacije testiranja je v teku. Del predlaganih sprememb se že uporablja.

1 TESTIRANJE PROGRAMSKE OPREME

1.1 Osnovni pojmi in metodologije testiranja programske opreme

1.1.1 Verifikacija in validacija

Verifikacija funkcije pomeni, da se po smiselnem končanju določene faze razvoja preverja uspešnost delovanja funkcije. Validacija funkcije pomeni preverjanje pravilnosti delovanja (Solina, 1997).

1.1.2 Metode bele, črne in sive skrinjice

Poznani sta dve glavni metodi testiranja programske opreme: metoda bele skrinjice in metoda črne skrinjice.

Metoda bele skrinjice testira notranjost zgradbe programa. Potrebno je znanje o programski kodi in arhitekturi programa (Solina, 1997). Testni podatki pri testiranju po metodi bele skrinjice pogosto izhajajo iz programske logike in pogosto zanemarjajo specifikacijo (Myers, Badgett in Sandler, 2011). Metoda črne skrinjice testira pravilnost obnašanja sistema samo s pregledovanjem izhodnih rezultatov. Notranja zgradba programa in struktura kode sta pri tej metodi skrita (Solina, 1997). Metoda črne skrinjice ne odkrije vseh napak, zato lahko uporabljamo komplementarno metodi črne skrinjice metodo sive skrinjice. Metoda sive skrinjice zahteva poznavanje arhitekture in ključnih komponent strukture programa, ki so podlaga pripravi načrta testiranja. Po metodi sive skrinjice lahko tester sistematično oži nabor potencialnih komponent programa, ki so vir napak (Dustin, Garett in Gauf, 2009).

1.1.3 Ad hoc testiranje

Ad hoc testiranje nima pripravljenih načrtov testiranja, testi se ne dokumentirajo in testni primeri se ne shranjujejo (Wigmore, 2012).

1.1.4 Testiranje enot

Enota je najmanjša zaokrožena celota programa. Obravnavano podjetje enoto imenuje funkcija. Največkrat je cilj testiranja funkcije verifikacija. V primeru dobro definiranih zahtev na ravni funkcije lahko funkcijo tudi validiramo. Testiranje enot izvajamo po metodi bele skrinjice (Čebokli, 2006).

1.1.5 Regresijsko testiranje

Tehniko regresijskega testiranja lahko izvajamo na kateri koli ravni testiranja z namenom obvladovanja sprememb programske opreme in preprečevanja napak, ki so enkrat že bile odpravljene. Regresijski testi so obstoječi testi, ki se izvajajo po vsakem popravku v programski opremi. Popravek je lahko posledica dodajanja nove funkcionalnosti ali pa popravljanja odkritih napak (Čebokli, 2006).

1.1.6 ISO/IEC 29119

Standard ISO/IEC/IEEE 29119 definira tehnike načrtovanja testiranja programske opreme, ki jih lahko uporabljamo tako med načrtovanjem kot med implementacijo procesov testiranja (ISO/IEC 29119, 2016).

1.2 Avtomatsko testiranje programske opreme

Nekaterim pomeni pojem avtomatsko testiranje programske opreme razvoj na podlagi načrtov testiranja ali testiranja enot, drugim pa avtomatizacijo testiranja z implementacijo orodij za zajem, snemanje in predvajanje. Avtomatsko testiranje je mogoče v vseh fazah testiranja programske opreme. Vse teste, ki se izvajajo kot del ročnega testiranja, lahko avtomatiziramo. Avtomatsko testiranje krepi prizadevanja ročnega testiranja s poudarkom na tistih testih, ki jih ročno testiranje težko izvaja. Avtomatsko testiranje ne nadomešča potreb po ročnih testerjih z analitičnim znanjem, znanjem priprave testnih strategij in razumevanja tehnik testiranja. Ekspertna znanja ročnih testerjev služijo kot podlaga za pripravo načrta avtomatizacije testiranja. Avtomatskega in ročnega testiranja ni mogoče strogo ločiti. Med seboj se prepletata in dopolnjujeta. Če je avtomatsko testiranje implementirano pravilno, skrajša čas in zniža stroške testiranja, izboljša kakovost programske opreme in poveča učinke ročnega testiranja zaradi povečanja pokritosti testov ter razbremenitve testerjev od monotonega ročnega klikanja (Dustin, Garett in Gauf, 2009).

