STANDARDNI MODELI ODRŽAVANJA SOFTVERA

STANDARD MODELS OF SOFTWARE MAINTENANCE

Samir Lemeš, v.prof.dr.
Univerzitet u Zenici, Politehnički fakultet
Zenica, Bosna i Hercegovina

Nevzudin Buzađija, doc.dr.
Mješovita srednja tehnička škola
Travnik, Bosna i Hercegovina

REZIME
Održavanje softvera je integralni dio životnog ciklusa softvera. Ono obuhvata sve aktivnosti koje se poduzimaju da bi se obezbijedila efikasna i ekonomski prihvatljiva podrška korisnicima softvera. Područje održavanja softvera je pokrivenom nizom standarda, kao što su IEEE 1219, IEEE 1061, ISO/IEC 14764, MIL-HDBK-347. U ovom radu dat je pregled međunarodnih standarda koji se bave održavanjem softvera, te njihova veza s priručnikom softverskog inženjerstva SWEBOK v3.0.

Ključne riječi: Softver, Održavanje, Standardi, SWEBOK

ABSTRACT
Software maintenance is an integral part of the software life cycle. It includes all the activities undertaken to provide efficient and economically acceptable support to software users. The software maintenance is covered by a number of standards, such as IEEE 1219, IEEE 1061, ISO/IEC 14764, MIL-HDBK-347. This paper provides an overview of international software maintenance standards and their relationship to the Guide to the Software Engineering Body of Knowledge SWEBOK v3.0.

Key words: Software, Maintenance, Standards, SWEBOK

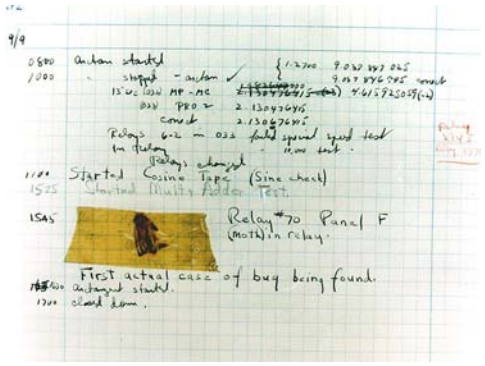
1. UVOD

Osnovni cilj svakog softverskog proizvoda je da zadovolji potrebe korisnika. Da bi se taj cilj ispunio, neophodno je da kroz svoj životni ciklus pretrpi modifikacije. Moglo bi se zato reći da je održavanje softvera u stvari upravljanje procesom modifikacija.
Jedna od najcitiranijih studija o održavanju softvera, koju su objavili 1980. godine istraživači sa *UCLA*, *Lientz* i *Swanson* [1] ponudila je klasifikaciju održavanja softvera na 4 kategorije, koje su kasnije prihvaćene i standardizirane u ISO/IEC 14764:2006 [2]:

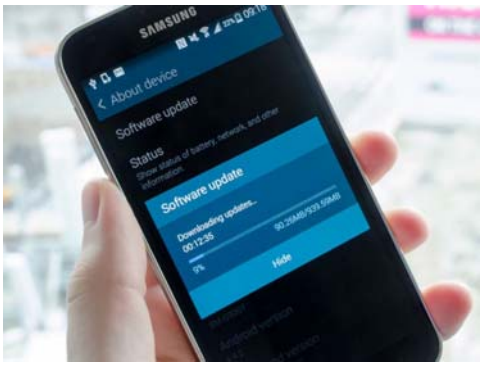
- Korektivno održavanje: Reaktivne modifikacije softvera koje se vrše nakon isporuke, s ciljem popravljanja uočenih problema,
- Adaptivno održavanje: Modifikacije softvera koje se vrše nakon isporuke da bi softverski proizvod ostao upotrebljiv u okruženju koje se mijenja,
- Perfektivno održavanje: Modifikacije softvera koje se vrše nakon isporuke, da bi se popravile performanse ili pogodnost za održavanje,
- Preventivno održavanje: Modifikacije softvera koje se vrše nakon isporuke s ciljem otkrivanja latentnih grešaka prije nego što te greške dovedu do otkaza ili zastoja.

