

D. Čubranić, M. Kaluža, J. Novak: Standardne metode u funkciji razvoja softvera u Republici Hrvatskoj
Zbornik Veleučilišta u Rijeci, Vol. 1 (2013), No. 1, pp. 239-256

Dominik Čubranić¹

Marin Kaluža²

Jadranko Novak³

Stručni rad

UDK 65.012.12:004.413>(479.5)

STANDARDNE METODE U FUNKCIJI RAZVOJA SOFTVERA U REPUBLICI HRVATSKOJ⁴

SAŽETAK

Razvoj softvera složen je posao koji se sastoji od velikog broja koraka, počevši od početnog zahtjeva korisnika, preko analize i izgradnje, pa do implementacije i korištenja. Razvoj softvera podrazumijeva korištenje većeg broja različitih tehnologija i tehnika. Postoje zakonitosti, pravila i redoslijed u korištenju tehnologija i tehnika, pa nije moguće razviti dobar, koristan i pouzdan softver, ako se njegovom razvoju pristupa ad-hoc. Da bi se osigurala pouzdanost i kvaliteta softvera potrebno je koristiti metodike u njegovom razvoju. Metodike propisuju vrijeme i mjesto korištenja metoda. Radom će se pokazati neke metodike razvoja softvera. Da bi se osigurala brzina, učinkovitost i točnost razvoja softvera, te jednostavnost održavanja, potrebno je u razvoju softvera koristiti CASE tehnologije. Radom će se analizirati značajke CASE tehnologija. CASE tehnologije omogućuju standardizaciju u razvoju softvera i primjenu normi. Dat će se osvrt na neke standarde i norme koji se primjenjuju u razvoju softvera. U radu će se analizirati podaci Hrvatske gospodarske komore o broju tvrtki i broju zaposlenih u Republici Hrvatskoj za djelatnost računalno programiranje. Iz istog izvora koristit će se podaci o adresama elektroničke pošte radi provedbe anketnog upitnika o upotrebi metoda i standarda u razvoju softvera. Cilj istraživanja je pokazati kvantitetu i kvalitetu (opisnost, pokrivenost faza razvoja softvera) najčešće korištenih tehnika, metoda i standarda.

Ključne riječi: CASE alati, razvoj softvera, razvoj softvera u Hrvatskoj, softverski inženjering

1. UVOD

U današnjem informacijskom dobu, u svijetu tehnike, računala i komunikacija, softver je jedno od najvažnijih dostignuća ljudskog stvaralaštva. Softver je svakim danom sve složeniji zbog neprestanog razvoja sustava koji ga koriste. Kao neizostavna činjenica nameće se problematika razvoja softvera koji će biti isporučen na vrijeme, sa što manje otkrivenih pogrešaka i sa što zadovoljnijim naručiteljima i kupcima. Ova problematika nameće, kako u prijašnjim godinama razvoja složenog softvera, a tako i u današnje doba, korištenje odgovarajućih procesa za razvoj softvera, nasuprot neorganiziranom i nesređenom razvoju.

¹ Stručni spec. inf., projektant/programer, Mapro d. o. o., Bože Vidasa 20, Rijeka, Hrvatska. E-mail: dominik@mapro.hr

² Mr. sc., viši predavač, Veleučilište u Rijeci, Vukovarska 58, Rijeka, Hrvatska. E-mail: mkaluza@veleri.hr

³ Mr. sc., direktor, Caminus d. o. o., Krešimirova 20, Rijeka, Hrvatska. E-mail: jadranko.novak@caminus.hr

⁴ Datum primitka rada: 31. 1. 2013.; datum prihvaćanja rada: 3. 4. 2013.

Kakva je kvaliteta softvera dostupnog na tržištu? Je li se u njegovu razvoju pristupilo inženjerski? Jesu li se u postupku razvoja softvera koristile metode i tehnike s aktualnim tehnologijama razvoja? Hipotetski se može reći da se razvoj softvera u domaćim tvrtkama provodi inženjerskim pristupom i prema aktualnim svjetskim trendovima.

Radom će se pokazati je li razvoj softvera na domaćem tržištu metodološki potpomognut i koje su često korištene metode, tehnike i tehnologije u razvoju. Pokazat će se što utječe na odabir metoda, alata i postupaka za razvoj softvera u tvrtkama kojima je proizvodnja softvera osnovna djelatnost.

Analizirat će se dostupne metode, metodologije, metodike i standardi za razvoj softvera, te provjeriti njihovu primjenu u domaćim tvrtkama za razvoj softvera.

2. SOFTVERSKI INŽENJERING

Strukturu softverskog inženjeringa sačinjavaju tri osnovne komponente: metode, alati i postupci (procedure). Njihovo jedinstvo definira kvalitetu razvoja, pa je zbog toga značajno da se odaberu one komponente koje se postavljenim zadacima i problemima u razvoju najlakše prilagođavaju (Sommerville, 1992). Donošenje prave odluke nije jednostavan zadatak, zbog individualnog i kreativnog karaktera postupka razvoja, zbog različitosti pojedinih sustava za koji se razvija softver i različitog okruženja sustava, ali i zbog nepostojanja „receptata“ za izbor.

Metode pokrivaju širok spektar zadataka među kojima su: planiranje i procjenjivanje projekata, analiza sistemskih i softverskih zahtjeva, projektiranje strukture podataka, definiranje arhitekture programa, kodiranje, testiranje i održavanje. Izabrali odgovarajuću metodu koja će se primijeniti u razvoju nije jednostavno, s obzirom na to da su raspoložive brojne mogućnosti. Metode se mogu međusobno kombinirati nezavisno od toga koja je metoda najpogodnija za rješavanje danog problema. Najznačajnija težnja prilikom pravilnog izbora metoda je minimalizacija troškova razvoja i osiguranje visoko kvalitetnog proizvoda.

Alati osiguravaju automatiziranu ili poluautomatiziranu podršku u primjeni metoda. Oni predstavljaju pomoć neophodnu da bi se sistematizirale i što je više moguće automatizirale aktivnosti razvoja softvera kao: upravljanje projektom odnosno planiranje, procjenjivanje, raspoređivanje, modeliranje, analiza, projektiranje, kodiranje, dokumentiranje, testiranje, integracija elemenata sa sustavom, upravljanje konfiguracijom, kontrola kvalitete softvera, upravljanje podacima i dr. Danas gotovo svaka metoda posjeduje određeno pomoćno sredstvo, instrument, alat. Kada su alati na takav način integrirani u jedan sustav da se rezultati stvoreni jednim alatom mogu upotrijebiti i od drugog alata, tada su to CASE alati (Bachman, 1988).

Postupci ili procedure su veza između metoda i alata. Postupkom se naziva niz konkretnih koraka koje je potrebno izvršiti prilikom rješavanja danog problema ili grupe problema primjenom određene metode. Postupak može imati alternativu (ili opcije), a međusobnim

povezivanjem više postupaka mogu se kreirati i potpuno novi postupci. Za primjenu određenog postupka također se ne može dati recept, jer se priroda sustava koji se razvija i očekivanja korisnika međusobno značajno razlikuju.