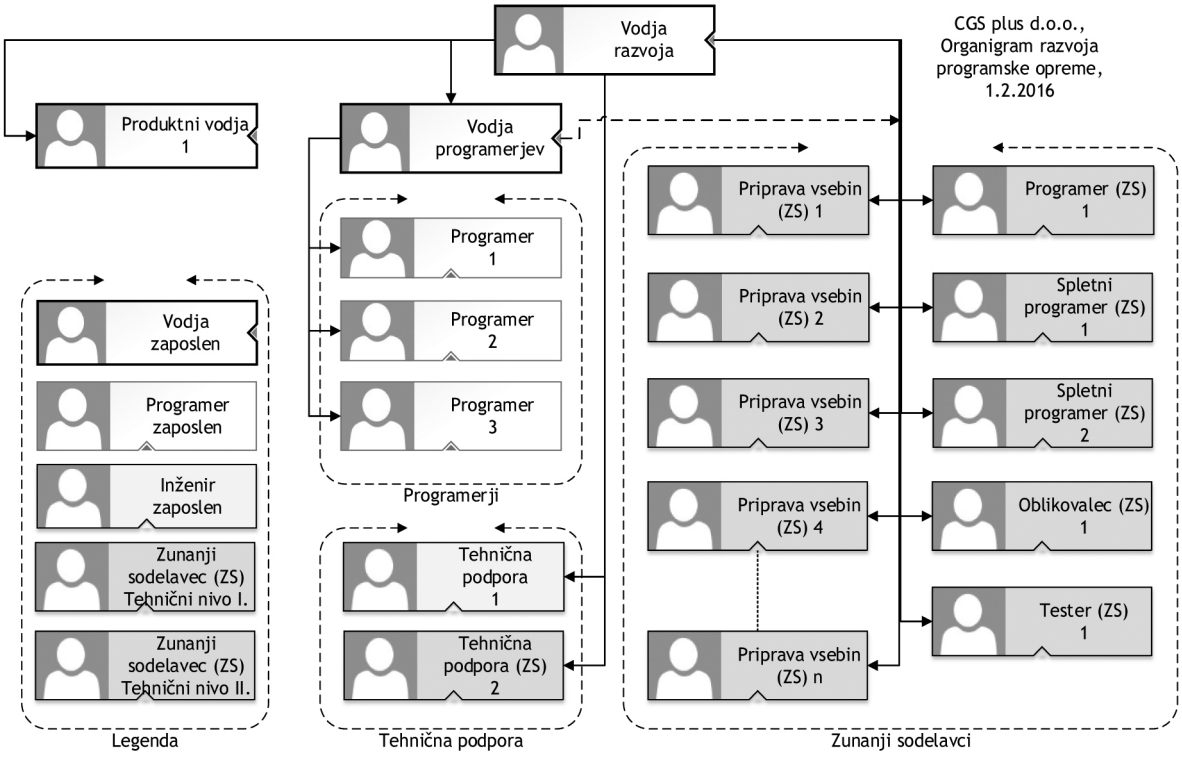
2 ŠTUDIJA PRIMERA

2.1 Omejitve v študiji primera

Podjetje deli programsko opremo na programske rešitve in produkte. Študij primera obravnava tri modularno zasnovane programske rešitve in tri produkte. Plateia je programska rešitev, namenjena načrtovanju in rekonstrukciji cest, Ferrovia načrtovanju in rekonstrukcij železnic, Aquaterra urejanju in načrtovanju vodotokov. Zasnova produktov je ravno tako modularna, produkti pa so manj kompleksni od programskih rešitev in rešujejo specifične funkcionalnosti, ki so sicer vgrajene tudi v programskih rešitvah. Autopath je produkt, namenjen načrtovanju zavijalnih krivulj ter izdelavi analiz prevoznosti. Autosign je produkt, namenjen načrtovanju prometnih ureditev, Aquaflood pa učinkoviti izmenjavi podatkov med okoljem CAD (DWG) in programi MIKE Powered by DHI in HEC-RAS. Metodologija razvoja in testiranja programskih rešitev in produktov je enaka.

2.2 Analiza stanja

Analizirana je organizacijska struktura razvojne ekipe, proces izdaje verzije programske rešitve ali pro-



Slika 1: **Organigram razvojne ekipe**

dukta ter proces testiranja v procesu izdaje verzije programske rešitve ali produkta.

2.2.1 Organizacijska struktura razvojne ekipe

Razvojno ekipo sestavljajo zaposleni in zunanji sodelavci. Organigram razvojne ekipe prikazuje slika 1.

Zunanji sodelavci zagotavljajo določeno fleksibilnost, njihova zahtevana strokovna znanja so različna. Pri manj zahtevnih nalogah je fluktuacija kadra večja. Običajno gre za pripravo vsebin, kot so npr. izdelava in dopolnjevanje knjižnic vozil, prometnih znakov idr. Zunanji sodelavci, ki sodelujejo pri zahtevnejših nalogah, so vezani na trajanje projektov. V procesu testiranja sodelujejo tester 1, vodja razvoja, produktni vodja 1, vodja programerjev in vsi programerji. Za razvoj zagotavljanja kakovosti programske opreme in testiranja je odgovoren vodja razvoja.

2.2.2 Proces izdaje verzije programske rešitve ali produkta

Podjetje izdaja novo verzijo programskih rešitev in produktov enkrat letno. Izidi popravkov verzije so predvideni štirikrat letno. V primeru ugotovljenih kritičnih napak izdaja popravke verzije tudi večkrat.

Proces izdaje verzije programske rešitve ali produkta vsebuje štiri podprocese: programiranje funkcij, prevajanje programske kode, priprava orodnih trakov, menujev in funkcionalnosti ter priprava namestitvenih datotek. Proces izdaje verzije programske rešitve ali produkta prikazuje slika 2.