Niz istraživanja provedenih na tu temu pokazala su da preventivno održavanje čini manje od 5% svih aktivnosti održavanja, korektivno i adaptivno čine po oko 20%, dok perfektivno

održavanje obuhvata više od 50% svih aktivnosti održavanja [2]. Primjer korektivnog održavanja je uklanjanje grešaka u programskom kodu, poznato kao *debugging*. Termin "bug" (buba) se u tehnici koristio od druge polovine XIX vijeka, ali se u računarstvu prvi put koristio na Univerzitetu *Harvard* 1946. godine, kad je jedan od pionira računarstva *Grace Hopper* dokumentovao kvar u računaru *Mark II* koju je izazvao moljac tako što je napravio kratki spoj u releju računara (slika 1). Moderno održavanje softvera je postalo skoro u potpunosti automatizirano i često se dešava nezavisno od volje korisnika. Na primjer, ažuriranje sistemskog ili aplikativnog softvera na mobilnim uređajima se vrši automatski, a može se čak i prilagoditi načinu spajanja na mrežu, tako da se instalacija nove verzije vrši samo kad je dostupan *Wi-Fi* signal radi uštede troškova konekcije (Slika 2). Povećanje brzine prenosa podataka i dostupnosti interneta unaprijedilo je proces održavanja prikupljanjem ogromne količine povratnih podataka od korisnika, na osnovu čega proizvođači softvera vrše modifikacije, odnosno održavaju softverski proizvod.



Slika 1. Izvještaj o greški u računaru Mark II koju je izazvao moljac – prva upotreba termina "bug" za greške u radu računara [3]



Slika 2. Ažuriranje (update) modernog softvera kao kombinacija perfektivnog i preventivnog održavanja je više od otklanjanja bugova

Održavanje softvera je tema o kojoj su pisali mnogi autori, a često se referiraju na tzv. *Lehmanovim* zakonima evolucije softvera, koje je *Lehman* prvi put objavio u [4], a koji su nakon toga više puta modificirani. Taj set pravila uspostavljen je kako bi se proizvodom tako podložnom promjenama kao što je softver, lakše upravljalo tokom životnog ciklusa. Tih osam *Lehmanovih* zakona [5] glase:

- "Kontinuirane promjene" - sistem se mora kontinuirano adaptirati ili progresivno postaje manje zadovoljavajući (1974)
- "Narastajuća kompleksnost" - kako sistem evoluira, njegova kompleksnost raste osim ako se ne radi ništa na održavanju ili smanjenju kompleksnosti (1974)
- "Samoregulacija" - procesi evolucije sistema su samoregulirajući sa distribucijom proizvoda i mjera procesa bliskih normalnom (1974)
- "Očuvanje stabilnosti organizacije (invarijantni nivo rada)" - prosječni efektivni globalni nivo aktivnosti u evoluirajućem sistemu je invarijantan po životnom ciklusu proizvoda (1978)
- "Očuvanje familijarnosti" - kako sistem evoluira, svi povezani s njim, developeri, prodavači i korisnici, moraju ostati odlično upoznati sa sadržajem i ponašanjem softvera kako bi se postigla zadovoljavajuća evolucija. Pretjeran rast umanjuje tu informisanost. Stoga, prosječni inkrementalni rast ostaje invarijantan kako sistem evaluira. (1978)

- "Kontinuirani rast " - funkcionalni sadržaj sistema mora se kontinuirano povećavati da bi se održalo zadovoljstvo korisnika tokom životnog ciklusa (1991)
- "Opadajući kvalitet" - izgledat će da kvalitet sistema opada ako se ne održava rigorozno i adaptira promjenama operativnog okruženja (1996)
- "Sistem povratne veze" - Evolutivni procesi sastoje se od sistema povratne veze sa više slojeva, više petlji, više agenata, i moraju se tretirati na način da se postiže značajno unapređenje nad svakom razumnom bazom (1996)

Ovi zakoni propisuju da je potreba za funkcionalnim promjenama u softverskom sistemu neizbježna, a ne da je posljedica nepotpune ili netačne analize zahtjeva ili lošeg programiranja. Oni navode da postoje granice koje razvojni tim softvera može postići u smislu bezbjedne implementacije promjena i novih funkcionalnosti.