2.1 Životni ciklus softvera

Razvoj softvera predstavlja ciklus aktivnosti u razvoju, korištenju i održavanju softvera. Tijekom života, softver prolazi kroz više faza razvoja: početka odnosno inicijalnog dijela razvoja, specifikacije potreba, izgradnje i validacije, produktivnog funkcioniranja, održavanja (Baik, 2000), te povlačenja (gašenja) sustava. Životni ciklus razvoja softvera može se okarakterizirati sljedećim aktivnostima:

- Inicijalizacija sustava je aktivnost u kojoj se navodi podrijetlo softvera.
- Analiza i specficiranje zahtjeva je aktivnost u kojoj se identificiraju problemi koje je potrebno riješiti novim softverom.
- Specifikacija funkcija je aktivnost u kojoj se identificiraju i formaliziraju podaktivnosti definiranja predmeta obrade, identificiranje atributa i veza objekata i operacija.
- Strukturiranje i izbor dijelova su aktivnosti kojima se na osnovi identificiranih zahtjeva i specifikacije funkcija strukturira softver na takve dijelove kojima se može upravljati, a koji predstavljaju logičke cjeline.
- Specifikacija strukture je aktivnost u kojoj se definiraju međusobne veze između dijelova strukture i sučelje između modula sustava.
- Specifikacija detaljnih komponenti dizajna je aktivnost u kojoj se definiraju procedure putem kojih se izvori podataka svakog pojedinog modula transformiraju iz potrebnih ulaza u zahtijevane izlaze.
- Implementacija komponenti i otklanjanje nedostataka je aktivnost u kojoj se kodiraju dizajnirane procedure i procesi i pretvaraju u izvorni kod.
- Integracija i testiranje softvera je aktivnost koja potvrđuje i održava cjelokupnu integralnost komponenti softvera putem verifikacije konzistentnosti i kompletnosti uvedenih modula.
- Provjera dokumentacije i uvođenje softvera je aktivnost koja obuhvaća izradu systemske dokumentacije i uputa za korisnika.
- Obuka i upotreba je aktivnost koja osigurava korisnicima softvera instrukcije i upute za razumijevanje mogućnosti i ograničenja u cilju uspješne upotrebe sustava.
- Održavanje softvera je aktivnost koja podržava operacije sustava u cilnom okruženju na način da osigura potrebna unapređenja, proširenja, popravke, zamjene i dr.
- Gašenje softvera (povlačenje iz primjene) je posljednja aktivnost u životnom ciklusu.

2.2 Metodologije

Razvijene su različite metodologije koje naglašavaju različite pristupe u razvoju softvera. Opstale su i danas se koriste one koje su se sa svojim praksama pokazale dovoljno kvalitetnima i prilagodljivima da mogu odgovoriti na rastuće probleme u današnjem razvoju softvera. Mogu se izdvojiti tradicionalne, procesno orijentirane metode tradicionalnih pristupa u razvoju softvera te agilne, relativno nove i manje opsežne metode usredotočene na male jedinice posla i s naglaskom na vrijednostima i principima umjesto na procesu.

Samo jedna metodologija razvoja softvera ne može djelovati na čitavom spektru različitih projekata. Sve metodologije i nisu primjenjive u svim projektima. Usprkos tomu, projektni menadžment treba identificirati specifičnu prirodu projekta te odabrati najprikladniju razvojnu metodologiju. Zato se i razvila potreba za novim, agilnim metodologijama i za procesno-orijentiranim metodologijama razvoja softvera.

Metodologije razvoja softvera mogu se općenito podijeliti u dvije glavne skupine (<http://www.martinfowler.com/articles/newMethodology.html>):

1. Teške i opsežne metodologije razvoja softvera (engl. *heavyweight methodologies*) sadrže mnogo pravila, načina postupanja i dokumentacije. Traže vrijeme i disciplinu za ispravno slijedenje. Nazivaju se i „debelim“ metodama (engl. *thick methods*).
2. Lakše i manje opsežne metodologije razvoja softvera (engl. *lightweight methodologies*) obično sadrže tek nekoliko pravila i načina postupanja koji su lagani za slijedenje. Nazivaju se i „tankim“ metodama (engl. *thin methods*).

Cilj razvoja softvera je proizvodnja kvalitetnog softvera, u zadanom vremenu, unutar predviđenog budžeta i uz zadovoljavanje stvarnih potreba naručitelja. Uspjeh projekta ovisi o dobrom upravljanju zahtjevima (engl. *requirement management*).

Metodologija odabire metode, prilagođava ih konačnom cilju, propisuje redoslijed upotrebe metoda, propisuje proces modeliranja od početka do kraja životnog ciklusa softvera. Cjelovit pristup razvoju softvera podrazumijeva upotrebu metodologije koja u sebi sadrži metode primjenjive u pojedinim fazama razvoja softvera, tako da je svaka faza pokrivena s jednom ili više metoda. Ideja cjelovitosti pristupa razvoju informacijskih sustava dovela je do nastanka niza metodologija i metodika, kao što su:

- SPIS (engl. *strategic planning of information systems*) (Lederer, Sethi, 1988),
- MIRIS (*metodika za razvoj informacijskih sustava*) (Pavlić, 1996),
- IE (*information engineering*),
- TAFIM (engl. *technical architecture framework for information management*) (McLaughlin 1996),
- OOIE (engl. *object oriented information engineering*) (Montgomery, 1994),
- AD/Cycle (engl. *application development cycle*) (Mercurio et al., 2006),
- SDM (engl. *system development methodology*) (Bolton et al., 2006),
- BSP (engl. *business system planning*).

Svaka metodologija nastoji svojim metodama pokriti što više faza u razvoju softvera. Neke od njih koriste slične metode (MIRIS, AD/Cycle, BSP), neke imaju naglasak na višim (početnim) fazama razvoja (SPIS), a neke na nižim (kasnijim) fazama u razvoju softvera (AD/Cycle, MIRIS).

2.3 CASE tehnologije

CASE tehnologije pokrivaju područje od pojedinačnih alata za automatizaciju određenih zadataka u razvoju softvera do cjelovitih rješenja za automatizaciju izrade većine koraka u razvoju softvera kao cjeline. CASE tehnologije ne predstavljaju zamjenu za bilo koju metodu ili tehniku razvoja, već samo dodatak metodi ili tehnici u generiranju kvalitetnog proizvoda. Njihovo korištenje je interaktivno, prilagođeno korisniku (razvojnom inženjeru) uz naglasak na upotrebi grafike.

Tablica 1. Popis najraširenijih CASE tehnologija

| CASE tehnologija | Metodologije i razvojni alati | Sustavi upravljanja bazama podataka |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| Promod PLUS (http://www.gesys.com/pro-mod.htm) | Yourdon, de Marco, Hatley/Pirbhai; Uniface Six, Source Pilot, C | Sybase, Oracle, Informix, Ingres |
| Oracle Designer (http://www.oraFAQ.com/wiki/Designer) | J. Martin, de Marco, Ernst & Young; Uniface Six, CASE Generator SQL, Forms | Oracle, DB2 |
| Westmount I-CASE (http://www.unl.csi.cuny.edu/faqs/software-engineering/blurb/Westmount.html) | Ward-Mellor, de Marco, Chen, SSADM; Ingres 4GL, Informix 4GL | Ingres, Informix, SQL |
| PTECH (http://en.wikipedia.org/wiki/Ptech) | Martin-Odell, OOAD; ugrađeni C++ generator koda | OODBMS |
| Paradigm Plus (http://www.encyclo.co.uk/define/Paradigm%20Plus) | Rumbaugh OMT, Martin-Odell, OOIE, Booch OOAD, Coad, Yourdon; ProtoScript, C, C++, Ada, SmallTalk, PowerBuilder, SQL, JAVA, Corba IDL, Visual Basic, Visual Studio | ORACLE 7, dBase, DB2, uniSQL, Access, Centura, SQLBase, Sybase/SQL, objectStore, gemStone |
| Rational CASE family (http://www-01.ibm.com/software/rational/) | Rumbaugh OMT, Booch OOAD, Jacobson Objectory Use Case, UML;C, C++, Forte, Java, SmallTalk, PowerBuilder, Gupta, SQLWindows, VisualBasic | Oracle 7, Sybase, SQLBase, SQLServer, Watkom SQL, Ansi SQL |
| CA ERwin Modeling (http://erwin.com/) | IDEF1X, IE, DM | DB2, IDS (Informix), MySQL, Oracle, Progress,SQL Server, Sybase, Teradata |
| PowerDesigner (http://www.sybase.com/products/modelingdevelopment/powerdesigner) | BPEL4WS, BPMN, DTD, ebXML, IDEF, RDBMS, Rich Text Format (RTF), UML 2.0 diagrams, XML, XML Schema | DB2, Informix SQL, Ingres, Interbase, MS Access, MS SQL, MySQL, NonStop SQL, Oracle, PostgreSQL, Sybase, Teradata |
| Enterprise Architect (http://www.sparxsystems.com.au/) | UML, DDS, MDA, UPDM, Corba IDL, SysML, BPEL, ArchiMate, WSDL, DoDAF, BPMN, TOGAF, SoaML, MODAF, SPEM | DB2, Firebird/InterBase, Informix, Ingres, MS Access MS SQL Server, MySQL, Oracle, PostgreSQL, Sybase |