2.2.3 Testiranje v procesu izdaje verzije programske rešitve ali produkta

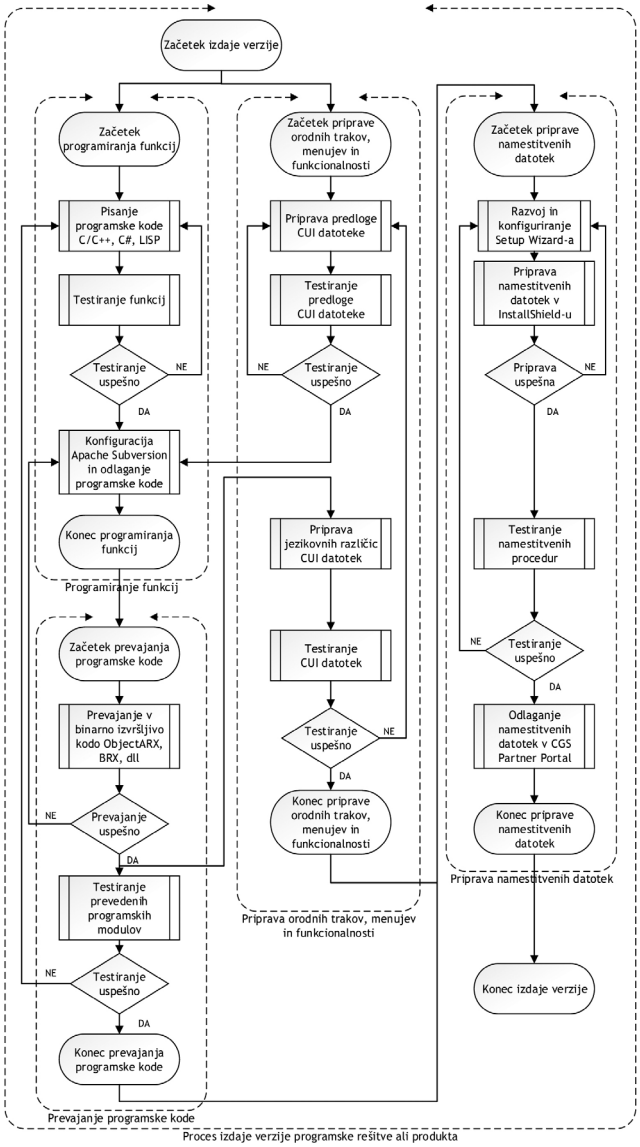
V procesu izdaje verzije so testirane funkcije, predloge datotek CUI (Custom User Interface), datotek CUI, prevedeni programski moduli ter namestitvene procedure. Procesi testiranja v posameznih podprocesih izdaje verzije programske rešitve ali produkta prikazuje slika 3.

2.2.4 Testiranje funkcij

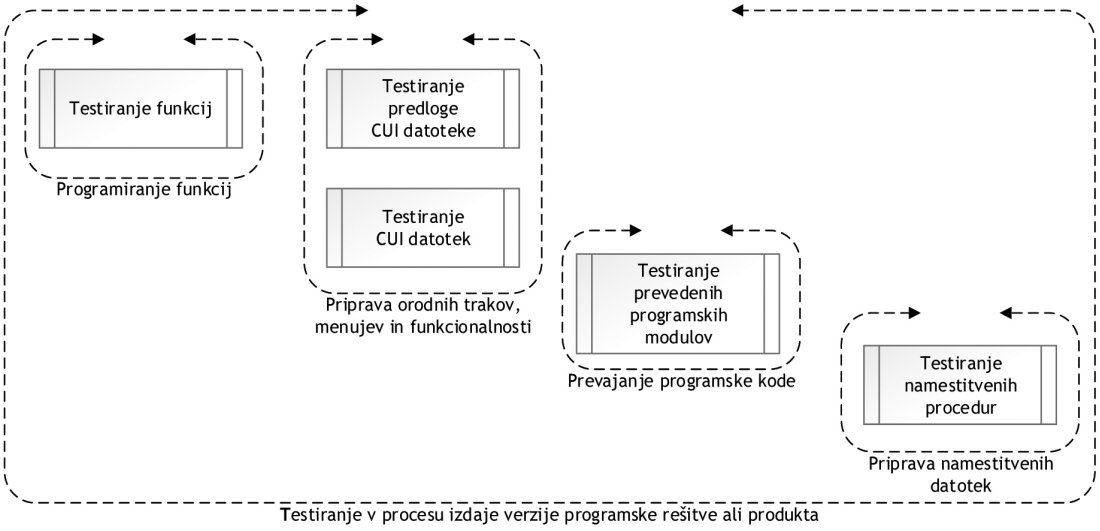
Funkcije testirajo programerji med razvojem. Lastno kodo testirajo ad hoc. Običajno testirajo delovanje funkcije z namenom verifikacije. Testirajo zagon funkcije in vizualno pregledajo pogovorno okno. Občasno testirajo funkcijo tudi z namenom validacije. Pri testiranju ne uporabljajo realnih testnih primerov. Lažje programirajo in testirajo na manjši količini vhodnih podatkov, saj lažje odkrijejo napake, funkcija deluje hitreje in lažje izračunajo pravilni rezultat.

2.2.5 Testiranje predloge in datotek CUI

Predloge datotek CUI pripravljajo vodja razvoja, vodja programerjev in produktni vodja 1 – vsak za programsko rešitev ali produkt, za katerega je zadolžen. V fazi procesa priprave orodnih trakov, menujev in funkcionalnosti testira predlogo in datoteke CUI oseba, ki je pripravila predlogo; testira ad hoc. Datoteke CUI testira z namenom verifikacije. Testira zagon ukazov v menujih in orodnih trakovih. Vizualno pregleda, ali se ob zagonu ukaza odpre pravo pogovorno okno. Preveri pravilnost jezikovne različice, sintakso besed, usklajenost imen med ukazi v menujih in orodnih trakovih idr.



