Metode vrednovanja kvalitete programskih proizvoda

Banjavčić, Josipa

Undergraduate thesis / Završni rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Pula / Sveučilište Jurja Dobrile u Puli**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:137:477307

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Image not found or type unknown

Repository / Repozitorij:

Digital Repository Juraj Dobrila University of Pula

zir.nsk.hr

Image not found or type unknown

Sveučilište Jurja Dobrile u Puli

Odjel za informacijsko-komunikacijske tehnologije

JOSIPA BANJAVČIĆ

METODE VREDNOVANJA KVALITETE PROGRAMSKIH PROIZVODA

Završni rad

Pula, listopad, 2017. godine

Sveučilište Jurja Dobrile u Puli
Odjel za informacijsko-komunikacijske tehnologije

JOSIPA BANJAVČIĆ

METODE VREDNOVANJA KVALITETE PROGRAMSKIH PROIZVODA

Završni rad

JMBAG: 0303047593, redovita studentica
Studijski smjer: Informatika

Predmet: Programsko inženjerstvo
Znanstveno područje: Društvene znanosti
Znanstveno polje: Informacijske i komunikacijske znanosti
Znanstvena grana: Informacijski sustavi i informatologija
Mentor: doc. dr. sc. Tihomir Orehovački

Pula, listopad, 2017. godine

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani _____Josipa Banjavčić_____, kandidat za prvostupnika _____informatike_____ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student

____Josipa Banjavčić____

U Puli, __listopad__, _2017._ godine

IZJAVA

o korištenju autorskog djela

Ja, _____Josipa Banjavčić_____ dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj završni rad pod nazivom: _____Metode vrednovanja kvalitete programskih proizvoda_____ koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, _____03.10.2017._____ (datum)

Potpis

____Josipa Banjavčić____

SADRŽAJ

UVOD 1
1. PROGRAMSKI PROIZVOD 3
1.1. Atributi dobrog programskog proizvoda 4
1.2. Trošak programskog proizvoda 6
2. KVALITETA PROGRAMSKOG PROIZVODA 7
2.1. Kvaliteta softvera 8
2.2. CISQ model kvalitete 9
2.3. Standardi kvalitete 9
2.4. ISO standardi: ISO/IEC 25001:2014 11
3. MODELI KVALITETE 14
3.1. Modeli otvorenog izvora 17
3.1.1. CapGemini model otvorenosti 17
3.1.2. Model OpenBRR 17
3.1.3. SQO-OSS model. 18
3.1.4. Model Qualoss 18
4. VREDNOVANJE PROGRAMSKIH PROIZVODA 19
4.1. Vrednovanje kvalitete softverskih proizvoda primjenom ISO/IEC 25000 19
4.2. Vrednovanje softvera: procjena temeljena na kriteriju 20
4.3. Nadzor kvalitete softvera u velikim razmjerima 24
4.3.1. Alati za analizu 24
4.3.2. Model vrednovanja 25
4.3.3. Lab organizacija 25
5. WEB KVALITETA 27
5.1. Metode pregledavanja 29
5.1.1. Heurističko vrednovanje 29
5.1.2. Kognitivna šetnja 30
5.1.3. Multidisciplinarna kognitivna šetnja 30
5.1.4. Pregled funkcionalnih dijelova 30
5.2. Metode testiranja 31
5.2.1. Metoda praćenja oka 31
5.2.1.1. Toplinske mape 32

5.2.1.2. „Saccade“ putanje 32
5.2.1.3. Prednosti i nedostaci metode praćenja oka 33
5.2.2. Udaljeno testiranje 34
5.2.2.1. Prednosti i izazovi udaljenog testiranja 35
5.2.3. Razmišljanje naglas 35
5.2.4. Postavljanje pitanja 36
5.2.5. Automatsko zapisivanje postupaka 36
5.3. Metode ispitivanja 37
5.3.1. Upitnici 37
5.3.1.1. SUS 37
5.3.1.2. WUS 38
5.3.2. Kontekstualno ispitivanje 39
5.3.3. Ciljane skupine 40
5.3.4. Terensko ispitivanje 40
6. KVALITETA MOBILNIH APLIKACIJA 41
6.1. Model mobilne aplikacije 42
6.2. Karakteristike mobilne domene 43
6.3. Vrednovanje mobilnih aplikacija 44
6.4. Bugfender 46
6.5. Robotium 46
ZAKLJUČAK 47
LITERATURA 48
POPIS SLIKA 52
POPIS TABLICA 53
SAŽETAK 54
ABSTRACT 54

UVOD

Od prve pojave softvera pa kroz posljednjih pedesetak godina, softver se konstantno razvijao te tako danas postoje programi, aplikacije i web sustavi koji mogu izvršiti gotovo svaku zadaću koju korisnik pred njih postavi. Naravno, uz tu mogućnost je vezana i kvaliteta koja se nalazi na samom vrhu korisničkih zahtjeva zajedno sa dizajnom i mogućnošću uporabe. Dakle, nekvalitetan softver je loš softver jer korisnici će se uvijek bazirati na pronalazak onog softvera koji će osigurati laku uporabu, na kojima će se lako snaći i koji je ponajprije kvalitetan. Upravo iz toga razloga se razvija pojam vrednovanja softvera. Vrednovanjem, kroz prateće metode, proizvođač traži moguće probleme te pronalazi rješenja za njih kako bi poboljšao samu kvalitetu svog softvera. Postoji velik broj različitih metoda kojima se softver može vrednovati, a odabir odgovarajuće metode ovisi o brojnim čimbenicima.

Vrednovanje programskih proizvoda koji se korite na stolnim računalima i laptopima se razlikuje od vrednovanja mobilnih aplikacija i samim time se javljaju i drukčije metode vrednovanja. Vrednovanje web aplikacija je zanimljivo jer se web aplikacije mogu prilagoditi da se koriste i na stolnim računalima i na mobilnim aplikacijama. Naravno, kada bi se svaka vrsta platforme gledala zasebno, tada bi se mogao napisati cijeli rad samo o vrednovanju mobilnih aplikacija ili web aplikacija. Upravo zbog te opsežnosti, cilj ovog rada je opisati najvažnije dijelove svakog tog aspekta te dati prikaz najvažnijih standarda i metoda vezanih za vrednovanje kvalitete.

Glavna motivacija za vrednovanje kvalitete programskih proizvoda bi bila upravo u tome da se vrednovanjem omogući korištenje proizvoda gdje korisnik neće nailaziti na probleme. Stoga bi programeri trebali sagledati sve aspekte životnog ciklusa softverskog proizvoda ili web sustava te provoditi vrednovanja od samih začetaka tih proizvoda kako bi krajnji rezultat bio upotrebljiv, a samim time i korisnik bio zadovoljan. Tu veliku ulogu preuzimaju metode jer neće svaka metoda biti pogodna za svako vrednovanje, već je važno odrediti ispravnu metodu ili metode. Korištenje metoda bi trebalo utvrditi većinu ili sve potencijalne probleme koji su u konfliktu sa kvalitetom te bi se tada programeri trebali posvetiti rješavanju tih problema. Upravo vrednovanjem i rješavanjem problema, se dolazi do kvalitete, a pogotovo danas je važno koliko će neki softver biti kvalitetan te se konstantno radi na tome da se korisnicima ponuđuju samo kvalitetni programskih proizvodi i kvalitetna web mjesta.

Ovaj rad će se upravo bazirati na različite metode za vrednovanje softvera. Prvo poglavlje će se orijentirati na opis softvera ili programskog proizvoda. Zatim, u drugom poglavlju će biti više riječi o samoj kvaliteti te ulozi ISO standarda kada se govori o njoj. Treće poglavlje će se odnositi na samo vrednovanje kvalitete, a na to se dovezuje četvrto poglavlje koje će se odnositi na modele kvalitete. Peto poglavlje će se bazirati na drukčije vrednovanje te će njime biti opisano vrednovanje softvera na webu, pri čemu će se najviše bazirati na web upotrebljivost. Na kraju slijedi zaključak.

1. PROGRAMSKI PROIZVOD

Programski proizvod (eng. software ili program product) nastaje kako bi bio prodan nekom korisniku. On može biti razvijen za sasvim određenog korisnika ili općenito za tržište. On se može izraditi oblikovanjem novih računalnih programa, (re)konfiguracijom postojećih programa, ili ponovnom uporabom postojećih komponenata programa. Njegov razvoj uvelike se razlikuje od proizvodnje ostalih materijalnih proizvoda. Stoga, modeli koji bi se koristili kod izrade nekih materijalnih proizvoda nisu potpuno primjenjivi i zahtijevaju određene preinake. Uslijed osnovnih razlika samog programskog proizvoda s obzirom na druge materijalne proizvode, programska industrija susreće se s nizom problema kao što su (Sommerville, 2011):

- Programski proizvod je apstraktan i fizički neopipljiv za razliku od bilo kojeg drugog proizvoda. Zbog toga mogućnosti mjerenja i nadzora tijekom njegovog razvoja postaju poprilično ograničene.
- Osnova za razvoj programskog proizvoda je ljudska inovativnost, kreativnost, vještina i znanje, za razliku od proizvodnje bilo kojeg drugog proizvoda gdje se proizvodnja najčešće temelji na materijalnim vrijednostima, što otežava upravljanje proces njegova razvoja.
- Zbog nesavršenosti čovjekovog rada, potrebno je uložiti veliki napor u verifikacijske aktivnosti kako bi se ispunili zahtjevi koje zahtijeva tržište, ali i udovoljili visoki zahtjevi za kvalitetom. Pod verifikacijskim aktivnostima se podrazumijevaju sve aktivnosti koje služe za provjeru i ocjenu rezultata određene faze unutar procesa životnog ciklusa programskog proizvoda.
- Programski proizvod odmah nakon faze razvoja odlazi u primjenu i nije potrebna faza masovne produkcije za njegovu ponudu na tržištu, a to se postiže zbog njegove jednostavne multiplikacije.
- Kod proizvodnje programskog proizvoda svaki novi razvoj nadograđuje se na već razvijenoj osnovi pa stoga kompleksnost programskog proizvoda je u kontinuiranom porastu. Taj razvoj programskog proizvoda čini evolucijske, iterativne, spiralne i slični modele procesa životnog ciklusa prikladnijima od klasičnog modela vodopada kod kojega se aktivnosti odvijaju kao faze sekvencijalno jedna iza druge.