Izvor: Internetske stranice proizvođača/autora CASE tehnologija

2.4 Normizacija softvera i standardi

Korištenje metoda, tehnika i tehnologija treba biti što je više moguće prilagođeno i usklađeno. Da bi se postigla maksimalna povezanost između korištenih metoda, tehnika i tehnologija korištenih u razvoju softvera, potrebna je standardizacija u kanalima komunikacije između njih. Norme i standardi definiraju jasne protokole komunikacije između metoda, tehnika i tehnologija. U nastavku su neke važnije institucije koje se bave propisivanjem i/ili održavanjem standarda u razvoju softvera:

- **SEI** (engl. *software engineering institute*)
- **DoD** (engl. *US department of defense*)
- **ANSI** (engl. *american national standards institute*)
- **IEEE** (engl. *institute of electrical & electronic engineers*)
- **ESA** (engl. *european space agency*)
- **ISO** (engl. *international organization for standardization*)
- **IEC** (engl. *commission electrotechnique internationale - international electrotechnical commission*).

Osnovni izvor normi u razvoju softvera su u početku bile ANSI i IEEE norme koje je nakon određenog vremena preuzeo ISO u suradnji s IEC-om i te se norme nalaze kao rezultat zajedničkog rada združene tehničke komisije - JTC (engl. *joint technical committee*). Za softverski inženjering zadužen je potkomitet 7 (engl. *subcommittee 7, SC7*) koji izrađuje upute za upravljanje i standardizaciju metoda i alata za razvoj, održavanje i ocjenu kvalitete tijekom životnog ciklusa softvera.

2.5 Standardizacija i automatizacija razvoja softvera

Izgradnja softvera završni je korak u proizvodnom ciklusu. Kod iterativnog razvoja takvih ciklusa (iteracija) ima više, a proces izgradnje bitno se ne mijenja, stoga je logično zahtijevati njegovo automatizirano provođenje. Zamisao je jasna i jednostavna: sve postupke u izgradnji (preuzimanje kôda iz repozitorija, prevođenje, izvođenje testnih skripti, prilagodba izvršnih elemenata za različite okoline i platforme, implementacija u radnu-produkcijsku okolinu) treba povezati u jednostavnu, jedinstvenu i sustavno prilagodljivu proceduru. Takva procedura sastoji se od niza aktivnosti povezanih serijski i/ili paralelno. Neke aktivnosti moguće je automatizirati.

Procedura izgradnje softvera uključuje višestruko provođenje pojedinih aktivnosti kao npr.: prevođenje izvornog koda u oblik koji omogućuje testiranje, testiranje programske funkcionalnosti, integracija programskih komponenti s drugim dijelovima softverske infrastrukture (baza podataka, komunikacijski protokoli između komponenta). Potrebno je automatizirati takve postupke kako bi se povećala brzina izgradnje i povećala kvaliteta softvera.

Testiranje programske funkcionalnosti, pokraj osnovnih prevoditeljskih i komponentalnih testova, zahtijeva i korisničke testove na temelju stvarnih podataka. To je uglavnom monoton postupak provođenja istih koraka (i nadopunjavanja novim koracima sukladno sustavnom rastu), verificiranja ispravnosti izvedenih akcija i validiranja nastalih podatkovnih promjena. Takve postupke moguće je u cijelosti ili barem u nekoj mjeri automatizirati.

3. RAZVOJ SOFTVERA NA DOMAĆEM TRŽIŠTU

Razvojne tvrtke koje proizvode softver veći niz godina svakako koriste metode i alate duži period, pa se može pretpostaviti da je korištenje metoda i alata u razvoju softvera u praksi „naslijeđeno“. Ako je proces razvoja softvera ograničen „nasljedstvom“, to može voditi manjoj usklađenosti sa svjetskim trendovima u razvoju softvera.

CASE alati koji se koriste također su u određenoj mjeri naslijeđeni, jer nove tehnologije u razvoju softvera zahtijevaju i nove prilagođene alate. Izučavanje i implementacija novih prilagođenih alata za izgradnju softvera koristeći nove tehnologije (npr. sustav troslojne arhitekture) je složen postupak, i svakako daje razlog višegodišnjem („naslijeđenom“) korištenju razvojnih alata.

Prilagodba poslovanja i implementacija nekog standarda kvalitete (npr. ISO 9000, ISO 27000) zahtjevan je posao i zahtijeva kompetencije iz različitih poslovnih područja (ekonomija, IT-inženjering, komunikacija - zahtjevi kupca). S obzirom na to da većina hrvatskih tvrtki koja se bavi razvojem softvera pripada kategoriji „male tvrtke“, odnosno tvrtke s do 3 zaposlenika, teško je očekivati da će iste biti akreditirano usklađene s nekim normama. To ne znači da iste tvrtke nisu na toj organizacijskoj i poslovnoj razini da mogu izgrađivati kvalitetan softver. Veći broj izvršioaca u složenom procesu razvoja softvera povećava vjerojatnost implementacije standarda kvalitete.

Prema podacima Hrvatske gospodarske komore (<http://www.biznet.hr/>) u vrijeme pisanja ovog rada, broj tvrtki koje su predale godišnje financijsko izvješće (GFI-POD) za 2011. godinu, a koje su registrirale Računalno programiranje (J6201) kao djelatnost prema Nacionalnoj klasifikaciji djelatnosti od 2007. godine (NKD2007) iznosi 1450, a ukupan broj zaposlenih u tim tvrtkama je 7311. Podaci o broju tvrtki grupirani po broju zaposlenih prikazani su u tablici 2.