Grupa autora je u [6] analizirala faktore kvaliteta pogodnosti softvera za održavanje i utjecaj njihovih karakteristika na proces održavanja. Oni su dali i kritički osvrt na upotrebu cijelog niza standarda koji su vezani za održavanje softvera, ali i standarda koji nisu isključivo vezani za održavanje, nego se bave kvalitetom i/ili životnim ciklusom softvera, kao što je serija standarda ISO/IEC 25000.

Koponen i *Hotti* su analizirali aspekte održavanja softvera otvorenog koda (*Open Source*) [7] *Apache* i *Mozilla*. Pored ISO/IEC modela procesa održavanja koji je opisan u SWEBOK [8] i standardima ISO/IEC 12207 [9] i ISO/IEC 14764 [2], analizirali su i alternativne opise procesa održavanja iz standarda ISO/IEC 15288 [10] i britanskog *de facto* standarda ITIL (*IT Infrastructure Library*), koji održavanju prilaze sa aspekta životnog ciklusa sistema, umjesto životnog ciklusa softvera.

Kajko-Mattsson je u [11] poredila model zrelosti korektivnog održavanja i dijelove modela zrelosti evolucije i održavanja sa odredbama standarda IEEE 1219 [12] i zaključila da se taj standard neophodno unaprijediti kako bi pravilno reflektirao domen korektivnog održavanja u industrijskim okvirima.

Stojanov i dr. su u [13] predstavili studiju slučaja provedenu u mikro softverskoj kompaniji usmjerenu ka uvođenju šeme za klasifikaciju zadataka održavanja, te identifikaciji trendova u distribuciji zadataka održavanja softvera među programerima u kompaniji. Autori su dali prijedlog vlastite klasifikacije zadataka održavanja, koja se razlikuje od onih prikazanih u standardima [2, 9, 10].

Alrawashdeh i dr. su u [14] analizirali više aspekata softvera za posebne namjene (*Enterprise Resource Planning - ERP*) prema odredbama standarda ISO 9126 [15], između ostalog i pogodnost za održavanje. Poredili su modele za ocjenu kvaliteta softvera iz standarda sa 4 alternativna modela, a rezultat istraživanja je identifikacija komponenti pogodnosti za održavanje ERP softvera: pogodnost za analizu (da li dijagnoza grešaka ili identifikacija promjena zahtijeva minimalne napore?), pogodnost za promjene (može li se ERP sistem lako mijenjati?), stabilnost (može li ERP sistem funkcionirati nakon promjena?) i pogodnost za testiranje (može li se modifikovani ERP sistem lako validirati?).

Kumar i *Parul* su u [16] opisali nekoliko praktičnih problema s kojima je suočen tim za razvoj softvera i s tim vezane aspekte procesa održavanja softvera: reinženjering, mjerenja, veličinu baze podataka, analiza rada, broj osoblja, kvalitet proizvoda, restruktuiranje koda i podataka, budžet za održavanje, te iskustvo osoblja za održavanje.

Al-Sarayreh i dr. su pokušali organizovati nekoliko koncepata održavanja u referentne modele zasnovane na standardima [17].

Muhonen i dr. su u [18] analizirali standardizaciju u IoT (*Internet of Things*) i industrijskom internetu, i osvrnuli su se na veliki broj standarda, te organizacija koje razvijaju standarde, posebno za održavanje prema stanju ne samo informatičke opreme, nego i industrijske opreme koja je sve više digitalizirana i povezana, čime se povećava potreba za održavanjem hardvera ali još važnije i za održavanjem softvera.