- Razvoja takvih kompleksnih programskih sustava ima rastuće potrebe za ljudskim resursima određenih znanja i vještina koji bi razvijali takve programske proizvode. Da bi se udovoljilo takvim potrebama poduzeća često distribuiraju svoje poslove na nekoliko odvojenih lokacija, koje su sve češće globalno distribuirane.

1.1. Atributi dobrog programskog proizvoda

Za postizanje konkurentne prednosti potrebno je brzo, učinkovito i kontinuirano prilagođavanje zahtjevima okoline, što se omogućuje kontinuiranim provođenjem poboljšanja procesa razvoja programskog proizvoda koja su ugrađena u svaki od programa učinkovitog upravljanja kvalitetom. Budući da programski proizvod mora biti kvalitetan, od njega se očekuje da ima određene atribute (tablica 1.).

Tablica 1. Atributi kvalitete (Sommerville, 2011, str. 656)

Sigurnost	**Razumljivost**	**Prenosivost**
Zaštita	**Testiranje**	**Upotrebljivost**
Pouzdanost	**Prilagodba**	**Mogućost održavanja**
Fleksibilnost	**Modeliranje**	**Efikasnost**
Robusnost	**Složenost**	**Sposobnost učenja**

Tablicom 1 su prikazani atributi važni za kvalitetu programskog proizvoda, a najvažniji od njih su (Sommerville, 2011, str. 7-14):

- Mogućnost održavanja: programski proizvod se mora moći mijenjati ili prilagođavati u skladu s promijenjenim potrebama korisnika.
- Pouzdanost i sigurnost: programski proizvod se mora ponašati na predvidiv način te ne smije izazivati fizičke ili ekonomske štete.
- Efikasnost: programski proizvod mora imati zadovoljavajuće performanse, ali i upravljati strojnim resursima na štedljiv način.

- Upotrebljivost: programski proizvod treba raditi ono što korisnici od njega očekuju, sučelje mu treba biti zadovoljavajuće te za njega mora postojati pripadna dokumentacija.

Naravno, svaki korisnik ima svoja očekivanja i poglede na to što znači dobar proizvod. Tako je za kupca dobar programski proizvod onaj koji rješava problem uz prihvatljivu cijenu. Za krajnjeg korisnika, to je onaj proizvod koji se lagano nauči i prihvaća i pomaže da se posao lakše obavi. Za osobu koja razvija i oblikuje, to je onaj proizvod koji se lagano oblikuje i održava i koji se može ponovno iskoristiti.

Sustav još uvijek nije moguće optimizirati za sve navedene atribute. Stoga, plan kvalitete treba definirati najvažnije svojstvo kvalitete za softver koji se razvija. Ponekada je važno da je softver učinkovit i drugi atributi moraju biti žrtvovani kako bi se to postiglo.

Postoji jasna veza između procesa i kvalitete proizvoda u proizvodnji jer proces je relativno lako standardizirati i nadzirati. Jednom proizvedeni sustavi se mogu ponovno i ponovno pokrenuti kako bi se dobio proizvod visoke kvalitete. Međutim, takav softver nije proizveden, on je dizajniran. U razvoj softvera, odnos između kvalitete procesa i kvalitete proizvoda je složeniji. Razvoj softvera je kreativan, a ne mehanički proces, tako da je utjecaj pojedinih vještina i iskustvo vrlo značajno. Vanjski čimbenici, kao što su noviteti temeljeni na zahtjevima ili komercijalni pritisak na rani proizvoda također utječu na kvalitetu proizvoda bez obzira na postupak koji se koristi.

Nema sumnje da proces razvoja ima značajan utjecaj za kvalitetu softvera i da dobro provedeni procesi imaju višu vjerojatnost da će dovesti do dobre kvalitete. Također, proces upravljanja kvalitetom i njeno poboljšanje može dovesti do manje nedostataka u softveru koji se razvija. Međutim, teško je procijeniti kvalitetna svojstva, kao što su održivost, bez korištenja softvera na duže razdoblje. Prema tome, teško je reći kako karakteristike procesa utječu na atribute kvalitete. Nadalje, zbog uloge dizajna i kreativnosti u softveru, proces standardizacije ponekad može ugušiti kreativnost, što dovodi do lošije kvalitete softvera (Sommerville, 2011).

1.2. Trošak programskog proizvoda

O programskom proizvodu se ne može govoriti bez da se obrati pažnja na sam trošak. Trošak je bitan aspekt koji određuje je li razvoj proizvoda isplativ ili ne. U pravilu, programski proizvod razvija se tako da njegova korist bude veća od svih troškova vezanih uz razvoj i održavanje. Cijena programske potpore dobiva se od cijena specifikacije, oblikovanja, ispitivanja i održavanja (evolucije). Cijena ovisi o tipu programskog sustava, zahtjevima, traženim performansama i traženoj razini pouzdanosti. U svemu tome potrebno je naglasiti da postoji tržišno natjecanje u proizvodnji programske potpore, što znači da proizvod mora biti konkurentan i po pitanju cijene i po pitanju brzine izlaska na tržište, uz zadržavanje performansi i pouzdanosti. Za programske proizvode koji imaju dugi vijek trajanja (10 i više godina), kao što su razni kontrolni sustavi, trošak promjene, prilagodbe i održavanja (evolucije) je često višestruko veći od troška izvornog razvoja (Sommerville, 2011).

Manji poslovni programski proizvodi često imaju kraći rok trajanja i posljedično manji trošak evolucije. Prilikom razvoja programskog proizvoda mogući su i sljedeći scenariji. Primjerice, povećanje efikasnosti specijalizacijom čini programski proizvod manje razumljivim i može smanjiti mogućnost održavanja ili ponovnog korištenja. Povećanje lakoće korištenja, kao što je uključivanje uputa tijekom rada, može smanjiti efikasnost. Upravo stoga je nužno unaprijed postaviti razinu kvalitete, što je ključna inženjerska aktivnost. Programski proizvod mora biti dovoljno dobar te se pritom rade stalni kompromisi i optimizacije ograničenih resursa. Dobra inženjerska praksa kod razvoja i evolucije proizvoda je da se izbjegava suvišan posao dodavanja funkcionalnosti koje korisnik nije tražio, jer se time štedi i vrijeme i novac.

2. KVALITETA PROGRAMSKOG PROIZVODA

Kada se govori o kvaliteti, ona se najbolje može definirati od strane Međunarodne organizacije za normizaciju (ISO). Prema normi ISO 9000 kvaliteta je stupanj do kojeg skup svojstvenih karakteristika ispunjava zahtjeve (ISO/IEC, 2011).

Kada je riječ o programskom proizvodu, kvaliteta se stavlja kao prioritet u njegovom razvoju, ali i u njegovoj uporabi. Problem kvalitete softvera je otkriven 1960-ih s razvojem prvih velikih programskih sustava, te je on nastavio mučiti softverske inženjere tijekom 20. stoljeća. U početcima se je isporučivao softver koji je bio spor i nepouzdan. Bilo ga je teško održavati, ali i teško ponovno upotrijebiti. Kao rezultat se je javilo nezadovoljstvo korisnika koje je dovelo do donošenja formalnih tehnika upravljanja kvalitetom softvera, koje su bile razvijene od metoda korištenih u prerađivačkoj industriji (Sommerville, 2011). Te tehnike upravljanja kvalitetom, u suradnji s novim softverskim tehnologijama i boljim softverskim testiranjima, doveli su do značajnih poboljšanja u općoj razini kvalitete softvera.

Kada je riječ o kvaliteti, pojmovi "osiguranje kvalitete" i "kontrola kvalitete" se široko koriste u proizvodnoj industriji. Osiguranje kvalitete (QA) je definicija procesa i standarda koji bi trebali dovesti do visokokvalitetnih proizvoda, ali i uvođenja kvalitetnih procesa u samu proizvodnju. Kontrola kvalitete (QC) je primjena kvaliteta osiguranja u proizvodne procese kako bi uklonili proizvode koji nisu na potrebnoj razini kvalitete. Kada se ti pojmovi gledaju u softverskoj industriji, različite tvrtke i industrije sektorima interpretiraju osiguranje kvalitete i kontrolu kvalitete na različite načine. Ponekad, osiguranje kvalitete jednostavno znači definiciju postupaka, procesa i standarda koji su usmjereni na osiguranje koja kvalitete softvera je postignuta. U drugim slučajevima, osiguranje kvalitete također uključuje sve upravljanje konfiguracijom, provjere i validacije aktivnosti koje se primjenjuju nakon što je proizvod predan od strane razvojnog tima (Sommerville, 2011, str. 652-657).

2.1. Kvaliteta softvera

Kada kvalitetu softvera promatramo sa stajališta softverskog inženjeringa i kada na nju gledamo u poslovnom kontekstu, tada ju možemo podijeliti na: kvalitetu funkcionalnih funkcija i kvalitetu strukture softvera.

Kvaliteta funkcionalne funkcije softvera odražava koliko je udovoljena kvaliteta ili koliko je u skladu s određenim dizajnom, a temelji se na funkcionalnim zahtjevima ili specifikaciji. Funkcionalna funkcija također se može opisati kao prikladnost za svrhu dijela softvera ili kako se uspoređuje s konkurentima na tržištu kao vrijedan proizvod. Ona također predstavlja i stupanj do kojeg je proizveden ispravan softver. Funkcionalna kvaliteta se obično procjenjuje dinamički, ali je također moguće koristiti statična ispitivanja (kao što su recenzije softvera) (Pressman, 2005).