Tablica 2. Broj tvrtki prema broju zaposlenih u Republici Hrvatskoj za djelatnost J6201 prema NKD2007 za tvrtke koje su predale GFI-POD 2011.

| Broj zaposlenih | Broj tvrtki | Ukupno zaposlenih (Broj zaposlenih x broj tvrtki) |
|-----------------|-------------|------------------------------------------------------|
| 1 | 827 | 827 |
| 2 | 164 | 328 |
| 3 | 99 | 297 |
| 4 | 45 | 180 |
| 5 | 52 | 260 |
| 6 | 31 | 186 |
| 7 | 27 | 189 |
| 8 | 16 | 128 |
| 9 | 22 | 198 |
| 10 | 22 | 220 |
| 11 | 12 | 132 |
| 12 | 12 | 144 |
| 13 | 7 | 91 |
| 14 | 11 | 154 |
| 15 | 14 | 210 |
| 16 | 8 | 128 |
| 17 | 4 | 68 |
| 18 | 5 | 90 |
| 19 | 7 | 133 |
| 20 i više | 65 | 3348 |
| UKUPNO | 1450 | 7311 |

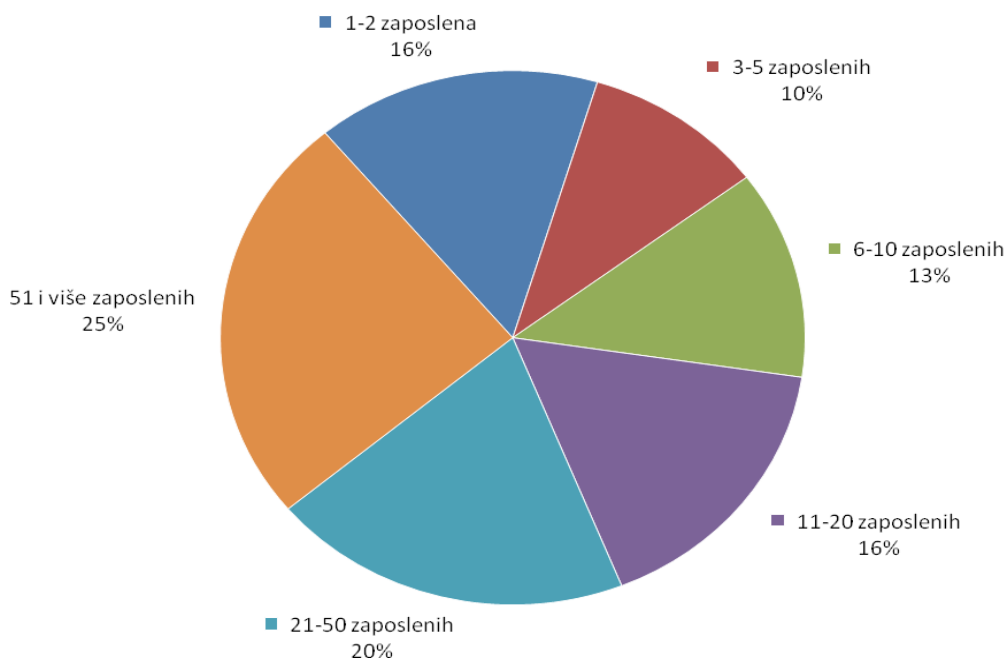
Izvor: obrada autora prema <http://www.biznet.hr/>

Iz tako dobivenog rasporeda tvrtki prema broju zaposlenih, može se vidjeti da od ukupno 1450 tvrtki njih 1090 ima 3 ili manje zaposlenih, što je više od 75 % tvrtki prema navedenim kriterijima. Gotovo 60 % tvrtki ima jednog zaposlenog. Tek 4 % tvrtki ima više od 20 zaposlenih.

Ne znači nužno da tvrtka ne investira u edukaciju i/ili implementaciju standarda ako ima mali broj zaposlenih, ali za pretpostaviti je da će standardi koji osiguravaju kvalitetu razvoja softvera češće biti prisutni kod većih tvrtki.

Veće tvrtke imaju odjele koji se bave isključivo proučavanjem, razvojem i implementacijom novih tehnika razvoja, dok se male tvrtke uglavnom razvijaju inercijom i slučajem. Male tvrtke vrlo malo ulažu (i financijski i vremenski) u promjene i koriste uglavnom iste tehnike duži period.

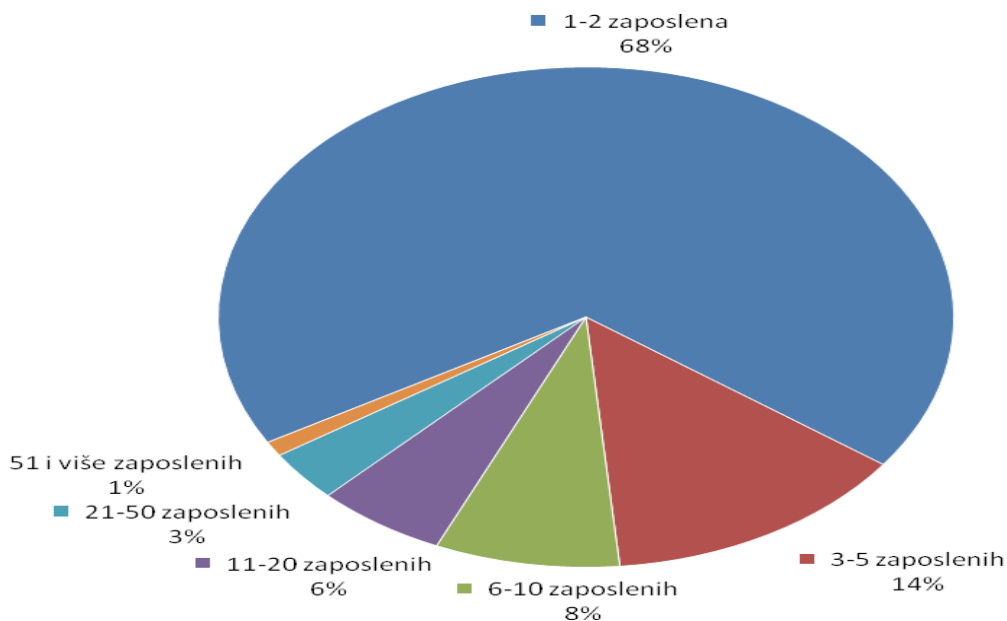
Grafikon 1. Udio broja tvrtki prema broju zaposlenih u ukupnom broju zaposlenih u RH za djelatnost J6201 prema NKD2007 za tvrtke koje su predale GFI-POD 2011.



Izvor: obrada autora prema analizi podataka s <http://www.biznet.hr/>

Rad u maloj tvrtki znači da zaposlenici u pravilu nemaju ili imaju vrlo malo vremena istraživati i pronalaziti bolja razvojna rješenja, vlasnik je ujedno i direktor i projektant i programer i tester i komercijalist i tako dalje. Djelatnici većinu vremena provode u održavanju isporučenog softvera, a razvoj novog softvera uglavnom se radi na bazi postojećeg, korištenjem istih metoda, alata i/ili procedura.

Grafikon 2. Udio broja tvrtki prema broju zaposlenih u ukupnom broju tvrtki u RH za djelatnost J6201 prema NKD2007 za tvrtke koje su predale GFI-POD 2011.



Izvor: obrada autora prema analizi podataka s <http://www.biznet.hr/>

3.1 Anketni upitnik

Prema prethodnom poglavlju populacija koja je obuhvaćena istraživanjem su sve osobe zaposlene u tvrtkama kojima je osnovna djelatnost računalno programiranje. Polazne pretpostavke za odabir internetske ankete kao tehnike prikupljanja podataka o upotrebi metoda i standarda u razvoju softvera u hrvatskoj praksi su (Tkalec Verčić, Sinčić Ćorić, Pološki Vokić, 2010:104):

- osobina populacije koja se istražuje - pismeni ljudi s kojima se može kontaktirati elektroničkom poštom i koji se služe internetom,
- u slučaju internetske ankete vjerojatnost da je „prava“ osoba popunila anketu visoka je,
- veličina anketnog uzorka je velika i zemljopisno raspršena,
- upitnik mora biti kratak,
- pitanja ne smiju biti presložena,
- minimalan trošak slanja i obrade rezultata.

S obzirom na tehniku prikupljanja podataka uzorak za istraživanje su sve osobe zaposlene u tvrtkama kojima je osnovna djelatnost računalno programiranje i koje imaju javno dostupnu adresu elektroničke pošte.

Važno je naglasiti da u anketni uzorak, zbog obimnosti, nisu uzete u obzir organizacije kojima računalno programiranje i proizvodnja softvera nije primarna djelatnost, iako mnoge imaju informatičke odjele.