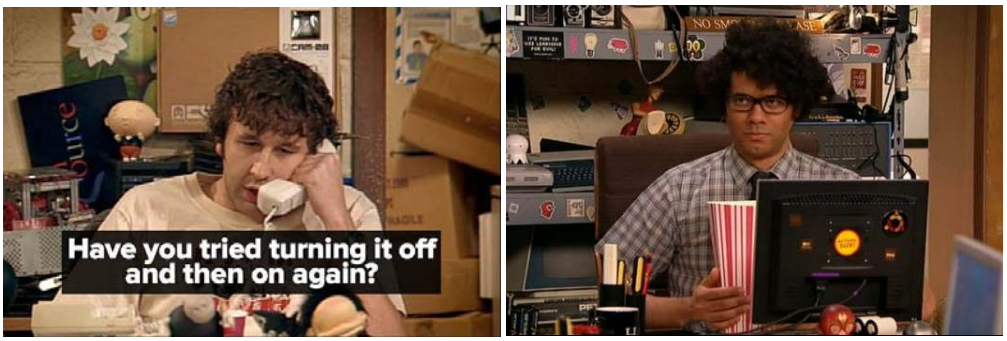
2. ODRŽAVANJE SOFTVERA U SWEBOK v.3

Pojam "softver" je uveo statističar *John Tukey* 1958. godine [8]. Pojam "softversko inženjerstvo" prvi put se javlja na NATO konferenciji održanoj u Njemačkoj 1968. godine. Međunarodno strukovno udruženje *IEEE Computer Society* objavilo je 1972. godine dokument "Transakcije o softverskom inženjerstvu", a 1976. godine osnovan je i komitet za razvoj standarda softverskog inženjerstva u okviru *IEEE Computer Society*. Prvi međunarodni standard o softverskom inženjerstvu usvojen je 1995. godine (ISO/IEC 12207), a posljednje izmjene tog standarda usvojene su 2008. godine [9].

Softversko inženjerstvo obuhvata dizajn, razvoj, upravljanje i dokumentiranje softvera, uz primjenu računarskih nauka, tehnika upravljanja projektima, inženjerstva, dizajna i drugih disciplina. Ono predstavlja inženjersku disciplinu koja se bavi svim aspektima proizvodnje softvera. Na osnovu standarda ISO/IEC 12207, *IEEE Computer Society* kreiralo je dokument pod nazivom Vodič kroz osnove znanja softverskog inženjerstva (*SWEBOK: The Software Engineering Body of Knowledge*), koji je do danas tri puta revidiran, a aktuelna je treća verzija tog dokumenta [8]. Softversko inženjerstvo po SWEBOK-u obuhvata sljedeće oblasti:

1. Softverski zahtjevi (*Software Requirements*)
2. Dizajn softvera (*Software Design*)
3. Konstruiranje softvera (*Software Construction*)
4. Testiranje softvera (*Software Testing*)
5. Održavanje softvera (*Software Maintenance*)
6. Upravljanje softverskim konfiguracijama (*Software Configuration Management*)
7. Upravljanje softverskim inženjerstvom (*Software Engineering Management*)
8. Procesi softverskog inženjerstva (*Software Engineering Process*)
9. Alati i metode softverskog inženjerstva (*Software Engineering Tools and Methods*)
10. Kvalitet softvera (*Software Quality*)
11. Stručna praksa softverskog inženjerstva (*Software Engineering Professional Practice*)
12. Ekonomika softverskog inženjerstva (*Software Engineering Economics*)
13. Osnove računarstva (*Computing Foundations*)
14. Matematičke osnove (*Mathematical Foundations*)
15. Inženjerske osnove (*Engineering Foundations*)

Peto poglavlje SWEBOK-a obrađuje oblast održavanja softvera, kao integralnog dijela životnog ciklusa softvera. Održavanje se definira kao skup svih aktivnosti neophodnih da se obezbijedi efikasna podrška korisnicima softvera [8], koja se dijeli na podršku prije i nakon isporuke softvera, a podrška nakon isporuke obuhvata izmjene softvera, obuku i službu za podršku (slika 3).