Strukturalna kvaliteta softvera odnosi se na način na koji su zadovoljeni nefunkcionalni zahtjevi koji podržavaju isporuku funkcionalnih zahtjeva, kao što su robusnost ili održivost. Ona ima puno više veze s stupanjem na koji softver funkcionira po potrebi. Mnogi aspekti strukturne kvalitete mogu se procijeniti samo statičkom analizom unutarnje strukture softvera, zatim analizom njegovog izvornog koda, analizom tehnologije i drugim analizama. Neke strukturalne kvalitete, kao što je upotrebljivost, mogu se procjenjivati samo dinamički, dok aspekti kao što je upotrebljivost, mogu se procjenjivati i statički i dinamički (CISQ, 2017).

Nadalje, vrlo su važni i razlozi zbog kojih se ispituje kvaliteta softvera. Sa poslovnog stajališta ti razlozi se mogu podijeliti u dvije skupine (Pressman, 2005):

- Upravljanje rizicima: Vrlo važan dio kod mjerenja kvalitete jer pogreške mogu uzrokovati više od neugodnosti. Budući da se softver koristi u gotovo svim aspektima ljudskog djelovanja, kvaliteta se ovdje odnosi na sigurnost svih koji dolaze u doticaj sa takvim softverom. Na primjer, zrakoplovi se mogu svrstati u tu kategoriju jer neispravan softver može dovesti do posljedica kao što je pad zrakoplova.
- Upravljanje troškovima: Softver koji ima dobru strukturu i dobru kvalitetu će biti „jeftiniji“ za održavanje u odnosu na softver koji mu je suprotnost. Problemi održavanja se najčešće javljaju kod korištenja sustava za planiranje resursa poduzeća (ERP), sustava za upravljanje odnosa sa klijentima (CRM) ili

sustavima obrade velikih transakcija. Loša strukturna kvaliteta snažno je povezana i sa prekidima poslovanja zbog oštećenih podataka, prekida aplikacija, kršenja sigurnosti ili problema s performansama.

2.2. CISQ model kvalitete

IT organizacije mogu koristiti standarde kvalitete koda za otkrivanje i kvantificiranje kritičnih područja kodiranja u softveru. Ti standardi su razvijeni od strane CISQ grupe (eng. Consortium for IT Software Quality). U središtu CISQ standarda su nefunkcionalni zahtjevi softvera koji se mogu pratiti do najvećih štetnih kvarova sustava i kršenja sigurnosti. Mjerenje je osmišljeno da bude automatizirano na izvornom kodu kako bi identificirali kritični propusti u softveru koji su dovoljno ozbiljni da moraju biti popravljeni. CISQ mjere kvalitete pokrivaju osamdeset i šest dobro uspostavljenih inženjering pravila kako bi se osigurao siguran, pouzdan, učinkovit i jednostavan način za održavanje softvera. Mjere su u skladu s ISO/IEC 25010 definicijom, a odnose se na (CISQ, 2017):

- Pouzdanost: mjeri razinu rizika i vjerojatnost potencijalnih kvarova,
- Učinkovitost: važna za aplikacije koje moraju postići visoke performanse,
- Sigurnost: mjeri vjerojatnost potencijalnih kršenja sigurnosti koji mogu nastati zbog grešaka prilikom kodiranja ili izgradnje arhitekture,
- Sposobnost prilagodljivosti: važna za aplikacije koje se mogu koristiti u raznim okružjima i gdje je važno kako će aplikacija reagirati na promjene.

2.3. Standardi kvalitete

Standardi su stvoreni kako bi davali vrijednost u obliku povećane kvalitete proizvoda. Nema smisla definirati standard koji je skup i koji u smislu vremena i truda ne dovodi do poboljšanja u kvaliteti. Standard proizvoda mora biti osmišljen tako da se može primjenjivati i provjeravati na ekonomičan način, a procesni standard treba sadržavati definiciju procesa koja provjerava da su se standardi proizvoda slijedili.

Različite vrste softvera trebaju različite razvojne procese tako da standardi moraju biti prilagodljivi. Nema smisla propisivati određeni način rada, pogotovo ako je on neprikladan za projekt ili projektni tim. Svaki voditelj projekta treba imati ovlast za izmjenu procesnih normi prema individualnim okolnostima. Međutim, kada se naprave izmjene, važno je osigurati da promjene ne dovode do gubitka kvalitete proizvoda jer to može utjecati na odnos organizacije sa svojim kupcima i vjerojatno će dovesti do povećanja troškova projekta. Voditelj projekta i voditelj kvalitete mogu izbjeći probleme neprikladnih normi ako pomno planiraju atribute kvalitete rano u projektu. Oni bi trebali odlučiti koji se od organizacijskih standarda treba koristiti bez promjene, što bi se trebalo mijenjati, a što bi trebalo biti zanemareno (Test-institute, 2017).

Softverski standardi imaju vrlo važnu ulogu u upravljanju kvalitetom softvera. Kao važan dio osiguranja kvalitete, javlja se definicija ili odabir standarda koji bi trebao primijeniti na proces razvoja softvera ili softverskog proizvoda. Kao dio tog procesa osiguranja kvalitete, alati i metode za potporu korištenje tih standarda također se mogu odabrati. Nakon što su standardi odabrani za uporabu, projektno specifični procesi moraju biti definirani kako bi pratili primjenu normi i provjerili da su one bile provedene.

Za standarde kvalitete se može reći da su važni iz tri razloga (Test-institute, 2017):

1. Standardi predstavljaju vrijednost za organizaciju. Oni se temelje na znanju o najboljoj ili najprikladnijoj praksi za tvrtku. To znanje je često rezultat mnogih pokušaja i pogrešaka. Izgradnjom standarda organizacija gradi uspješnu poslovnu praksu, ali i izbjegava prethodne greške.
2. Standardi osiguravaju okvir za definiranje onoga što 'kvaliteta' znači u određenom smislu. Budući da je kvalitetan softver subjektivan, standard pomaže uspostaviti osnovu za odlučivanje i prikazuje jesu li postignute obvezne razine kvalitete. Naravno, to ovisi o postavljanju standarda koji odražavaju očekivanja korisnika za softversku pouzdanost, upotrebljivost i performanse.
3. Standardi pripomažu kontinuitetu kada rad jedne osobe preuzima i nastavlja neka druga osoba. Standardi osiguravaju da svi inženjeri unutar organizacije usvojiti iste prakse kako bi svaki mogao nastaviti dani posao.

Razvoj međunarodnih standarda za programsko inženjerstvo je obično produljen proces u kojem se susreću oni koji su zainteresirani za standard i u kojemu se na kraju dogovaraju o najboljim standardima. Nacionalna i međunarodna tijelima kao što su US

DoD, ANSI, BSI, NATO i IEEE podržavaju nastanak i primjenu standarda. Tijela kao što su NATO mogu zahtijevati da se njihovi standardi koriste zajedno sa drugim standardima (Test-institute, 2017).

2.4. ISO standardi: ISO/IEC 25001:2014

Od mnoštva standarda koji postoje pod kraticom ISO (eng. International Organization for Standardization), za potrebe ovoga rada je najbitniji standard ISO/IEC 25001:2014 (ISO/IEC, 2014). Službeni naziv tog standarda je: Sistem i softversko inženjerstvo – Sistemski i softverski kvalitetni zahtjevi i vrednovanje – Planiranje i upravljanje (eng. Systems and software engineering – Systems and software Quality Requirements and Evaluation – Planning and management (dalje u tekstu kao SQuaRE)). Ovaj standard osigurava uvjere i preporuke za organizaciju koja je odgovorna za provedbu i upravljanje sustavima, te za zahtjeve kvalitete i aktivnosti i metode vrednovanja o kojima će biti riječi u daljnjem tekstu.

Ova međunarodna norma sadrži pojedinosti o zahtjevima planiranja i upravljanja povezane sa sustavima i softverskim zahtjevima kvalitete proizvoda i ocjenjivanja. Standard ISO/IEC 25001:2014 proizlazi iz standarda ISO/IEC 25001:2007, ali također i zamjenjuje ISO/IEC 9126 i ISO/IEC 14598 standarde. Standard se sastoji od sljedećih odjeljaka (ISO/IEC,2014):

- ISO / IEC 2500n, Sektor upravljanja kvalitetom,
- ISO / IEC 2501n, Model kvalitetne podjele,
- ISO / IEC 2502n, Mjerenje kvalitetne podjele,
- ISO / IEC 2503n, Podjela zahtjeva kvalitete
- ISO / IEC 2504n, Vrednovanje kvalitete.

AD1) Sektor upravljanja kvalitetom pripada u međunarodne standarde koji definiraju sve zajedničke modele, pojmove i definicije upućene iz svih ostalih standarda SQuaRE serije. Ova norma predlaže primjenu odgovarajućih standarda za određene aplikacije kako bi se njihovom primjenom pružila pomoć svim vrstama korisnika. Standard također pruža uvjete i smjernice za potpornu funkciju rukovodstvu koje je odgovorno za upravljanje specifikacijama zahtjeva proizvoda i vrednovanje.

AD2) Model podjele kvalitete predstavlja detaljne modele za kvalitete sustava i softverskih proizvoda, ali i modele za kvalitetu u uporabi i podatke.

AD3) Norma mjerenja kvalitetne podjele uključuje model za mjerenje kvalitete proizvoda i softvera, zatim matematičke definicije mjerenja kvalitete kao i upute za njihovu primjenu. Ovaj odjeljak predstavlja unutarnje mjere za kvalitetu softvera, vanjske mjere sustava ili softvera za kvalitetu proizvoda i mjere za kvalitetu korištenja.