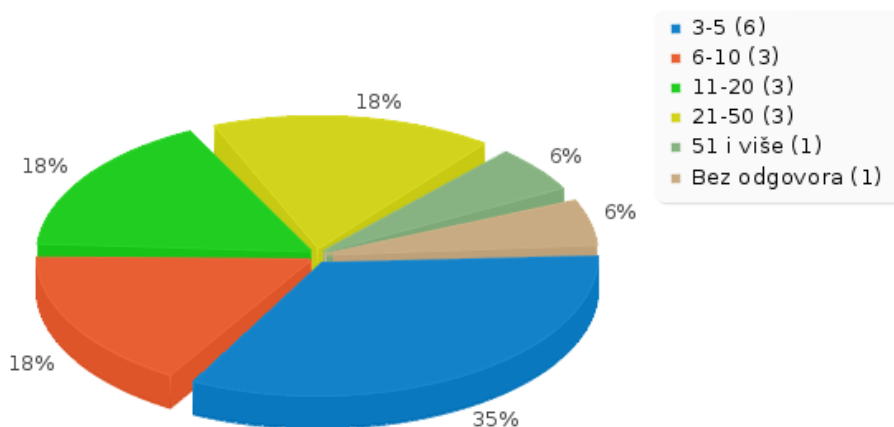
U svrhu istraživanja upotrebe metoda i standarda u razvoju softvera u hrvatskoj praksi napravljen je anketni upitnik i poslan na 122 adrese elektroničke pošte. Adrese su preuzete s internetske stranica Hrvatske gospodarske komore u kolovozu 2012. i iz adresara hrvatske ICT industrije (*Croatian ICT Tender Tuide*) koji je 2011. objavila izdavačka kuća VID I iz Zagreba.

3.2 Rezultati anketnog upitnika

3.2.1 Broj zaposlenih, godine iskustva i pozicija u tvrtki

Udio tvrtki po broju zaposlenih prema odgovorima u anketnim upitnicima gotovo u potpunosti odgovara analizi udjela broja tvrtki prema broju zaposlenih u ukupnom broju zaposlenih u Republici Hrvatskoj za djelatnost J6201 Računalno programiranje prema NKD2007 za tvrtke koje su predale GFI-POD 2011, što je prikazano ranije u grafikonu 1.

Grafikon 3. Udio anketiranih prema broju zaposlenih u tvrtki u kojoj rade



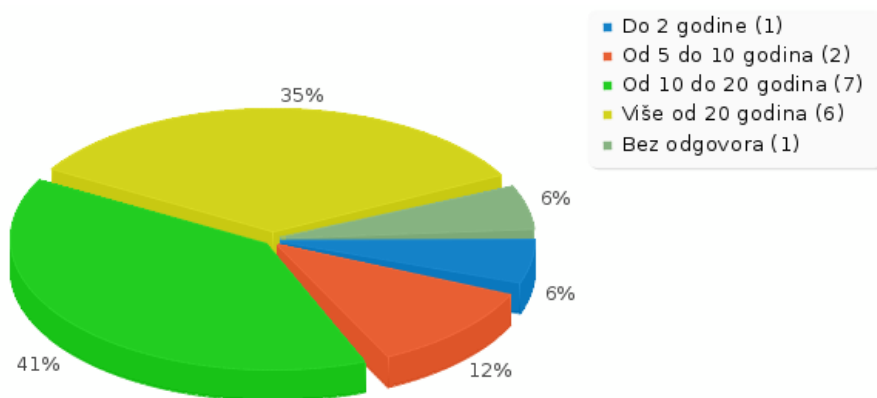
Izvor: obrada autora

U tvrtkama do 5 zaposlenih radi 35 % anketiranih, dok je u ukupnim podacima za Republiku Hrvatsku taj udio 10 %. Udio anketiranih u tvrtkama sa 6 do 10 zaposlenih je 18 % u odnosu na 13 % u ukupnim podacima. U tvrtkama od 11 do 20 zaposlenih radi 18 % anketiranih, dok u ukupnim podacima taj udio iznosi 16 %. U tvrtkama od 21 do 50 zaposlenih radi 18 % anketiranih, a u ukupnim je podacima udio 20 %. U tvrtkama s više od 50 zaposlenih radi 6 % anketiranih, a u ukupnim je podacima taj udio 25 %. Podaci o broju zaposlenih iz rezultata ankete (grafikon 3) u velikom dijelu odgovaraju postocima za ukupne udjele prema broju zaposlenih (grafikon 1), te se procjenjuje da je uzorak u odnosu na populaciju prikladan za donošenje zaključaka.

U ukupnom broju anketiranih do 2 godine iskustva u IT-industriji ima 6 % anketiranih, od 2 do 5 godina iskustva u IT-industriji nema ni jedan anketirani. Od 5 do 10 godina iskustva u IT industriji

ima 12 % anketiranih, od 10 do 20 godina iskustva u IT-industriji ima 41 % anketiranih, a više od 20 godina iskustva ima 35 % anketiranih.

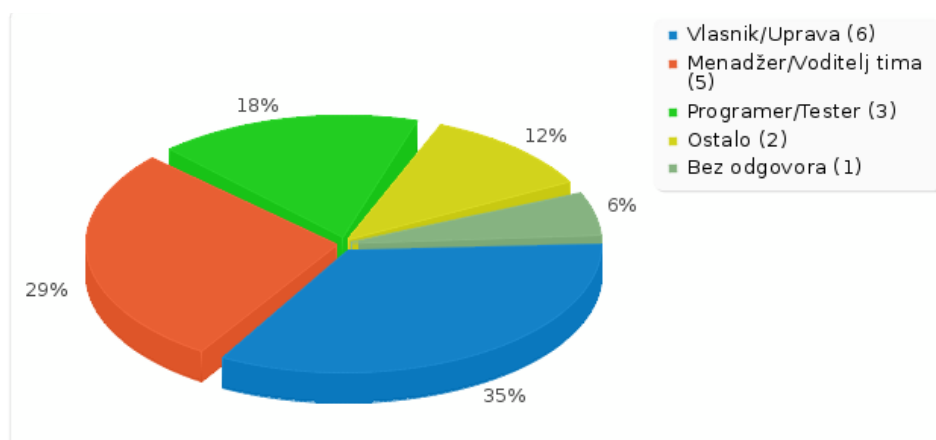
Grafikon 4. Udio anketiranih prema godinama iskustva u IT industriji



Izvor: obrada autora

U ukupnom broju anketiranih 12 % anketiranih radi na funkcijama menadžera ili voditelja tima, 34 % anketiranih rade kao programeri ili testeri, a 54 % anketiranih su vlasnici ili članovi uprave.