Slika 3. U svakodnevnoj praksi pod održavanjem softvera podrazumijeva se služba za podršku. Jedna od najpoznatijih humorističkih serija na temu IT podrške je britanska serija "The IT Crowd" (2006-2013), koja banalizuje održavanje softvera na frazu "isključiti pa uključiti"

Svrha održavanja softvera opisana je u standardu ISO/IEC 14764 [2], a slika 4 pokazuje teme koje obuhvata održavanje softvera prema tom standardu.

Održavanje softvera				
Osnove održavanja softvera	**Ključni problemi održavanja softvera**	**Proces održavanja**	**Tehnike održavanja**	**Alati za održavanje softvera**
Definicije i pojmovi	Tehnička pitanja	Procesi održavanja	Razumijevanje programa	
Priroda održavanja	Pitanja menadžmenta	Aktivnosti održavanja	Reinženjering	
Potrebe za održavanjem	Ocjena troškova održavanja		Reverzni inženjering	
Većina troškova održavanja	Mjerenje održavanja softvera		Migracija	
Evolucija softvera			Povlačenje	
Kategorije održavanja				

Slika 4. Pregled tema koje obuhvata održavanje softvera prema standardu ISO/IEC/IEEE 14764 [2]

U prvoj skupini tema, opisane su četiri glavne vrste održavanja, koje se mogu podijeliti na proaktivne (preventivno i perfektivno) i reaktivne (korektivno i adaptivno održavanje).
U drugoj skupini tema opisan je primjer važnosti pisanja preglednog koda i dokumentiranja za slučajeve gdje programeri koji razvijaju kod i oni koji kasnije vrše održavanje nisu iste osobe. Na pogodnost za održavanje jako utječe preglednost koda, posebno ako se ima u vidu velika fluktuacija IT kadrova, koji su jedna od najmobilnijih profesija koja često mijenja radno mjesto, kako unutar iste, tako između različitih firmi. Pored esencijalnih vještina kodiranja, poznavanja sintakse programskih jezika, matematičke logike, veoma važan je i stil kodiranja (slika 5). U oba slučaja, kod će biti isto kompajliran, ali održavanje i naknadne izmjene koda mogu se značajno olakšati korištenjem standarda kodiranja. Dobra literatura za standarde kodiranja, dostupna *online*, je knjiga [19], a pokriva veliki broj konvencija i pravila koji olakšavaju čitljivost koda i čine ga lakšim za održavanje. Ta pravila odnose se na imena varijabli, uvlačenje struktura (slika 5), način pisanja komentara i sl.

```
#include <iostream>
using namespace std;

int main()
{
    int n, num, digit, rev = 0;

    cout << "Enter a positive number: ";
    cin >> num;

    n = num;

    do
    {
        digit = num % 10;
        rev = (rev * 10) + digit;
        num = num / 10;
    } while (num != 0);

    cout << " The reverse of the number is: " << rev << endl;

    if (n == rev)
        cout << " The number is a palindrome";
    else
        cout << " The number is not a palindrome";

    return 0;
}
```

```
#include <iostream> using
namespace std; int main()
{int n, num, digit, rev =
0; cout << "Enter a
positive number: "; cin >>
num; n = num; do { digit =
num % 10; rev = (rev * 10)
+ digit; num = num / 10;}
while (num != 0); cout <<
" The reverse of the
number is: " << rev <<
endl; if (n == rev) cout
<< " The number is a
palindrome"; else cout <<
" The number is not a
palindrome"; return 0;}
```

Slika 5. Stil kodiranja nema utjecaja na brzinu izvođenja programa, ali značajno može olakšati naknadne izmjene i održavanje