AD4) Kao sljedeći u SQuaRE seriji, odjeljak podjele zahtjeva kvalitete predstavlja standard koji olakšava utvrđivanje zahtjeva kvalitete. Zahtjevi definirani ovim odjeljkom mogu se koristit u procesu stvaranja proizvoda ili biti primjenjivani za postupak njihovog vrednovanja.

AD5) Odjeljak vrednovanja kvalitete osigurava zahtjeve preporuke i smjernice za procjenu proizvoda bez obzira da li se oni izvode od strane nezavisnih procjenitelja, programera ili nekog drugog. Ova norma se dijeli na dodatne odjeljke, a to su:

- ISO/IEC 25040 - proces vrednovanja (evaluacije): sadrži opće uvjete za specifikaciju i ocjenjivanje kvalitete softvera i pojašnjava opće pojmove. Daje opis procesa za procjenu kvalitete softverskih proizvoda, te navodi uvjete za primjenu ovog postupka. Postupak vrednovanja je osnova za procjenu kvalitete programskog proizvoda te ima različite svrhe i pristupe. Stoga se postupak može primijeniti za procjenu kvalitete u uporabi, vanjske mjere kvalitete softvera. unutarnje mjere kvalitete softvera, ali i za procjenu kvalitete razvijanja softver ili prilagođavanja softvera tijekom razvojnog procesa. Vrednovanje kvalitete softverskih proizvoda može biti provedeno od strane stjecatelja ili organizacije, programera ili pak nezavisnog procjenitelja.
- ISO/IEC 25041 - Sadrži upute za programere, preuzimatelja i vrednovatelja koje se odnose na specifične zahtjevi i preporuke za vrednovanje.
- ISO/IEC 25042 – Moduli za vrednovanje određuju strukturu i sadržaj dokumentacije koja se koristi za opis modula ocjenjivanja. Ovi moduli sadrže specifikaciju kvalitete modela (obilježja, karakteristike i odgovarajuće unutarnje i vanjske mjere uporabe), povezane podatke i podatke o planiranoj primjeni modela kao i podatke o njegovoj stvarnoj primjeni. Pravilni moduli su odabrani za svaku procjenu. U nekim slučajevima možda će biti potrebno razviti nove module ocjenjivanja. Smjernice za razvoj novih modula za ocjenjivanje nalaze

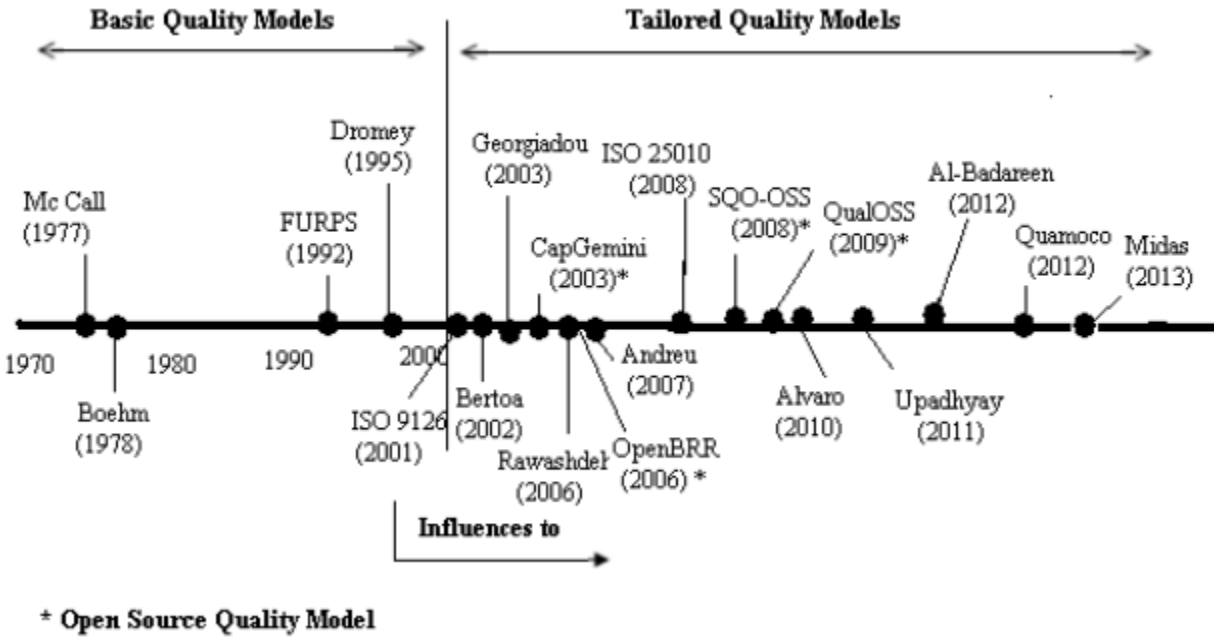
se u ISO/IEC 25042. Ova međunarodna norma također se može koristiti od strane organizacija koje proizvodi nove module ocjenjivanja.

- ISO/IEC 25045 - modul vrednovanja za oporavak daje specifikaciju kako bi se procijenile karakteristike oporavka definirane pod obilježjem pouzdanosti kvalitete modela. On određuje vanjske mjere kvalitete softvera za otpornost i indeks autonomnog oporavka kad je informacijski sustav sastavljen od jednu ili više softverskih proizvoda podvrgnut nizu poremećaja. Poremećaj može biti operativan (npr. naglo zatvaranje proces operacijskog sustava) ili neki događaj (npr. značajan porast korisnika u sustavu).

3. MODELI KVALITETE

Modeli kvalitete se počinju razvijati već 70-ih godina prošlog stoljeća i do danas postoji veliki broj modela. Slika 1. ažurira rad Thapa (Miguel et al., 2014) i pokazuje evoluciju modela kvalitete od prvog modela Mc Call iz 1977. pa do 2013. godine. Ova evolucija je kategorizirala modele u:

- Osnovne modele (1977. – 2001.): čiji je cilj ukupna i sveobuhvatna procjena proizvoda,
- Prilagođeni modeli kvalitete (od 2001. nadalje): orijentirani na procjene komponenata. U ovim modelima su također uvršteni i modeli orijentirani na evaluaciju slobodnog softvera.



Slika 1. Modeli kvalitete prema Thapu (Miguel et al., 2014, str. 32)

Slijedeći vremensku skalu na slici 1., glavni modeli osnovne kvalitete su (Miguel et al., 2014):

- Mc Call Model: uspostavio je kvalitetu proizvoda kroz nekoliko značajki. Oni su bili grupirani u tri perspektive: pregled proizvoda (održavanje, fleksibilnost i testiranje), rad proizvoda (ispravan, pouzdan, učinkovit, integritet i upotrebljivost) i prijelaz proizvoda (prenosivost, ponovna upotreba i

interoperabilnost). Glavni doprinos McCallove metode bio je razmotriti odnose između kvalitetnih obilježja i mjernih podataka. Ovaj model korišten je kao baza za stvaranje drugih modela kvalitete. Glavni nedostatak modela Call Mac je preciznost u mjerenju kvalitete, jer se temelji na odgovorima da ili ne. Nadalje, model ne uzima u obzir funkcionalnost tako da je smanjena vidljivost korisnika.

- Boehmov model: Boehm uspostavlja značajke velikih razmjera koji predstavljaju poboljšanje u odnosu na model Mc Call, jer dodaje čimbenike na različitim razinama. Čimbenici visoke razine su:
    - a) Uslužni program koji ukazuje na lakoću, pouzdanost i učinkovitost korištenja softverskog proizvoda,
    - b) održivost koja opisuje objekte za izmjenu, provjerljivost i aspekte razumijevanja,
    - c) prenosivost u smislu mogućnosti daljnje upotrebe s promjenom okoliša.
- Dromeyov model: Model Dromey temelji se na perspektivi kvalitete proizvoda. Na taj se način procjena kvalitete svakog proizvoda razlikuje i uspostavlja se dinamičnija procjena. Model navodi da za kvalitetan proizvod, svi elementi koji ga čine, trebaju biti tako. Međutim, ne postoji rasprava o tome kako se to može učiniti u praksi, a ovaj teorijski model koristi se za dizajniranje drugih specifičnijih modela.
- Model FURPS: Model kategorizira karakteristike kao funkcionalne zahtjeve (RF) i nefunkcionalne (NF). RF su definirani očekivanim ulazima i izlazima ili funkcionalnost (F) dok su NF grupirani kao upotrebljivost (U), pouzdanost (R), izvođenje (P) i podrška za proizvod (S). Njegov glavni problem je da se neke glavne značajke, poput prenosivosti, ne uzimaju u obzir.
- ISO 25010 model: Ovaj standard pojavio se u 2007. ažuriranju modela ISO 9126 koji je bio korišten kao osnova za modele kvalitete po mjeri. Podijeljen je na 8 obilježja i karakteristika. Izrađuju skup standarda temeljenih na ISO 9126, a jedan od njegovih glavnih ciljeva je voditi u razvoju softverskih proizvoda s specifikacijom i procjenom zahtjeva kvalitete. Ovaj model smatra novim karakteristikama sigurnost i kompatibilnost koja grupira neke od prvih karakteristika prenosivosti i one koje nisu logički dio prijenosa iz jednog okruženja u drugu. Ona koristi izraz transferabilno kao produžetak prenosivosti.

Kao i kod ISO / IEC 9126, ovaj standard održava tri različita gledanja u proučavanju kvalitete proizvoda.