Slika 2: **Proces izdaje verzije programske rešitve ali produkta**



Slika 3: **Procesi testiranja v podprocesih izdaje verzije programske rešitve ali produkta**

2.2.6 Testiranje prevedenih programskih modulov

Prevedene programske module testira tester 1. Testira po metodi črne skrinjice z namenom verifikacije. Testira po navodilih vodje razvoja ali vodje programerjev. Načrt testiranja ni definiran in dokumentiran. Kratka navodila so pripravljena sproti in so posredovana testerju 1 ustno ali v e-poštnem sporočilu. Programsko kodo prevajajo vsak dan, vendar testiranja ne izvajajo vsak dan. Tester 1 testira samo izbrano kodo. Rezultate testiranja dokumentira v Excelovem dokumentu. Struktura dokumenta ni določena. Odkrite napake prijavlja v MantisBT, sistemu za spremljanje in upravljanje programskih napak. Tester 1 pri prijavi klasificira napako ter podrobno opiše in določi stopnjo resnosti. Poda podatke o platformi CAD, operacijskem sistemu, verziji programske rešitve ali produkta, po potrebi priloži datoteko, na kateri je bil opravljen test, in nato objavi napako. Prijavljena napaka dobi svojo identifikacijsko številko. Prijavljene napake v reševanje razporeja vodja razvoja ali vodja programerjev. Popravljeno napako tester 1 ponovno testira. V primeru, da napake ne more več ponoviti, konča zadevo tako, da spremeni status prijavljene napake v sistemu v »closed«. Po končanem testiranju po e-pošti pošlje kratko poročilo in Excelov dokument vodji razvoja. V sporočilu na kratko opiše odkrite in popravljene napake ter kam je shranil testne primere.

2.2.7 Testiranje namestitvenih procedur

Namestitvene procedure testira tester 1 po metodi črne skrinjice. Testira z namenom verifikacije. Vodja razvoja v e-poštnem sporočilu posreduje testerju 1 navodila za testiranje, ki vsebujejo podatke o verziji programske rešitve ali produkta, platformi in operacijskem sistemu, na katerem opravi testiranje ter informacijo o tem, kako podrobno testirati. Testira namestitveno proceduro programske rešitve ali produkta in zagon osnovnih ukazov. Seznam ukazov, ki naj bi jih testiral, običajno ni pripravljen. Vedno opravi tudi vizualni pregled uporabniškega vmesnika in pogovornih oken. Rezultate testiranja dokumentira v Excelovem dokumentu. Struktura dokumenta ni določena. Odkrite napake prijavlja v MantisBT. Po končanem testiranju pošlje po e-pošti kratko poročilo in Excelov dokument vodji razvoja. Programske rešitve in produkti so v več jezikih, delujejo na različnih platformah CAD in operacijskih sistemih ter vsebujejo različne standarde, zato vseh možnih kombinacij namestitvenih procedur ni mogoče testirati. Testiramo samo izbrane verzije v izbranih virtualnih okoljih. Poenostavljeni izračun pokaže, da bi za testiranje vseh 7.002 možnih kombinacij namestitvenih procedur ene verzije potrebovali 14.333 ur. Slika 4 prikazuje možno število namestitvenih kombinacij in potrebni čas testiranja ene verzije. Čas testiranja vsebuje pripravo testnih primerov, namestitev programske opreme ali produkta v že pripravljeno virtualno okolje, testiranje, prijavo napak v MantisBT in dokumentiranje. Za pripravo ene konfiguracije virtualnega okolja z nameščenim operacijskim sistemom in CAD platformo je dodatno potrebno še 1,75 ure in 60 GB razpoložljivega prostora na disku.

Programske rešitve in produkti	Jeziki	Standardi	Operacijski sistemi	CAD platforme	Število možnih kombinacij namestitvenih procedur	Povprečni čas testiranja namestitvene procedure v urah	Skupni čas testiranja v urah
Plateia	11	12	2	9	2.376	4	8.316
Ferrovia	7	6	2	9	756	2	1.512
Aquaterra	6	0	2	9	108	2	162
Autopath	9	14	2	10	2.520	1	2.520
Autosign	6	11	2	9	1.188	2	1.782
Aquaflood	3	0	2	9	54	1	41
					7.002		14.333

Slika 4: **Stevilo možnih kombinacij namestitev z oceno časa testiranja verzije v urah**

2.3 Predlogi izboljšav

Na podlagi analize pripravljeni predlogi optimizacij posameznih procesov testiranja so podani v nadaljevanju.

2.3.1Testiranje funkcij

Predlog optimizacije procesa testiranja funkcij predvideva izvedbo v dveh fazah. V prvi fazi je predlagana dopolnitev obstoječega procesa testiranja po metodi bele skrinjice z vpeljavo strokovnega pregleda programske kode (Peer Review). Težav pri implementaciji podjetje ne pričakuje, čeprav se lahko pojavijo potencialne težave (Wiegers, 2002). Testiranje kode drugega programerja običajno odkrije napake, ki jih programer v lastni kodi ne vidi. Pomemben je tudi vidik, da se s kodo funkcije podrobneje seznanita dva programerja, kar sicer podaljša čas testiranja in posledično zviša strošek testiranja. Predlog optimizacije prve faze je potrjen in je trenutno v fazi izvajanja pilotnega projekta implementacije. Za drugo fazo optimizacije je sprejet predlog priprave projektne naloge implementacije avtomatskega testiranja enot. Projektna naloga je končana. Analizirana je možnost avtomatskega testiranja enot, vendar je sprejeta odločitev, da avtomatizacija glede na trenutni način programiranja ni smiselna.