Treća grupa tema opisuje aktivnosti održavanja: implementaciju procesa, analizu problema i modifikacija, implementaciju modifikacija, pregled/prihvaćanje promjena, migraciju i povlačenje softvera. Tehnike agilnog razvoja softvera također se prilagođavaju potrebama održavanja, tako da se proces održavanja podupire specijaliziranim modelima zrelosti sposobnosti održavanja softvera (*Software Maintenance Capability Maturity Models*) [8]. Postoji niz procesa, aktivnosti i praksi koje su jedinstvene za održavanje softvera: razumijevanje programa, tranzicija, prihvaćanje/odbijanje zahtjeva za izmjenu, podrška za održavanje, analiza utjecaja, te sporazumi o nivou usluga održavanja (*Service-Level Agreements*) [8].
Četvrta i peta grupa tema pokrivaju tehnike održavanja tokom cijelog životnog ciklusa softvera, od razumijevanja softvera do napuštanja softvera, te alate za održavanje softvera koji obuhvaćaju statičke i dinamičke analizatore, analizatore toka podataka i analizatore ovisnosti pojedinih komponenti programa.

3. STANDARDI ZA ODRŽAVANJE SOFTVERA

Pored dokumenta SWEBOK, održavanje softvera je prilično standardizirano, i razvijen je čitav niz međunarodnih standarda koji pokrivaju tu oblast. Čak se i SWEBOK u području održavanja softvera najviše poziva na standard ISO/IEC 14764:2006. Važno je napomenuti da se međunarodni standardi u BiH primjenjuju na načelu dobrovoljnosti, a većinom se preuzimaju od međunarodnih ili regionalnih organizacija za standardizaciju ISO, IEC i EN.
Standard je dokument za opću i višekratnu upotrebu, donesen konsenzusom i odobren od priznatog tijela, koji sadrži pravila, smjernice ili karakteristike aktivnosti ili njihove rezultate i koji ima za cilj postizanje optimalnog stepena uređenosti u datom kontekstu [20]. Standardi se moraju temeljiti na provjerenim naučnim, tehnološkim i iskustvenim rezultatima, a cilj im je dostizanje optimalnog napretka zajednice. Standardi mogu biti industrijski, nacionalni, regionalni ili međunarodni, a nacionalno tijelo za standardizaciju u Bosni i Hercegovini je Institut za standardizaciju BiH. Institut za standardizaciju BiH je samostalna državna upravna organizacija za poslove u području standardizacije. Institut predlaže strategiju standardizacije u BiH, priprema i objavljuje bosanskohercegovačke standarde, zastupa i predstavlja Bosnu i Hercegovinu u međunarodnim, evropskim i drugim organizacijama za standardizaciju, te obavlja poslove koji proizlaze iz međunarodnih sporazuma i članstva u tim organizacijama. Sudjeluje u pripremanju tehničkih propisa, razvija i uspostavlja informacijski sistem o standardima BiH, organizira i provodi specijalističko obrazovanje kadrova u području standardizacije. Bavi se i izdavačkom djelatnošću iz područja standardizacije [20].
U tabeli 1 dat je pregled standarda koji su vezani za održavanje softvera, i to međunarodni standardi organizacija ISO, IEC i IEEE, te korespondentni standardi usvojeni u Bosanskohercegovačkom institutu za standarde kao BAS nacionalni standardi. Iz tabele se može vidjeti da su svi ISO/IEC standardi preuzeti u BAS, osim onih koji su već povučeni, jer ih je ISO zamijenio novim izdanjima standarda ili drugim standardima. Nijedan IEEE standard još nije preuzet u BiH kao nacionalni standard.
Standardi se u BiH usvajaju metodama proglašavanja, korica i prijevoda. Najčešće se koristi najjednostavnija - metoda proglašavanja, koja se primjenjuje ako postoji potreba za primjenom određenog standarda a nema mogućnosti ni potrebe da se u kratkom roku preuzme metodama prijevoda ili korica. Standarde iz oblasti informacionih tehnologija, u koje spadaju i standardi za održavanje softvera, vrši tehnički komitet BAS/TC1.
U Tabeli 1 dat je pregled standarda koji se koriste za održavanje softvera, sa odgovarajućim bosanskohercegovačkim standardima, ukoliko takvi postoje. Kako su standardi dinamička kategorija i podložni su promjenama, ima slučajeva da se neki standard zamijeni novom verzijom, ili se povuče ako je oblast primjene pokrivena nekim novim standardom.