Nadalje, prateći vremensku skalu sa slike 1. dolazi se do prilagođenih modela kvalitete. Od 2001. godine razvoj softvera temeljen je na komponentama (CBSD). Ne-osnovni modeli Razvoj softvera usredotočen je na korištenje komercijalno dostupnih prijevoda (eng. COTS - Commercial off-the-shelf). Tako imamo (Miguel et al., 2014):

- Bertoa model: Model kvalitete Bertoa temelji se na modelu ISO 9126. Ona definira skup atributa kvalitete za učinkovitu procjenu COTS-a. COTS koriste tvrtke za razvoj softvera za izgradnju složenijih programa. Model razlikuje one značajke koje imaju smisla za pojedine.
- GEQUAMO: Ovaj model nazvan GEQUAMO (generički, višeslojni i prilagodljiv model) stvorio je E. Georgiadou i sastoji se od postupnog dijeljenja u predloške značajki i karakteristika, a namijenjen je za enkapsulaciju različitih korisničkih zahtjeva na dinamičan i fleksibilan način , U ovom obliku korisnik (krajnji korisnik, programer i upravitelj) može izraditi vlastiti model koji odražava naglasak (težinu) svakog atributa ili zahtjeva.
- Alvaro model: Alvaro metoda smatra okvir za certifikaciju softverskih komponenti) kako bi se utvrdili elementi komponenata kvalitete. Ovaj okvir razmatra četiri modula:
    - 1) komponente kvalitete modela u svrhu određivanja karakteristika koje treba uzeti u obzir,
    - 2) okvir za tehničku certifikaciju koja određuje tehnike koje će se koristiti za ocjenu značajki koje nudi model,
    - 3) proces certificiranja koji definira skup tehnika koje procjenjuju i potvrđuju softverske komponente s ciljem uspostavljanja dobro definiranog standarda certificiranja i
    - 4) okvira koji sadrži mjerni podatak koji je odgovoran za definiranje skupa metrika koje vrednuju svojstva komponenata na kontrolirani način.
- Rawashdeh model: Model Rawashdeh kao glavni cilj ima potrebe različitih vrsta korisnika. Model se usredotočuje na korištenje komponenti COTS i na njega utječe ISO 9126 i Dromey modeli. Model određuje četiri koraka za stvaranje modela kvalitete proizvoda koji uključuju utvrđivanje male skupine atributa

visoke kvalitete, a zatim pomoću top down tehnike svaki atribut razgrađuje se u niz podređenih atributa. Prilikom toga treba razlikovati interne i eksterne mjerne podatke.

3.1. Modeli otvorenog izvora

Stvarno besplatni softverski proizvodi imaju veliku popularnost za različite karakteristike i slobode koje nude i zato što se koriste u različitim kontekstima. Mnogi od njih su usmjereni na obavljanje iste ili slične radnje temeljene na tradicionalnim proizvodima (Miguel et al., 2014). Primjerice, oni mogu biti operativni sustavi slobodnog softvera (kao što su Linux, Solaris, FreeBSD), middleware tehnologije ili baze podataka (Apache Web Server, MySQL) ili proizvodi za krajnjeg korisnika (Mozilla Firefox, Open Office).

Modeli za procjenu kvalitete proizvoda slobodnog softvera prilagođavaju modele, poput ISO modela, dodajući neke posebne aspekte slobodnog softvera. Važno je napomenuti da, iako postoji razlika između modela prve i druge generacije, još nije definiran idealan model koji obuhvaća sve aspekte kvalitete u slobodnom softverskom proizvodu (Miguel et al., 2014).

3.1.1. CapGemini model otvorenosti

Model se temelji na zrelosti proizvoda i postavljen je prema pokazateljima zrelosti. Ovi su pokazatelji grupirani u indikatore proizvoda i primjene. Za konačnu evaluaciju, svaki od podizbornika dobiva vrijednost između 1 i 5, dajući ukupni rezultat (Miguel et al., 2014).

3.1.2. Model OpenBRR

Model se naziva okvirom ocjene poslovne spremnosti i pod utjecajem je modela CapGemini i ISO 9126. U ovom kontekstu se identificiraju kategorije koje su važne za

procjenu otvorenog softvera. Model ima sedam kategorija i time ubrzava proces evaluacije, osiguravajući bolji izbor s malim setom. Sedam kategorija može se rafinirati radi veće granuliranosti i pokriti aspekte koji nisu uzeti u obzir na najvišoj razini. Cilj je uvijek držati na vrlo jednostavnoj razini (Miguel et al., 2014).

3.1.3. SQO-OSS model.

Ovo je hijerarhijski model koji procjenjuje izvorni kod i proces zajednice koji omogućuje automatsko izračunavanje mjernih podataka. Model se razlikuje od drugih u sljedećim aspektima:

- Jezgra je kontinuiranog sustava praćenja kvalitete i omogućuje automatsko prikupljanje mjernih podataka.
- Ne procjenjuje funkcionalnost.
- Usmjeren je na izvorni kod. Izvorni kod je najvažniji dio softverskog projekta.
- Razmatra samo faktore koji se mogu automatski mjeriti (Miguel et al., 2014).

3.1.4. Model Qualoss

Model Qualoss naglašava tri aspekta (Miguel et al., 2014):

1) karakteristike proizvoda,
2) karakteristike zajednice i
3) karakteristike softverskih procesa koji se jednako važni za kvalitetu slobodnog (open source) proizvoda.

Qualoss model navodi da je kvaliteta vrlo ovisna o kontekstu u kojem se koristi u svrhu koju tvrtka ili osoba slijedi s njom. Ovaj model odgovara drugoj generaciji besplatnih (eng. open source) modela i gdje je većina procjene visoko automatizirana.

4. VREDNOVANJE PROGRAMSKIH PROIZVODA

Vrednovanje programskog proizvoda predstavlja glavni način za utvrđivanje kvalitete i provedbom raznih metoda se mogu otkriti potencijalni problemi i tražiti njihova rješenja. U nastavku se pobliže objašnjava važnost vrednovanja.

4.1. Vrednovanje kvalitete softverskih proizvoda primjenom ISO/IEC 25000

Budući da posljednjih godina kvalitetan softver ima veliki značaj, prvenstveno zbog ključne uloge softverskih proizvoda u svakodnevnom životu, vrednovanje njegove kvalitete je također značajan proces bez obzira obavljala li se to vrednovanje prilikom njegovog nastanka ili uporabe (ISO/IEC, 2011).

Iako postoje brojni certifikati za kvalitetu softvera (npr. ISO/IEC15504, CMMI, itd.), malo je dokaza koji ukazuju da usklađenost sa bilo kojim od ovih standarda jamči dobre softverske proizvode. Kritičari su otišli tako daleko da kažu da jedina stvar koju ti standardi jamče je ujednačenost proizvodnje i njihova primjena zapravo može dovesti do proizvodnje loših proizvoda. Prema tome, ideja da se procjena softvera treba temeljiti na proizvodnim atributima postaje sve raširenija (ISO/IEC, 2011). Dakle, sve veći broj organizacija postaje zabrinuto zbog kvalitete proizvoda koji se razvijaju ili stječu, ali i zbog kvalitete procesa kojima se dolazi do tih proizvoda.

SQuaRE obitelj standarda stoga predstavlja najbolji odabir za vrednovanje softverskih proizvoda. Kao i kod drugih standarda, ISO/IEC 25000 opisuje što se treba procijeniti, ali ne precizira kako. Drugim riječima, on ne daje detaljne granice vrednovanja, niti opisuje kako grupirati te granice da bi se dobila kvalitetna vrijednost programskog proizvoda (ISO/IEC, 2011).

Poboljšanje tih procesa ocjenjivanja je postao temeljni cilj AQC-a tima tijekom posljednjih nekoliko godina. To je dovelo do stvaranja AQC Laba (aqclab.es) kao prvog laboratorija ovlaštenog za procjenu kvalitete softverskih proizvoda koji koriste ISO/IEC 25000. Laboratorij je također prepoznat od strane ILAC (eng. International Laboratory Accreditation Cooperation). AQC Lab koristi tri glavna elementa za provođenje procjene kvalitete, a oni su (AQC Lab, 2017):

- Postupak procjene koji je izravno usvaja aktivnosti standarda ISO/IEC 2504n i nadopunjuje ih s određenim laboratorijskim ulogama i uputama koje su razvijene u unutrašnjosti. Ovaj proces stvara evaluacijsko izvješće koje prikazuje razinu kvalitete koju postiže proizvod kao i bilo koje softverske aspekte koje trebaju poboljšanje.
- Model kvalitete koji definira karakteristike i potrebne mjere za procjenu softverskog proizvoda. Ovaj model je razvijen kroz MEDUSAS - eng. Improvement and Evaluation of Usability, Security and Maintainability of Software (poboljšanje i vrednovanje upotrebljivosti, sigurnosti i održavanja softvera) istraživački projekt financiran od strane MICINN/FEDER-a. Iako AQC Lab ocjenjuje više obilježja ISO/IEC 2501n standarda, početna procjena se usredotočuje na obilježje održivosti, prvenstveno zbog toga što je održavanje jedna od najskupljih faza razvoja životnog ciklusa softverskog proizvoda. Održavanja je i jedna od značajki koju softverski klijenti najčešće traže, te su najtraženiji proizvode koji se održavaju ili se mogu održavati od strane treće osobe. Ovaj model zahtijeva značajan napor da se razvije, što na kraju rezultira nizom kvalitetnih svojstava koja su vidljiva iz izvornog koda i odnose se na obilježja kvalitete predloženih u ISO/IEC2501n
- Vrednovanje okoliša je sljedeći element koji u velikoj mjeri nastoji automatizirati zadatke ocjenjivanja. To okruženje koristi alate za mjerenje koji se primjenjuju na softverskom proizvodu kako bi kombinirati dobivene vrijednosti, dodijelili razinu kvalitete na modelu obilježja i na kraju predstaviti ih na lako dostupan način.