Grafikon 5. Udio anketiranih prema poziciji u tvrtki



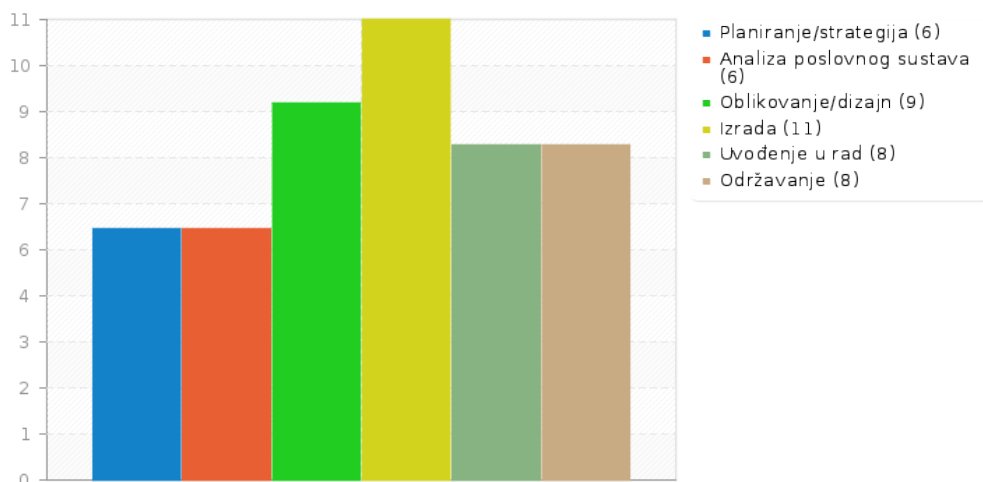
Izvor: obrada autora

S obzirom na to da je na pitanje o godinama iskustva više od 70 % anketiranih odgovorilo da je više od 10 godina u IT-industriji, a na pitanje pozicije u tvrtki više od trećine anketiranih odgovorilo da su vlasnici ili članovi uprave, može se zaključiti da se uzorak bazira na iskusnim osobama s dugim radnim stažem u IT industriji, što je vidljivo na grafikonima 4 i 5.

3.2.2 Pokrivenost životnog ciklusa softvera metodama i alatima

Na pitanje postoje li u tvrtki propisane ili ustaljene metode, alati ili procedure prilikom razvoja programskog rješenja anketirani su mogli odabrati jednu ili više faza razvoja. 64 % je odgovorilo potvrdno za fazu izrade, 52 % je odgovorilo potvrdno za fazu oblikovanja, 47 % je odgovorilo potvrdno za fazu održavanja, 47 % je odgovorilo potvrdno za fazu uvođenja, 35 % odgovorilo je potvrdno za fazu analize i 35 % anketiranih je odgovorilo potvrdno za fazu planiranja. Prema rezultatima anketnog upitnika može se zaključiti da su u hrvatskoj praksi sve faze u životnom ciklusu razvoja softvera pokrivene metodama i alatima, što je vidljivo iz grafikona 6. Pretpostavilo se da se u metode i alate ulaže samo ako je to neophodno, ali i da se bez alata za izradu softvera ne može ni govoriti o razvoju softvera. Za očekivati je bilo da će najpokrivenija faza u životnom ciklusu razvoja biti upravo izrada softvera.

Grafikon 6. Pokrivenost životnog ciklusa softvera metodama i alatima

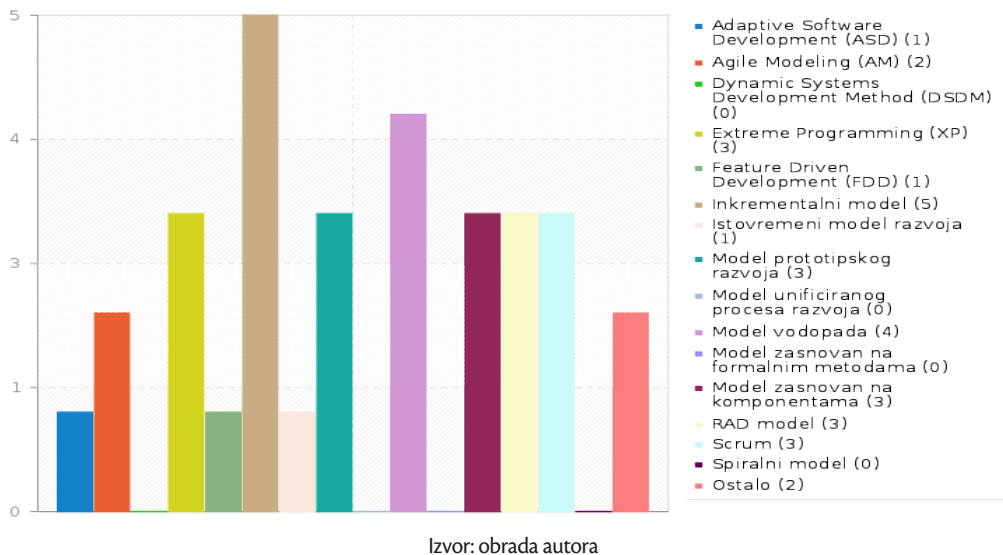


Izvor: obrada autora

3.2.3 Upotreba metoda, alata i procedura u praksi

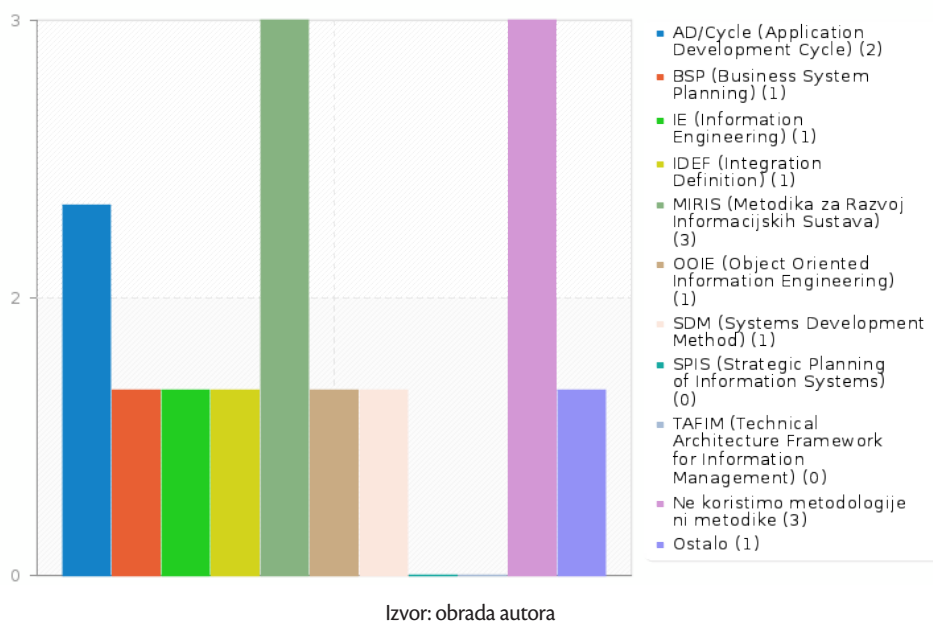
Na pitanje koje metode modeliranja koriste u praksi, odgovor je sljedeći: 5 anketiranih koristi Inkrementalni model, 4 anketirana koriste Model vodopada, 3 anketirana *Extreme Programming* (XP), 3 koriste Model prototipskog razvoja, a 3 anketirana koriste Model zasnovan na komponentama, 3 koriste RAD model, 3 koriste *Scrum*, 2 anketirana koriste *Agile Modeling* (AM), 1 anketirani koristi *Adaptive Software Development* (ASD), 1 koristi *Feature Driven Development* (FDD) i 1 koristi Istovremeni model razvoja.

Grafikon 7. Najčešće korištene metode modeliranja



Ni jedan anketirani ne koristi Model unificiranog procesa razvoja, Model zasnovan na formalnim metodama, *Dynamic Systems Development Method* (DSDM) ili Spiralni model. Prema rezultatima ankete najčešće se za modeliranje koriste metode inkrementalnog modela i modela vodopada (grafikon 7). Koriste se sve navedene metode modeliranja što je vidljivo u grafikonu 7, te se potvrđuje da se u Republici Hrvatskoj koristi mješavina različitih metoda modeliranja.

Grafikon 8. Najčešće korištene metodologije



Na pitanje koje propisane ili ustaljene metodologije i metodike koriste u praksi, troje anketiranih odgovorilo je da koristi MIRIS, dvoje anketiranih odgovorilo je da koristi AD/Cycle, 1 anketirani koristi BSP, 1 anketirani koristi IE, 1 anketirani koristi IDEF, 1 koristi OOIE, 1 koristi SDM. SPIS i TAFIM ne koristi niti jedan anketirani, a troje anketiranih ne koristi metodologije ni metodike.