Tabela 1. Pregled standarda vezanih za održavanje softvera [20]

Međunarodni standard	**BAS standard**
ISO/IEC 14764:2006 Software Engineering - Software Life Cycle Processes - Maintenance	BAS ISO/IEC 14764:2008 Softverski inženjering - Procesi životnog ciklusa softvera - Održavanje
ISO/IEC 12207:2008 Systems and Software Engineering - Software Life Cycle Processes	BAS ISO/IEC 12207:2008 Sistemi i softverski inženjering- Procesi životnog ciklusa softvera
ISO/IEC 15288:2015 Systems Engineering - System Life Cycle Processes	BAS ISO/IEC/IEEE 15288:2016 Sistemi i softverski inženjering - Procesi životnog ciklusa sistema
ISO/IEC TR 15271:1998 Information technology - Guide for ISO/IEC 12207 (Software Life Cycle Processes)	*(povučen 2013. godine i zamijenjen standardom BAS ISO/IEC TR 24748-3:2013)*
ISO/IEC TR 24748-1:2016 Systems and software engineering - Life cycle management - Part 1: Guidelines for life cycle management	BAS ISO/IEC TR 24748-1:2017 Sistemski i softverski inženjering - Upravljanje životnim ciklusom – Dio 1: Vodič za upravljanje životnim ciklusom
ISO/IEC TR 24748-2:2011 Systems and software engineering -- Life cycle management - Part 2: Guide to the application of ISO/IEC 15288 (System life cycle processes)	BAS ISO/IEC TR 24748-2:2013 Sistemi i softverski inženjering - Upravljanje životnim ciklusom – Dio 2: Vodič za primjenu standarda ISO/IEC 15288 (Procesi životnog ciklusa sistema)
ISO/IEC TR 24748-3:2011 Systems and software engineering -- Life cycle management - Part 3: Guide to the application of ISO/IEC 12207 (Software life cycle processes)	BAS ISO/IEC TR 24748-3:2013 Sistemi i softverski inženjering - Upravljanje životnim ciklusom – Dio 3: Vodič za primjenu standarda ISO/IEC 12207 (Procesi životnog ciklusa softvera)
IEEE 1219:1998 Standard for Software Maintenance	-
ISO/IEC 9126:2000 Software engineering - Product quality	*(povučen 2012. godine i zamijenjen standardom BAS ISO/IEC 25010:2012)*
ISO/IEC 25010:2011 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models	BAS ISO/IEC 25010:2012 Sistemski i softverski inženjering - Zahtjevi i procjena kvaliteta sistema i softvera (SQuaRE) - Modeli kvaliteta sistema i softvera
MIL-HDBK-347:1990 Mission-Critical Computer Resources Software Support	-
IEEE 1012:2004 Standard for Software Verification and Validation	-
IEEE 1061:1998 (R2009) Standard for a Software Quality Metrics Methodology	-

4. ZAKLJUČAK

Područje održavanja softvera je u velikoj mjeri pokriveno čitavim nizom standarda, od kojih je većina preuzeta kao nacionalni BAS standard u Bosni i Hercegovini. Standard je najveći stepen uređenosti neke oblasti, koji opisuje šta treba raditi, za razliku od "najbolje prakse" koja opisuje kako nešto treba raditi, ili "procedure" u sistemu kvaliteta koja opisuje kako se nešto radi u konkretnoj organizaciji.

Cilj standardizacije u održavanju softvera je definiranje zajedničkog okvira koji omogućuje da svi učesnici u procesu razvoja, projektovanja i upravljanja softverom međusobno komuniciraju i razumiju se. Pri tome se ne nameće određeni model, tehnika ili aktivnost, nego se sistem zasniva na prethodnom usaglašavanju seta pravila, kojih se svi učesnici u procesu nakon usvajanja moraju pridržavati. Na taj način postupak održavanja softvera, koji u nekim slučajevima može činiti čak 80% ukupne cijene softvera, postaje uređen, efikasan i ekonomičan. Iako je većina međunarodnih standarda o održavanju softvera usvojena kao BAS standard, BiH u oblasti softverskog inžinjerstva, a posebno u sistemskom održavanju softvera, zaostaje u primjeni tih standarda. To je posebno izraženo u primjeni softvera koje su napravile inostrane softverske firme. Dakle, kao i u drugim oblastima, postoji normativni okvir, ali je njegova implementacija loša.