2.3.2 Testiranje predloge in datotek CUI

Na podlagi analize procesa testiranja je sprejeta odločitev, da spremembe niso smiselne, saj tester 1 sistematično testira datoteke CUI v procesu testiranja namestitvenih procedur.

2.3.3 Testiranje prevedenih programskih modulov

Predlog optimizacije procesa testiranja predvideva izvedbo v treh fazah. Prva faza predvideva pripravo štirih glavnih dokumentov, ki bodo sistematizirali postopke in dokumentiranje. Prvi dokument je protokol testiranja v Wordovi obliki, ki predpisuje potek izvajanja testa določene funkcije, določene jezikovne različice na določeni platformi s predpisanimi vhodnimi parametri. Drugi dokument je protokol poimenovanja datotek v Wordovi obliki, ki predpisuje mesto shranjevanja datotek in dokumentov, poimenovanje map in testnih primerov ter dokumentov, ki nastajajo med testiranjem. Tretji dokument je predloga dokumenta testiranja, strukturirani Excelov dokument. Slika 5 prikazuje primer strukture dokumenta dnevnik testiranja.

2015.11.18 dnevnik testiranja - tester 1										Datum: 18.11.2015	
Začetek testiranja	Konec testiranja	Produkt	Verzija	Jezik	Build	Platforma	OS	x32/ x64	Status	Povezava do poročila o testiranju	Opombe
10.11.2015	11.11.2015	Ferrovia	2016	HUN	495	C3D 2016	Win 7	64	OK	CGSplus%202016\01%20Test	test popravljenih ukazov
12.11.2015	12.11.2015	Plateia	2016	DEU	501	BCAD V15	Win 8.1	64	OK	CGSplus%202016\CGSplus%2	test nivo 1
18.11.2015		Autosign	2016	US	514	C3D 2016	Win 7	64	NOK		

Slika 5: **Struktura dokumenta dnevnik testiranja**

Četrti dokument je predloga dokumenta poročilo o testiranju, strukturirani Excelov dokument. Slika 6 prikazuje primer strukture dokumenta poročilo o testiranju.

Prva faza optimizacije je potrjena in končana. Za drugo fazo optimizacije je potrjen predlog priprave projektne naloge priprave načrta testiranja skladno s standardom ISO/IEC/IEEE 229119. Realizacija priprave projektne naloge je bila predvidena v letu 2016, izvedba pa v letu 2017. Za tretjo fazo optimizacije je sprejet predlog priprave projektne naloge implementacije avtomatskih regresijskih testov v podprocesu prevajanja programske kode. Realizacija priprave projektne naloge in realizacija pilotnega projekta je bila predvidena v letu 2016, nadaljevanje projekta pa v letu 2017.

2.3.4 Testiranje namestitvenih procedur

Predlog optimizacije testiranja namestitvenih procedur predvideva izvedbo v treh fazah. Prva in druga faza sta dopolnjena predloga optimizacije procesa testiranja prevedenih programskih modulov. V prvi fazi so dodatno predpisani trije nivoji testiranja.

- Nivo 1 predpisuje testiranje namestitve programske rešitve ali produkta ter glavnih ukazov. Seznam ukazov pripravi vodja razvoja na podlagi strukturirane predloge dokumenta seznam ukazov za testiranje – nivo 1.xls.
- Nivo 2 predpisuje kompletno testiranje po specifikaciji nivoja 1 in dodatno še testiranje namestitev posodobitev programske opreme ali produktov in razširjen nabor ukazov. Seznam ukazov pripravi vodja razvoja na podlagi strukturirane predloge dokumenta seznam ukazov za testiranje – nivo 2.xls.
- Nivo 3 predpisuje kompletno testiranje po specifikaciji nivoja 2 in dodatno še testiranje vseh ukazov. Seznam ukazov pripravi vodja razvoja na podlagi strukturirane predloge dokumenta seznam ukazov za testiranje – nivo 3.xls.

2016.01.07 poročilo o testiranju - tester 1					
Autosign - testiranje ukazov					
Splošno	**Produkt**	**Platforma**	**OS**	**OS version**	**Datum testiranja**
1	Autosign DEU	C3D 2015	Win 8.1	x64	07.01.2016
Autosign DEU C3D 2015					
Test Id	**Ukaz**	**Vhodni podatki (risba)**	**Status**	**Komentar**	**MantisBT**
101	7 Extract Traffic Signs	Autosign_DEU_C3D2012.dwg	NOK	V primeru skupne uporabe CGSplus in Autosign slog pisave v tabeli napačen.	http://www.cgs.si/mantis/view.php?id=3264
102	2 Edit Traffic Signs	Autosign_DEU_C3D2012.dwg	OK		