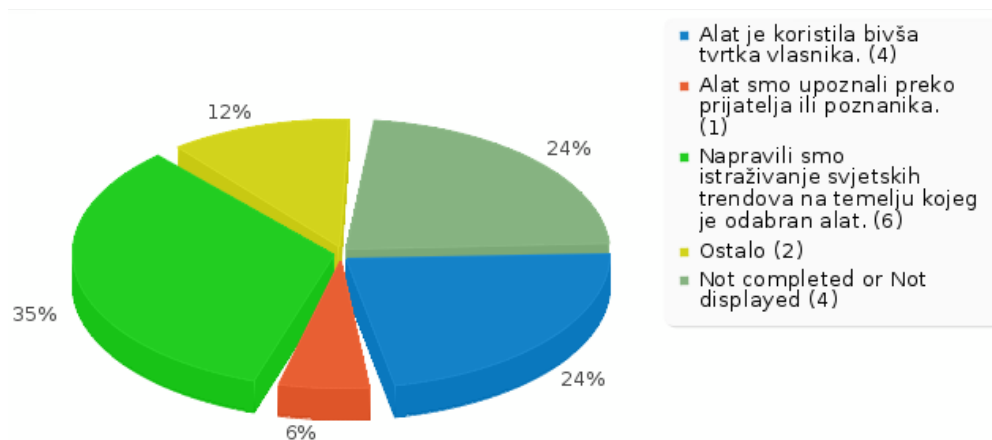
Dakle, najčešće korištena metodologija je MIRIS (metodika za razvoj informacijskih sustava), što je vidljivo u grafikonu 8.

3.2.4 Izvor i odabir CASE alata

Prema izvoru alata 35 % anketiranih je alat koji koristi u razvoju softvera izabralo prema istraživanju svjetskih trendova, 24 % koristi isti alat koji je koristila bivša tvrtka vlasnika, 6 % anketiranih je alat upoznalo preko prijatelja ili poznanika, a ni jedan od ispitanih nije isti alat koristio na fakultetu.

U grafikonu 9 vidljivo je da je prije odabira alata za razvoj softvera većina ispitanih sama napravila istraživanje svjetskih trendova, što se podudara s radnom hipotezom.

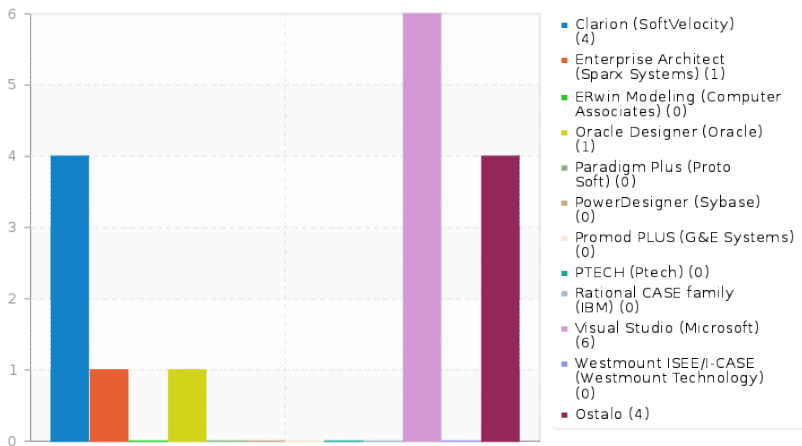
Grafikon 9. Izvor alata za razvoj softvera



Izvor: obrada autora

Na pitanje koje od ponuđenih CASE tehnologija/alata koriste 6 anketiranih odgovorilo je da koristi *Visual Studio*, 4 koriste *Clarion* tvrtke *SoftVelocity*, 1 koristi *Enterprise Architect* tvrtke *Sparx Systems*, 1 koristi *Oracle Designer* tvrtke *Oracle*. *ERwin Modeling*, *Paradigm Plus*, *PowerDesigner*, *Promod PLUS*, *PTECH*, *Rational CASE family* i *Westmount ISEE/I-CASE* ne koristi ni jedan anketirani. Najčešće korišten CASE alat je *Microsoft Visual Studio*, što i nije neobično s obzirom na činjenicu da u svojoj besplatnoj verziji ima gotovo sve funkcionalnosti. Velik broj ispitanika koristi *Clarion*.

Grafikon 10. Najčešće korištene CASE tehnologije/alati



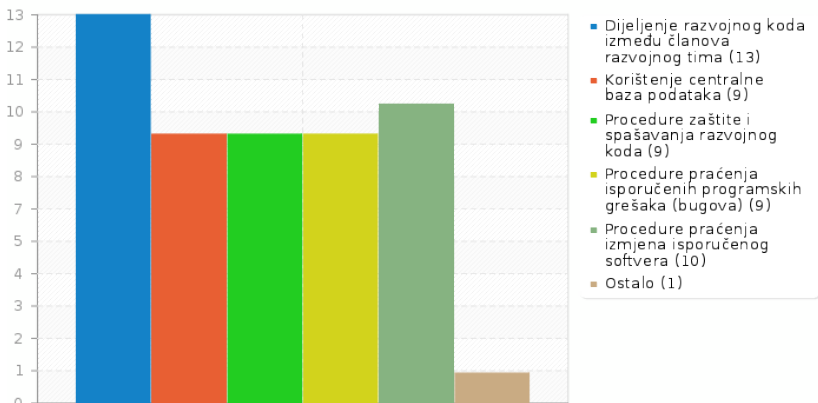
Izvor: obrada autora

3.2.5 Procedure

Svi ispitanici u odgovoru na pitanje postoje li u njihovoj tvrtki ostale propisane ili ustaljene procedure odgovorili su potvrdno za barem jednu proceduru. Kao odgovor su mogli odabrati jednu ili više faza razvoja. 76 % anketiranih koristi dijeljenje razvojnog koda između članova razvojnog tima, 53 % koristi centralnu bazu podataka, 53 % koristi procedure zaštite i spašavanja razvojnog koda, 53 % koristi procedure praćenja isporučenih programskih grešaka, a 59 % koristi procedure praćenja izmjena isporučenog softvera.

Iz grafikona 11 je vidljivo da se najviše pažnje posvećuje procedurama za dijeljenje razvojnog koda među članovima razvojnog tima, i procedurama za praćenje izmjena isporučenog softvera. To pokazuje da se razvoju softvera pristupa timski i može upućivati na potrebu za paralelnim višekorisničkim razvojem softvera.