4.2.Vrednovanje softvera: procjena temeljena na kriteriju

Procjena temeljena na kriteriju je kvantitativna procjena softvera u smislu održivosti i upotrebljivosti. To može pridonijeti donošenju odluka na visokoj razini na određenim područjima za poboljšanje softvera. Procjena kriterija daje mjere kvalitete u brojnim područjima (Jackson et al., 2011). Ta područja su izvedena iz ISO/IEC 9126-1 standarda za softversko inženjerstvo i kvalitetu proizvoda te uključuju iskoristivost, održivost i mogućnost održavanja.

Procjena uključuje provjeru da li je softver (a i projekt koji se razvija) u skladu s različitim karakteristikama i pokazuje li različite kvalitete koje su važne za njegovu održivost. Što je više obilježja zadovoljno, veća je održivost softvera, no potrebno je imati na umu da nisu sve osobine jednake težine (Jackson et al., 2011).

U obavljanju procjena, možda će biti potrebno razmisliti o tome kako različite korisničke klase utječu na značaj kriterija. Na primjer, za upotrebljivost i razumljivost, dokumentacije koja može biti konačna za korisnike neće biti dovoljna za programere jer su programerske dokumentacije opsežnije od korisničkih.

Bodovanjem se također može utjecati na prirodu samog softvera npr. ako se može zamisliti program koji dobro osmišljen, nudi pomoć ovisnu o kontekstu i sl., a istovremeno je jednostavan za korištenje da dodatne upute nisu potrebne (Jackson et al., 2011.). Prenosivost može primijeniti i na softver i na njegovu razvojnu infrastrukturu (npr. softver otvorenog koda može biti izgrađen, sastavljen i testiran na Unixu, Windowsu ili Linuxu, te je on lako „prenosiv" za korisnike i programere).

Kriteriji ocjenjivanja su grupirani na slijedeći način (tablica 2.).

Tablica 2. Grupiranje kriterija ocjenjivanja (Jackson et al., 2011, str. 1-2)

Kriterij	**(Pod)kriterij**	**Bilješke – do koje mjere je? / ima li softver?…**
Uporabljivost	Razumljivost	Je li lako razumljiv?
	Dokumentacija	Sveobuhvatna, prikladna, dobro strukturirana korisnička dokumentacija?
	Izgrađenost	Jednostavan za izgradnju na podržanom sustavu?
	Instalacija	Jednostavan za instalaciju na podržanom sustavu?
	Učenje	Jednostavno za naučiti kako koristiti njegove funkcije?
Održivost i mogućnost održavanja	Identitet	Identitet projekta/programa je jasan i jedinstven?
	Autorsko pravo	Lako se vidi tko je vlasnik projekta/programa?
	Licenciranje	Usvajanje odgovarajuće licence?
	Upravljanje	Lako je shvatiti kako se projekt izvodi i kako je njegov razvoj postignut?
	Zajednica	Dokaz o trenutnoj/budućoj zajednici u kojoj se softver nalazi?

	Pristupačnost	Dokaz o trenutnoj/budućoj sposobnosti za preuzimanje?
	Testiranje	Lako se testira ispravnost izvornog koda?
	Prenosivost	Iskoristiv na različitim platformama?
	Podrška	Dokaz o trenutnoj/budućoj razvojnoj podršci?
	Analiziranje	Lako je razumjeti izvornu razinu softvera?
	Promjenjivost	Jednostavan programerima za izmjenu i promjene?
	Evolucija	Dokaz o trenutnom/ budućem razvoju?
	Interoperabilnost	Zajednički rad sa drugim potrebnim/povezanim softverom?

Slijedi opis tablice 2. Grupiranje prema upotrebljivosti se dijeli na sljedeće dijelove (Jackson et al., 2011.):

- Razumljivost predstavlja osnovu softverske upotrebljivosti i njenim vrednovanjem se dobivaju odgovori na pitanja kao što su: Koliko je softver jednostavan i razumljiv? Što je softver radi i koja je njegova svrha? Koje je željeno tržište i tko su korisnici softvera? Koje su osnovne funkcije softvera? Koje su napredne funkcije softvera?
- Dokumentacija pruža pregled i opis softvera na visokoj razini. Ona se često dijeli u sekcije za korisnike, korisnike-programere i programere (ovisno o softveru). Dokumentacija odgovara na pitanja: Kvalitete? Potpunosti? Točnosti? Prikladnosti? Jasnoće? Dokumentacija navodi izvore za detaljne informacije i dodatne izvore. Uz softver najčešće dolazi u „Read Me“ obliku i sadrži informacije o verziji softvera.
- Izgrađenost postavlja pitanja vezana za spremnost softvera da bude izgrađen na nekoj platformi kao što su: Upoznavanje i koji su preduvjeti za izgradnju softvera na platformi? Kako izgraditi softver na platformi? Ukoliko se softver preuzima sa web stranice, ona bi trebala sadržavati upute za njegovu izgradnju. Također, ako se softver nabavlja od dobavljača, oni bi trebali dostaviti upute za izgradnju zajedno sa softverom.
- Instalacija softvera je korak koji se teško izbjegava. Naravno, prije same instalacije obavljaju se provjere koje odgovaraju na pitanja: Postoje li preduvjeti za njegovu instalaciju? Koja je ciljana platforma? Postoji li posebna

konfiguracija? Instalacija se ne mora uvijek odnositi na instaliranje nego može biti orijentirana i na izgradnju softvera kroz različite nadogradnje. Prilikom vrednovanja kriterija instalacije, obuhvaćaju se sve radnje koje korisnik mora obaviti da bi instalirao neki softver, ali također i radnje koje uključuju njegovo uklanjanje.

- Učenje se vrednuje na način da se postavlja pitanje kako naučiti funkcionalne zadatke softvera? Učenje se bazira na vrednovanje priručnika koji sadrži upute za početak rada sa softverom i njegovo korištenje. Ponekada se te upute znaju povezati zajedno sa dokumentacijom softvera, ali u većini slučajeva one dolaze odvojene.

Grupiranje prema održivosti i mogućnosti održavanja se dijeli na sljedeće dijelove (Jackson et al., 2011.):

- Identitet: određuje do koje je mjere identitet softvera jasan i jedinstven,
- Autorsko pravo: određuje tko posjeduje određeni softver te obično svaki softver dolazi sa izjavom o autorskom pravu koja se najčešće uključuje u čarobnjak za instalaciju ili u poratnu datoteku,
- Licenciranje: određuje posjeduje li softver odgovarajuće dozvole,
- Upravljanje: odnosi se na utvrđivanje tko je odgovoran za menadžment softvera,
- Zajednica: sagledava se koja će korisnička skupina postojati za proizvod i kod vrednovanja se gledaju čimbenici kao što su korisnički komentari,
- Pristupačnost: određuje do koje je mjere proizvod dostupan te kome je dostupan,
- Testiranje: najčešće se odnosi na sposobnost softvera da izvršava neke radnje i testovima se utvrđuje postoje li prepreke u tome,
- Prenosivost: jedan od glavnih atributa danas je prenosivost te se vrednovanje najviše bazira na mogućnost uporabe u različitim okružjima,
- Podrška: uključuje procjenjivanje hoće li i u budućnosti postojati podrška za određeni softver i taj se atribut usko veže na evoluciju koja pokriva područje budućih ažuriranja ili novijih verzija,

- Promjenjivost: uključuje sposobnost softvera da prihvaća izmjene napravljene na njemu kao što su: nova funkcionalnost, modifikacija trenutnih funkcionalnosti, itd., te sam taj proces također uključuje analiziranje,
- Interoperabilnost: utvrđuje do koje mjere se softver pridržava standarda.

4.3. Nadzor kvalitete softvera u velikim razmjerima

Ideja nadzora i vrednovanja softvera postaje zamijećena u 2004. godini kada Grupa za poboljšanje softvera (SIG) uvodi novu uslugu za nadzor ili monitoring. Tijekom posljednjih 10 godina, niz tehnoloških, organizacijskih i infrastrukturnih inovacija omogućile su SIG-ov rast i on postaje specijalist za procjenu kvalitete softvera koja pomaže širokom krugu organizacija kako bi lakše upravljale svojim aplikacijama, razvojnim projektima, ali i dobavljačima. SIG se je razvio do te mjere da pruža uslugu procjene koja se trenutno mjeri obradom 27 milijuna linija koda svaki tjedan (SIG, 2017).

4.3.1. Alati za analizu

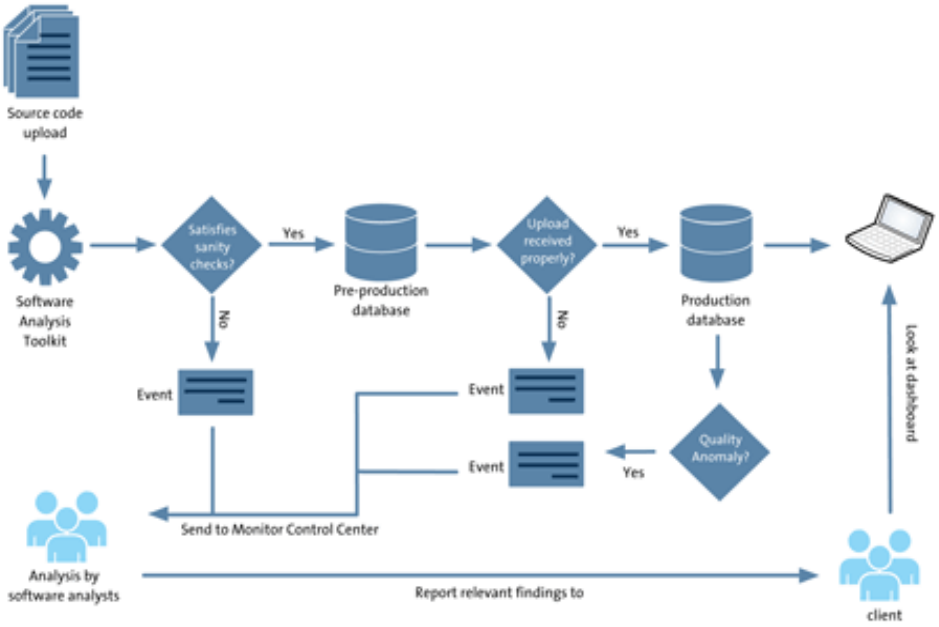
Softverski analitičari SIG-a su također i softverski programeri koji stvaraju i kontinuirano poboljšavaju svoje pakete softverskih alata za analizu. Ne samo da su ti alati vješti u prepoznavanju izvornog koda, već se mogu lako proširiti na podršku i niz dodatnih računalnih jezika. Do danas, ta snaga je iskorištavana za razvijanje podrške za oko 100 različitih jezika, od kojih se njih 50 koristi na kontinuiranoj osnovi. Ovi alati su također dobri u autonomnom radu i načinu skaliranja. Nakon početne konfiguracije, nove serije izvornog koda mogu se automatski i brzo analizirati, dopuštajući analitičarima da usmjere svoju pozornost na kvalitetne anomalije koje se mogu pojaviti. U prosjeku, te anomalije se javljaju u oko 15% analiziranog koda (SIG, 2017).