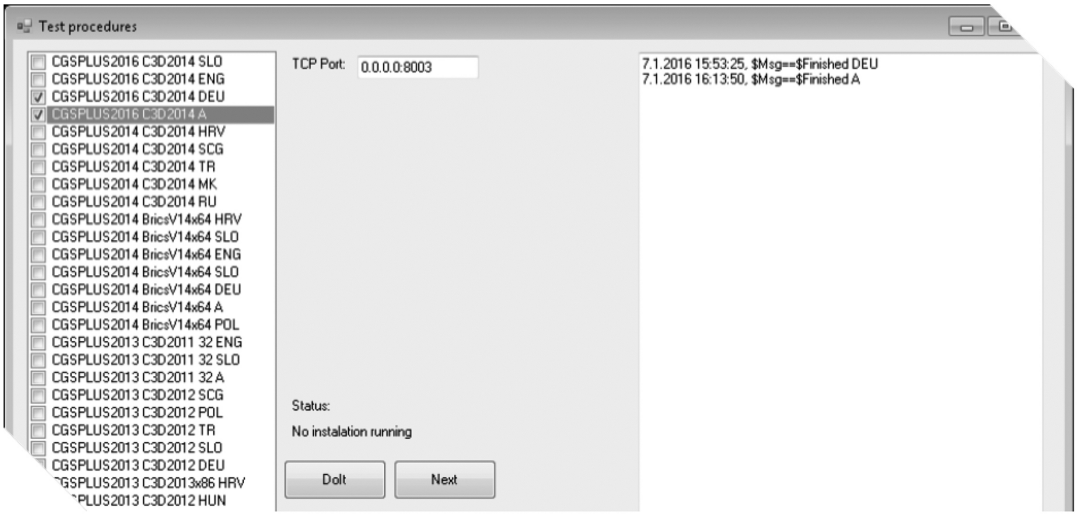
Slika 6: **Struktura dokumenta poročilo o testiranju**

2015.9.7. seznam ukazov za testiranje - nivo 3 - Plateia					
21 Plateia OSI					
Številka builda	**Namestitvena datoteka**	**OS version**	**OS**	**Platforma**	**Datum**
34	CGSplus 2015 ENG 64-bit Build 34	x64	Win 7	C3D 2016	7.9.2015
Plateia 2015					Ukaz
A Projekt	1 Projekt				A1
	2 Nastavitve	1 Nastavitev spremenljivk Osi			A21
		2 Nastavitev delovanja			A22
		3 Imena risalnih ravnin			A23
B Pomožni elementi	1 POMOŽNI ELEMENTI VZDOLŽNE OSI				B1
	2 Paketni vnos pomožnih elementov <-				B2
	3 Izris pomožnih elementov na osnovi				B3
....	....				....

Slika 7: **Struktura dokumenta seznam ukazov za testiranje**

Avtomatizacija testiranja namestitvenih procedur je predlagana v tretji fazi. Prva faza je končana. Realizacija priprave projektne naloge druge faze je bila predvidena v letu 2016, izvedba projektne naloge pa v letu 2017. Podjetje je končalo tretjo fazo izvajanja projekta optimizacije. Analiza procesa razvoja programske opreme je za avtomatizacijo nakazala ugodne pogoje. Programske rešitve in produkti imajo v aktualni verziji skupaj 1.329 ukazov. Pri prehodu na novo verzijo je običajno odstotek ukazov novih, odstotek ukazov dopolnjenih, medtem ko 98 odstotkov ukazov ostaja enakih. Projektna naloga avtomatizacije testiranja namestitvenih procedur narekuje izbiro programskega orodja za podporo avtomatskemu testiranju ter izvedbo pilotnega projekta avtomatizacije in je končana. Za izdelavo avtomatskih testnih procedur je izbrana odprtokodna programska rešitev QAliber Test Builder. Ta ne omogoča upravljanja virtualnih okolij, zato je podjetje razvilo lasten vmesnik CGS za konfiguracijo in upravljanje virtualnih okolij. Del avtomatske testne procedure se pripravi v vmesniku CGS. Pripravijo se različne kombinacije konfiguracij operacijskega sistema, npr. Windows 7, 64 bit, platforme CAD, npr. AutoCAD Civil 3D 2014, 64 bit, programske rešitve ali produkta, npr. Plateia 2016, SLO, 64 bit in testnega scenarija. Po izbranih konfiguracijah ukaz zažene QAliber Test Runner, predvajalnik testnih skript, pripravljenih v programu QAliber Test Builder. V okviru pilotnega projekta je bilo pripravljenih 41 konfiguracij. Pogovorno okno vmesnika CGS s seznamom konfiguracij ter dvema konfiguracijama, pripravljenima za zagon v programu QAliber Test Runner, prikazuje slika 8.

V okviru pilotnega projekta je bila izvedena avtomatizacija testiranja namestitvenih procedur na nivoju testiranja 1. V okviru pilotnega projekta so bile izvedene meritve in opravljena analiza avtomatizacije testiranja nivoja 1. Čas priprave virtualnih okolij, dokumentiranja in prijavljanja napak v MantisBT v meritvah ni upoštevan. Podatki, pridobljeni na podlagi meritev in izračunov, so prikazani v slikah 9 do 12. Časi priprave in izvedbe ročnega testiranja namestitvenih procedur nivojev 1, 2 in 3 so prikazani v sliki 9.



Slika 8: **Pogovorno okno vmesnika CGS**

Ročno testiranje	Nivo 1	Nivo 2	Nivo 3
Programske rešitve, produkti	Povprečni čas testiranje namestitvene procedure v urah	Povprečni čas testiranje namestitvene procedure v urah	Povprečni čas testiranje namestitvene procedure v urah
Plateia	2,83	4,17	20,5
Ferrovia	1,5	2,33	15
Aquaterra	1	1,33	13
Autopath	0,58	1	2,17
Autosign	1	1,33	4
Aquaflood	0,33	0,92	2,17

Slika 9: **Časi priprave in izvedbe ročnega testiranja nivojev 1, 2 in 3 v urah**

Časi priprave avtomatskih testnih procedur za nivo 1 so prikazani v sliki 10.