Grafikon 11. Korištenje propisanih ili ustaljenih procedura u razvoju softvera



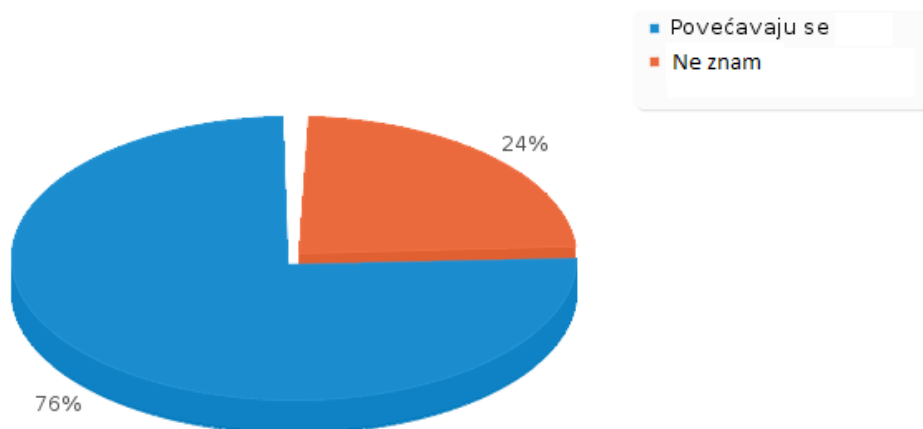
Izvor: obrada autora

3.2.6 Procjena odnosa upotrebe metodologija i razvojnih praksi na uspješnost i kvalitetu isporučenih programskih rješenja

Na pitanje o procjeni odnosa upotrebe metodologija i razvojnih praksi na uspješnost i kvalitetu isporučenih programskih rješenja ni jedan od anketiranih ne misli da nema značajnijeg utjecaja, 24 % anketiranih ne zna, a 76 % anketiranih misli da se uspješnost i kvaliteta isporučenog softvera povećava. Prema tome, može se reći da domaće tvrtke provode razvoj softvera inženjerskim pristupom, i prema aktualnim svjetskim trendovima, što potvrđuje osnovnu hipotezu.

Udjeli pojedinih odgovora mogu se vidjeti u grafikonu 12.

Grafikon 12. Utjecaj upotrebe metodologija i razvojnih praksi u odnosu na uspješnost i kvalitetu isporučenih programskih rješenja



Izvor: obrada autora

4. ZAKLJUČAK

Od pristupa razvoju softvera na jednom računalu, i isporuke na disketi, došlo se do goleme softverske i hardverske infrastrukture za koju treba pronaći opravdanje, što u konačnici nije teško: ozbiljan softver treba i ozbiljan pristup u samom razvoju.

Odabir metode, skupine metoda ili metodologije koje će tvrtka koristiti u razvoju softvera ovisi o mnogo faktora – operacijska platforma za koju se softver radi, vrsta i namjena softvera, softver koji se trenutno razvija u tvrtki i koji je već u primjeni, infrastruktura krajnjeg korisnika softvera.

U svijetu softvera i softverskog inženjerstva danas postoji gotovo 1500 standarda. Da bi se oblast softverskog inženjeringa podržala neophodnim standardima, uložen je ogroman trud i rad na međunarodnom, regionalnom i nacionalnom nivou.

Ne treba očekivati da su svi međunarodni standardi savršeno sustavni, usklađeni po metodologiji i sadržaju, ali su izvan svake sumnje snažno sredstvo, uz čiju pomoć se može vršiti unapređenje projektantskih metoda i osigurati kvalitetu gotovih softverskih proizvoda.

Trenutna situacija u Republici Hrvatskoj je takva da se metode, alati i standardi razvoja softvera koji postoje na tržištu u praksi koriste prema potrebi. U Republici Hrvatskoj je odabir metoda, alata i procedura za razvoj softvera baziran na istraživanju svjetskih trendova. U manjem dijelu se koriste metode i alati koji su se u radnoj okolini ustalili zato što se koriste već godinama, a koristi se mješavina različitih metoda, alata i postupaka. Koriste se one metode, alati i tehnike koje su najprimjerenije vrsti softvera koji se proizvodi, što je dobro, ali u malo mjeri, što nije dobro.

Proizvođači softvera u Hrvatskoj, bez obzira na pretpostavljeni manjak sredstava kojima raspolažu, ipak slijede svjetske trendove u korištenju metoda, alata i standarda za razvoj softvera.

LITERATURA

- Bachman, C. (1988) *A CASE for Reverse Engineering*, IEEE Datamation
- Baik, J. (2000) *The Effects of Case Tools on Software Development Efort*, Dissertation, University of Southern California
- Bolton, R. et al. (2006) *System development methodology*, New Hampshire Office of Information Technology – Agency Software Division
- Lederer, A. L., Sethi, V. (1988) *The Implementation of Strategic Information System Planning Methodologies*, MIS Quarterly
- McLaughlin, M. (1996) *Department of Defense Technical Architecture Framework for Information Management*, Volume 3: *Architecture Concepts and Design Guidance*, Defense Information Systems Agency, Center for Standards
- Mercurio, V. J., Meyers, B. F., Nisbet, A. M., Radin, G. (2006) *AD/Cycle strategy and architecture*, IBM Systems Journal
- Montgomery, S. (1994) *Object-oriented information engineering: analysis, design, and implementation*, Academic Press Inc., London
- Pavlić, M. (1996) *Razvoj informacijskih sustava*, Zagreb: Znak
- Sommerville, I. (1992) *Software Engineering*, Wokingham: Addison Wesley
- Tkalec Verčić, A., Sinčić Čorić, D., Pološki Vokić, N., (2010) *Priručnik za metodologiju istraživačkog rada: kako osmisliti, provesti i opisati znanstveno i stručno istraživanje*, Zagreb: M.E.P
- <http://www.martinfowler.com/articles/newMethodology.html> (11. 7. 2012.)
- <http://www.gesys.com/pro-mod.htm> (10. 8. 2012.)
- <http://www.orafaq.com/wiki/Designer> (10. 8. 2012.)
- <http://www.unl.csi.cuny.edu/faqs/software-engineering/blurb/Westmount.html> (10. 8. 2012.)
- <http://en.wikipedia.org/wiki/Ptech> (10. 8. 2012.)
- <http://www.encyclo.co.uk/define/Paradigm%20Plus> (10. 8. 2012.)
- <http://www-01.ibm.com/software/rational/> (10. 8. 2012.)
- <http://erwin.com/> (10. 8. 2012.)
- <http://www.sybase.com/products/modelingdevelopment/powerdesigner> (10. 8. 2012.)
- <http://www.sparxsystems.com.au/> (10. 8. 2012.)
- <http://www.biznet.hr/> (2. 6. 2012.)

Dominik Čubranić¹
Marin Kaluža²
Jadranko Novak³

Professional paper
UDC 65.012.12:004.413>(479.5)

STANDARD METHODS USED IN SOFTWARE DEVELOPMENT IN THE REPUBLIC OF CROATIA⁴

ABSTRACT

Software development is a complex process which consists of many steps, starting from the initial user requirements, through analysis and development to the implementation and distribution. Software development involves the use of a number of different technologies and techniques. There are laws, rules and order in the use of technology and techniques, so if software development approach is ad-hoc it is not possible to develop good, useful and reliable software. It is necessary to use methodology in software development in order to ensure the reliability and quality of software. Methodologies determine the time and place of the use of methods. This article will show some of those methods used for software development. The use of CASE technologies is necessary to ensure speed, accuracy, efficiency of software development and the ease of maintenance. This article will analyze features of some CASE technologies. CASE technologies also provide standardization in software development and the application of standards. The article will give a review of some of the standards and norms that apply to software development. This article will analyze the data using the Croatian Chamber of Economy Internet site as a source for the number of companies and the number of employees in the Republic of Croatia engaged in computer programming. The same source will be used to get e-mail addresses for a questionnaire on the use of methods and standards in software development. The goal of the research is to show the quantity and the quality (descriptions, coverage of stages in software development) of the most frequently used techniques, methods and standards.

Key words: CASE tools, software development, software development in Croatia, software engineering

¹ Professional Specialist of Information Technology, Designer and Programmer, Mapro d. o. o., Bože Vidasa 20, Rijeka, Croatia. E-mail: dominik@mapro.hr

² MSc, Senior lecturer, Polytechnic of Rijeka, Vukovarska 58, Rijeka, Croatia. E-mail: mkaluza@veleri.hr

³ MSc, Managing Director, Caminus d. o. o., Krešimirova 20, Rijeka, Croatia. E-mail: jadranko.novak@caminus.hr

⁴ Received: 31. 1. 2013; accepted: 3. 4. 2013