4.3.2. Model vrednovanja

Dok su svi softverski sustavi razlikuju (npr. njihova dob, korištene tehnologije, funkcionalnosti i arhitektura), postoje zajednički obrasci između njih. Oni postaju očiti kroz opsežne analize. SIG-ovi analitičari primijetili su te zajedničke obrasce u mjerenjima kvalitete softvera i udružili svoje iskustvo da proizvedu standardizirane modele vrednovanja. Ti modeli se odnose na različite aspekte kvalitete softvera koji su definirani standardom ISO-25001. Prvi model iz 2007. godine je bio usredotočen na "održivost" sustava. Od 2009. godine, ovaj model je korišten od strane tvrtke Technischer Überwachungs-Verein (TÜV) i služio za potvrdu softverskih proizvoda. Daljnjim razvojem pojavljuju se slični modeli za softversku sigurnost, pouzdanost, performanse, testiranje i energetsku učinkovitost (SIG, 2016).

4.3.3. Lab organizacija

Skalabilni alati i modela koji se mogu učinkovito primijeniti postaju izuzetno vrijedni, ali da se to postigne njihova pravilna organizacija mora biti na prvome mjestu. SIG organizira svoj softver za analizu aktivnosti u ISO-17025 ovlaštenom laboratoriju. To znači da analitičari moraju proći odgovarajuću obuku i slijediti standardizirane postupke za rad kako bi dobili mjerne rezultate koji se mogu ponoviti. Kada otkriju anomalije kvalitete, stručnjaci prolaze model za tijek rada baziran na snimci sustava (slika 2.) u kojemu otklanjaju negativne rezultate. Zatim, ovisno o težini ili vrsti nalaza, analitičar radi s klijentom kako bi utvrdio odgovarajuću rezoluciju. Ako se anomalija ne može riješiti, viši menadžment se uključuje u cijeli proces. Trenutno SIG prati više od 500 programskih sustava te preuzima 200 snimki sustava svaki tjedan. Od tih, njihovi analitičari su odgovorni za procjenu linija koda u više od 50 različitih jezika od COBOL-a i Scala do PL/SQL-a i Pythona (M2 Scientifics LLC, 2017).



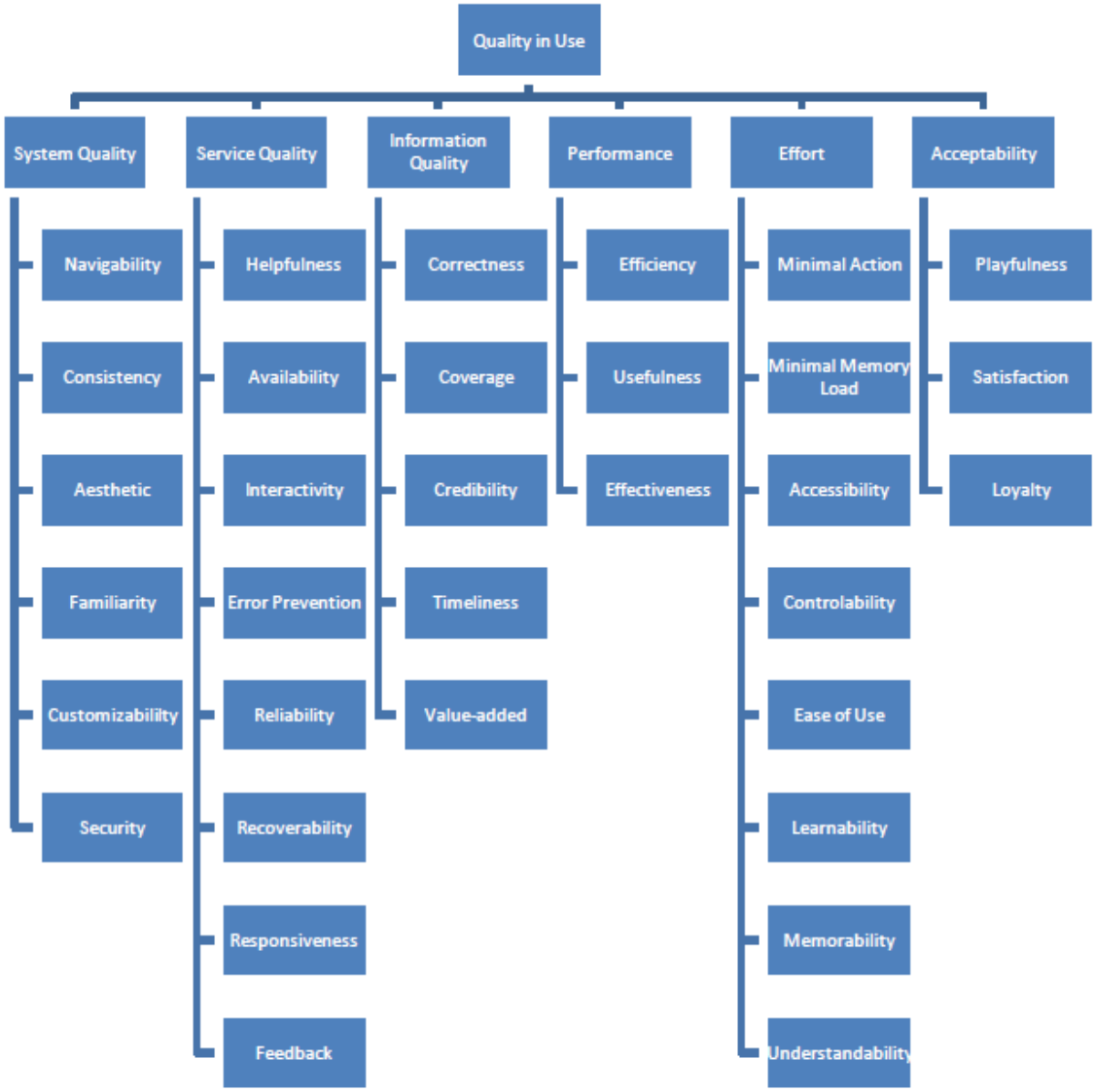
Slika 2. Model za tijek rada temeljen na snimci sustava (ECRIM News, 2017)

5. WEB KVALITETA

Kada se govori o softveru i njegovoj kvaliteti, potrebno se je osvrnuti i na sam web. Danas se na webu može naći praktički sve, a web stranice su glavni posrednici između korisnika i njihovih ciljeva. Upravo zbog sve veće koncentracije prema webu, postalo je vrlo važno kakva je kvaliteta stranica kojima korisnici pristupaju. Stoga se razvojem weba počelo i razvijati njegovo vrednovanje koje je postalo dio cjelokupnog vrednovanja kvalitete nekog sustava.

Vrednovanje web aplikacija može se podijeliti na šest atributa koji su prikazani slikom 3. (Orehovački, 2010):

- Kvaliteta sustava: uključuje atribute kvalitete na razini funkcionalnosti i fokusira se na atribute navigacije, konzistentnosti, estetike, upoznatosti, prilagodljivosti i sigurnosti.
- Kvaliteta usluge: uključuje atribute interakcije između web aplikacija i korisnika, a odnosi se na atribute korisnosti, dostupnosti, interaktivnosti, prevencije pogrešaka, pouzdanosti, mogućnosti oporavka, itd.
- Kvaliteta informacija: može se promatrati sa dva stajališta. Prvo, sa aspekta kvalitete informacija koje nastaju od korištenja web aplikacija i drugo, s aspekta kvalitete informacija sadržaja koji je dostupan na web aplikaciji. Prilikom toga su važni atributi ispravnosti, pokrivenosti, vjerodostojnosti, bezvremenosti i vrijednosti koja omogućava da je dostupan sadržaj ažuriran.
- Izvođenje: odnosi se na kvalitetu izvršavanja aktivnosti s web aplikacijom, a odnosi se na atribute učinkovitosti, korisnosti i efikasnosti.
- Napor: predstavlja očekivanja od korištenja web aplikacije, a uključuje atribute minimalnog kapaciteta memorije, minimalne akcije, pristupačnosti, kontrole, jednostavnosti korištenja, učenja, pamćenja i razumijevanja.
- Prihvatljivost: odnosi se na atribute koji doprinose uspjehu web aplikacije. Razigranost, zadovoljstvo i odanost su atributi koji najčešće određuju prihvatljivost.