Časi izvedbe avtomatskega testiranja namestitvenih procedur nivoja 1 so prikazani v sliki 11.

Simulacija časov priprave in izvedbe n avtomatskih testov v primerjavi z izvedbo n ročnih testov za nivo 1 prikazuje slika 12.

2.3.5 ROI implementacije avtomatizacije testiranja nivoja 1

V izračunu ROI so upoštevani časi priprave in izvajanja ročnih in avtomatskih testov. Izračun pokaže, da se investicija v implementacijo avtomatizacije testiranja nivoja 1 povrne po 10 opravljenih testih za Plateio, 16 za Ferrovio, 21 za Aquaterro, 19 za Autopath, 5 za Autosign in 51 za Aquaflood. ROI po izvedbi n testov testiranja nivoja 1 je prikazan v sliki 13.

V podjetju so se na podlagi spodbudnih rezultatov ROI odločili, da bodo v letu 2016 nadaljevali projekt avtomatizacije testiranja nivoja 2. Zaradi določenih težav z izbranima orodjema QAliber Test Builder in QAliber Test Runner med pilotnim projektom so pred nadaljevanjem opravili ponovno presojo o izbiri primerne programske rešitve za podporo avtomatskemu testiranju.

Priprava avtomatskih testnih procedur - nivo 1	Plateia	Ferrovia	Aquaterra	Autosign	Autopath	Aquaflood
Konfiguracija CGS vmesnika	0,25	0,25	0,25	0,25	0,25	0,25
Posegi v skripto	7,50	6,00	4,50	3,00	1,50	2,50
Priprava, pisanje in snemanje testnih scenarijev	14,00	10,50	8,00	3,00	1,50	2,50
SKUPAJ nivo 1 (ur)	**21,75**	**16,75**	**12,75**	**6,25**	**3,25**	**5,25**

Slika 10: **Časi priprave avtomatskih testnih procedur za nivo 1 v urah**

Izvedba avtomatske testne procedure - nivo 1	Plateia	Ferrovia	Aquaterra	Autosign	Autopath	Aquaflood
Zagon virt. okolja in namestitev op. sistema	0,03	0,03	0,03	0,03	0,03	0,03
Kopiranje namestitvene datoteke	0,01	0,01	0,01	0,01	0,01	0,01
Namestitvena procedura	0,13	0,13	0,13	0,13	0,13	0,13
Izvajanje testne procedure (povp. 30 s na ukaz)	0,35	0,26	0,20	0,08	0,04	0,06
SKUPAJ nivo 1 (ur)	**0,51**	**0,43**	**0,36**	**0,24**	**0,20**	**0,23**

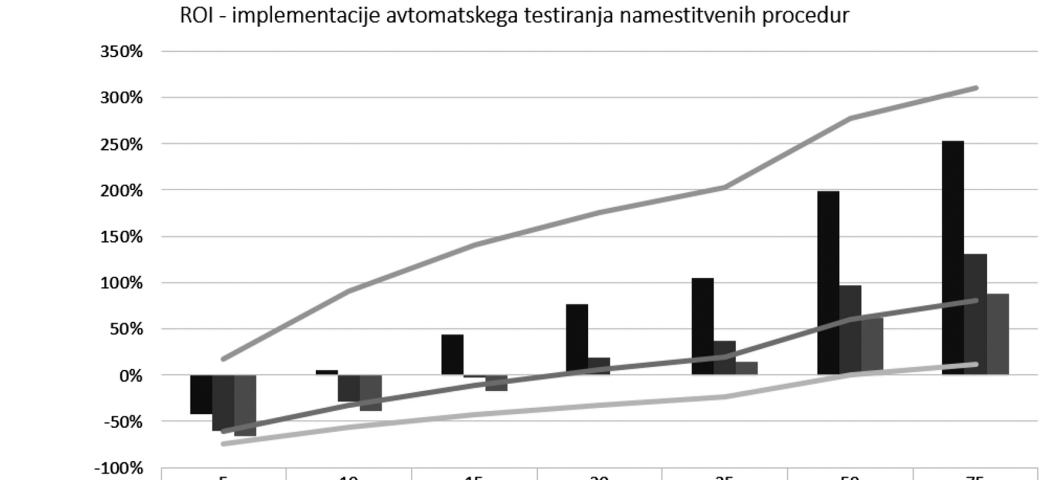
Slika 11: **Časi izvedbe avtomatskega testiranja nivoja 1 v urah**

Testiranje namestitvenih procedur - nivo 1	Plateia	Ferrovia	Aquaterra	Autosign	Autopath	Aquaflood
Priprava avtomatskih testnih procedur	21,75	16,75	12,75	6,25	3,25	5,25
Izvedba avtomatskega testa	0,51	0,43	0,36	0,24	0,20	0,23
Izvedba n avtomatskih testov	12,81	8,20	10,76	9,74	2,56	11,48
Skupaj čas - avtomatsko testiranje (ur)	**34,56**	**23,55**	**20,36**	**10,76**	**4,25**	**16,73**
Izvedba ročnega testa	2,83	1,50	1,00	0,58	1,00	0,33
Izvedba n ročnih testov	70,75	24,00	21,00	11,02	5,00	16,83
Skupaj čas - ročno testiranje (ur)	**70,75**	**24,00**	**21,00**	**11,02**	**5,00**	**16,83**
Število izvedenih testov - n	**25**	**16**	**21**	**19**	**5**	**51**