5. LITERATURA

[1] B.P. Lientz, E.B. Swanson (1980) Software Maintenance Management, A Study Of The Maintenance Of Computer Application Software In 487 Data Processing Organizations. Addison-Wesley, Reading MA, ISBN 0-201-04205-3.

[2] ISO/IEC 14764:2006 Software Engineering - Software Life Cycle Processes - Maintenance

[3] Naval Surface Warfare Center, Dahlgren, VA., 1988. - U.S. Naval Historical Center Online Library Photograph NH 96566-KN, Smithsonian National Museum of American History

[4] M.M. Lehman (1980) On Understanding Laws, Evolution, and Conservation in the Large-Program Life Cycle, *Journal of Systems and Software*, 1: 213–221. doi:10.1016/0164-1212(79)90022-0

[5] M.M. Lehman, J.F. Ramil, P.D. Wernick, D.E. Perry, W.M. Turski (1997) Metrics and laws of software evolution-the nineties view. *Proc. 4th International Software Metrics Symposium (METRICS '97)*, 20–32. doi:10.1109/METRIC.1997.637156.

[6] S. Ku, I. Ku, J.H. Yahaya, Z. Mansor, A. Deraman (2017) Towards the Quality Factor of Software Maintenance Process: A Review, Journal of Telecommunication, Electronic and Computer Engineering, e-ISSN: 2289-8131 Vol. 9 No. 3-4, pp 115-118

[7] T. Koponen, V. Hotti (2005) Open Source Software Maintenance Process Framework, *Proc. of Fifth Work. Open Source Softw. Eng.*, 1-5.

[8] P. Bourque, R.E. Fairley (2014) Guide to the Software Engineering Body of Knowledge (SWEBOK®): Version 3.0, ISBN 978-0-7695-5166-1, IEEE Computer Society

[9] ISO/IEC 12207:2008 Systems and Software Engineering - Software Life Cycle Processes

[10] ISO/IEC 15288:2015 Systems Engineering - System Life Cycle Processes

[11] M. Kajko-Mattsson (2006) Applicability of IEEE 1219 within corrective maintenance, *Proceedings of the International Conference on Software Maintenance (ICSMi02)*, 13-13.

[12] IEEE 1219:1998 Standard for Software Maintenance

[13] Z. Stojanov, J. Stojanov, D. Dobrilovic, N. Petrov (2017) Trends in software maintenance tasks distribution among programmers: A study in a micro software company, IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY), Subotica 23-28.

[14] T. Alrawashdeh, M. Muhairat, A. Althunibat (2013) Evaluating the quality of software in ERP systems using the ISO 9126 model, Int. J. Ambient Syst. Appl., vol. 1, no. 1, pp. 1–9.

[15] ISO/IEC 9126-1:2001 Software engineering - Product quality - Part 1: Quality model

[16] D. Kumar (201) Challenges during Software product maintenance, International Journal of Computer Science (IIJCS), vol. 2, no. 3, pp. 52–54.

[17] K.T. Al-Sarayreh, A. Abran, J.J. Cuadrado-Gallego (2013) A standards-based model of system maintainability requirements, J. Softw. Evol. Process, vol. 25, no. 5, pp. 459–505.

[18] T. Muhonen, H. Ailisto, P. Kess (2015) Standards In IOT, Industrial Internet And Condition-Based Maintenance, Internet of Things Finland, 1-2015, pp 56-59.

[19] H. Sutter, A. Alexandrescu (2004) *C++ Coding Standards: 101 Rules, Guidelines, and Best Practices*, Addison-Wesley Professional, https://doc.lagout.org/programmation/C/CPP101.pdf

[20] Katalog BAS standarda http://www.bas.gov.ba