Slika 12: **Časi priprave in izvedbe n avtomatskih in ročnih testov**

3 SKLEP

V podjetju poteka razvoj programskih rešitev in produktov po metodologiji dobave po fazah (Staged delivery). Značilnost metodologije je, da razstavi večji projekt na obvladljive zaporedne faze. Podrobno načrtovanje in ocena stroška razvoja sta možna samo do naslednje faze, končni izdelek je sestavljen po modulih, vsaka faza je zaključena celota, ki je lahko testirana. V življenjskem ciklu razvoja programske opreme podjetja tipično namenjajo 40 odstotkov časa analizi in načrtovanju, 40 odstotkov testiranju in 20 odstotkov programiranju (Solina, 1997). Obravnavano podjetje namenja 40 odstotkov časa analizi in načrtovanju, 45 odstotkov programiranju in samo 15 odstotkov testiranju. Delež časa, namenjen testiranju, je majhen, zato je pomembno, da je izkoriščen optimalno. Edina možnost, da podjetje poveča obseg testiranja programske opreme in se izogne povišanju stroškov, je zvišanje produktivnosti testiranja. Proces razvoja programske opreme je analiziran s poudarkom na testiranju. Predlogi sprememb izboljšujejo procese testiranja funkcij, testiranja prevedenih pro-



	5	10	15	20	25	50	75
ROI Plateia	-42%	5%	44%	77%	105%	199%	253%
ROI Ferrovia	-60%	-29%	-3%	19%	37%	97%	131%
ROI Aquaterra	-66%	-39%	-18%	0%	15%	62%	88%
ROI Autopath	-61%	-33%	-11%	5%	19%	60%	81%
ROI Autosign	18%	90%	140%	176%	203%	277%	311%
ROI Aquaflood	-74%	-56%	-43%	-32%	-24%	0%	12%

ŠTEVILO IZVEDENIH TESTNIH PROCEDUR

Slika 13: **ROI izvedbe n = 5, 10, 15, 20, 25, 50, 75 testov nivoja 1**

gramskih modulov ter testiranja namestitvenih procedur. Predlog dopolnitve procesa testiranja funkcij je vpeljava strokovnega pregleda programske kode (Peer Review). Predlog dopolnitve testiranja prevedenih programskih modulov predvideva sistematizacijo postopkov in dokumentov kot temeljni korak v pripravo načrtov testiranja skladno z ISO/IEC/IEEE 29119 ter regresijsko testiranje. Predlog dopolnitve testiranja procesa namestitvenih procedur zajema poleg sistematizacije postopkov in dokumentov sistematizacijo testiranja v treh nivojih. V nadaljevanju predvideva pripravo načrtov testiranja skladno z ISO/IEC/IEEE 29119 in avtomatizacijo testiranja. Vodstvo podjetja je podprlo predlog optimizacije procesov testiranja, ki predvideva več faz implementacije sprememb. Izvedba faz je odvisna tudi od rezultatov predhodnih projektov. Določene faze optimizacije so končane in rezultati so spodbudni. Izračun ROI izvedenega pilotnega projekta implementacije avtomatskega testiranja namestitvenih procedur nivoja 1 izkazuje vračilo investicije v avtomatizacijo testiranja posameznih programskih rešitev in produktov v roku 6 do 24 mesecev. Projekt optimizacije testiranja še vedno aktivno poteka.

4 LITERATURA

[1] Čebokli, P. (2006). *Avtomatizacija testiranja kot ključ agilnosti razvoja programske opreme.* Magistrsko delo. Ljubljana: Fakulteta za računalništvo in informatiko.

[2] Dustin, E., Garett, T., Gauf, B. (2009). *Implementing Automated Software testing: How to Save Time and Lower Costs While Raising Quality.* Boston: Pearson Education.

[3] Greblo, S. (2016). *Izboljšanje testiranja v življenjskem ciklu programske opreme.* Magistrsko delo. Kranj: Fakulteta za organizacijske vede.

[4] International Organization for Standardization. (2013). *ISO/IEC/IEEE 29119-1:2013: Software and systems engineering - Software testing - Part 1: Concepts and definitions.* Pridobljeno 13. 2. 2016 na http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=45142.

[5] Myers, G. J., Badgett, T., Sandler, C. (2011). *The Art os Software testing.* New Jersey: John Wiley & Sons.

[6] Solina, F. (1997). *Projektno vodenje razvoja programske opreme.* Ljubljana: Fakulteta za računalništvo in informatiko.

[7] Wiegers, E. K. (2002). *Peer Reviews in Software: A Practical Guide.* Boston: Addison-Wesley.

[8] Wigmore, I. (2012). *Ad hoc testing.* Pridobljeno 5. 1. 2016 na http://whatis.techtarget.com/definition/ad-hoc-testing.

■

Sašo Greblo je magister informatike in kot vodja prodaje zaposlen v podjetju CGS plus d.o.o.. Njegovo strokovno izpopolnjevanje poteka v okviru podjetja ter sodelovanjem s poslovnimi partnerji, kot so Autodesk, HP in podobno. Svoje strokovno znanje in ugotovitve implementira v okviru podjetja v okviru različnih razvojnih projetkov.