Slika 3. Grupacija atributa kvalitete i njihovi pod-atributi (Orehovački et al., 2011)

Vrednovanje kvalitete počinje već od samog razvoja web mjesta. U tome se najveću pozornost pridaje dizajnu i njegovoj orijentaciji na korisnike. Kao i kod razvoja softvera, važno je odrediti ciljanu publiku, karakteristike, te u kojim uvjetima će korisnici pristupati nekom web mjestu. Kako bi se izradile upotrebljive web stranice, pažnja se vraća na dizajn, te se nastoji doći do onog dizajna koji će povećati samu upotrebljivost. Postizanje takvog dizajna je moguće uz praćenje određenih smjernica, među kojima su poznatije Nielsen Normanove smjernice koje su dostupne na www.nngroup.com, zatim smjernice američke javne uprave www.usability.gov, kao i druge smjernice o dizajnu web mjesta koje su nastale prema Patrick J. Lynchu i Sarah Horton. Sve te

smjernice se oslanjaju na sljedeće elemente dizajna web mjesta (Nielsen Norman Group, 2017): strukturu web mjesta, navigaciju i pretraživanje, dizajn informacija, strukturu web stranice (najviše početne), grafički dizajn, tipografiju, korištenje grafike i multimedijskih sadržaja, stil pisanja, dostupnost, korisničko iskustvo, itd.

Važnost pridržavanja tih smjernica se može pronaći u raznim situacijama ponašanja korisnika. Na primjer, ako korisnik ima poteškoća tijekom korištenja web stranice te ne može pronaći potrebni sadržaj ili se ne može snaći u navigaciji, tada će korisnik potražiti drugu web stranicu na kojoj će ostvariti svoje ciljeve. Iako je izrada web stranica i njihov dizajn napredovao, dizajneri još uvijek rade greške koji loše utječu na upotrebljivost (Plantak Vukovac i Orehovački, 2010).

Kako bi se stvorilo upotrebljivo web mjesto, provode se vrednovanja upotrebljivosti koja se mogu svrstati u 3 kategorije: metode pregledavanja (eng. inspection methods), metode testiranja (eng. testing methods) i metode ispitivanja (eng. inquiry methods).

5.1. Metode pregledavanja

U metode pregledavanja se uvrštavaju metode koje procjenjuju usklađenost web aplikacije sa standardima i smjernicama upotrebljivosti (eng. standards inspection i guideline review). Krajnja ocjena web aplikacije ovisi o osobnom sudu pojedinca ili skupine ljudi koji su proveli vrednovanje. U važnije metode pregledavanja se uvrštavaju dalje objašnjene metode (Plantak Vukovac i Orehovački, 2010).

5.1.1. Heurističko vrednovanje

Heurističko vrednovanje (eng. heuristic evaluation) ili revizija upotrebljivosti je procjena sučelja od strane jednog ili više stručnjaka. Ocjenjivači mjere upotrebljivost, učinkovitost i učinkovitost sučelja temeljenog na 10 heuristika upotrebljivosti koju je prvobitno odredio Jakob Nielsen (1994). Nielsenova heuristika upotrebljivosti uključuje: pružanje korisnicima pravovremene i odgovarajuće povratne informacije o statusu sustava, usporedbu između sustava i stvarnog svijeta, kontrolu korisnika i

slobodu gdje bi korisnici trebali osjetiti kontrolu dok komuniciraju sa sustavom, te prevenciju pogrešaka (Usabilityfirst, 2017).

5.1.2. Kognitivna šetnja

Kako bi odredili razinu upotrebljivosti web stranice, jedan ili više stručnjaka provodi kognitivnu šetnju (eng. cognitive walkthrough) kroz skup najobičnijih korisničkih zadataka koje podržava web stranica, korak po korak. U svakom koraku u postupku zadatka, procjenitelj (i) se pita za sljedeća četiri pitanja o njezinim očekivanjima o ponašanju korisnika: Hoće li korisnik pokušati postići pravi učinak? Hoće li korisnik primijetiti da je ispravna radnja dostupna? Hoće li korisnik pridružiti ispravnu radnju s učinkom koji će se postići? Ako se izvrši ispravna radnja, hoće li korisnik vidjeti da je napredak u rješavanju zadatka? Vrednovatelj na taj način pokušava doći do "priče o uspjehu" za svaki korak u tom procesu. Ako ne uspije pronaći jednu, umjesto toga stvara "priču o neuspjehu" i procjenjuje zašto korisnik ne bi mogao izvršiti zadatak na temelju sučelja. Ti se uvidi upotrebljavaju za poboljšanje upotrebljivosti web stranice ili aplikacije (Usability, 2017).

5.1.3. Multidisciplinarna kognitivna šetnja

Jedna od verzija kognitivne šetnje je multidisciplinarna kognitivna šetnja (eng. pluralistic walkthrough) u kojoj krajnji korisnik, programer i stručnjak zajedno prolaze kroz cijelu web aplikaciju te pritom komentiraju svaki element sučelja. Prednost metode je djelotvorno uočavanje problema koji mogu proizaći iz pristupa sa aplikacijom. Nedostatak metode je pretjerana detaljnost u provođenju, a samim time i sporost (Plantak Vukovac i Orehovački, 2010).

5.1.4. Pregled funkcionalnih dijelova

Pregled funkcionalnih dijelova je (eng. feature inspection) tehnika inspekcije upotrebljivosti koja identificira zadatke koje će korisnik izvršiti s aplikacijom i značajke

aplikacije koja će se koristiti za obavljanje tih zadataka. Svaka značajka se zatim procjenjuje je li razumljiva, korisna i zapravo dostupna korisniku kada je to potrebno. Ovaj pristup naglašava važnost funkcionalnosti za postizanje upotrebljivosti (Usabilityfirst, 2017).

5.2. Metode testiranja

Metodama testiranja se dolazi do informacija o tome kako korisnici upotrebljavaju web stranice i aplikacije, a provode se kako bi utvrdili probleme koji se mogu pojaviti u procesu korištenja, ali i kako bi se utvrdila prikladnost web aplikacije za određenu skupinu zadataka. Tijek testiranja se provodi na način da korisnici izvršavaju određeni zadatak, a ocjenjivači prate njihov rad te bilježe dobivene rezultate. Neke od važnijih metoda testiranja su objašnjene u nastavku (Usabilityfirst, 2017).

5.2.1. Metoda praćenja oka

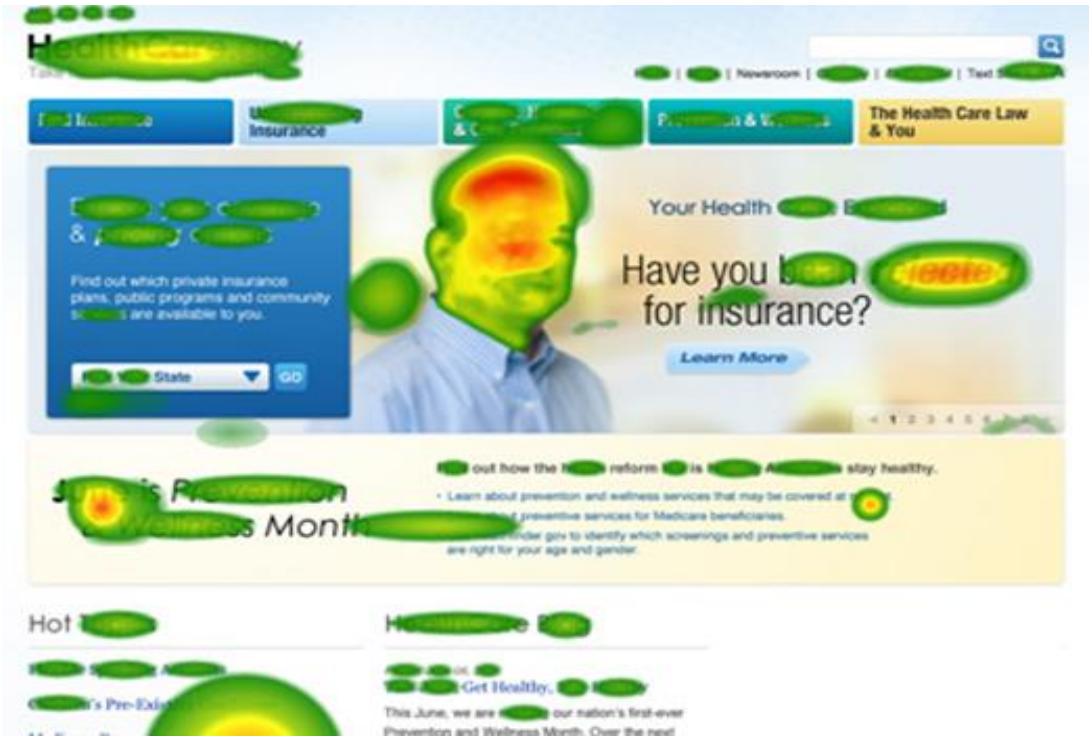
Praćenje oka (eng. eye tracking) uključuje mjerenje ili gdje je oko usmjereno ili gibanje oka kao pojedinac gleda na web stranicu. Svrha praćenja očiju je važna prilikom izrade web stranice ili web aplikacije.

Kada se metoda provede, iz rezultata se može reći: gdje su posjetitelji gledali, koliko dugo su nešto tražili, na koji se način njihovi fokusi kreću od stavke do stavke na web stranici, koji su dijelovi sučelja koje propuštaju, način na koji se kreću dužine stranice, kako veličina i položaj stavki na vašoj postojećoj web-lokaciji ili na predloženim projektima utječe na njihovu pozornost, itd (Usability, 2017).

Metoda radi na način da kada sudionik pregleda web stranicu, uređaj za praćenje očiju usredotočuje se na učinak oka sudionika i određuje smjer i koncentraciju njihova pogleda. Softver generira podatke o tim akcijama u obliku toplinskih karata i putova „saccade“ (Usability, 2017).

5.2.1.1. Toplinske mape

Toplinske karte predstavljaju gdje se posjetitelj usredotočio na pogled i koliko dugo su gledali na određenu točku. Općenito, ljestvica boja koja se kreće od plave do crvene označava trajanje fokusa. Dakle, crvena točka na području vaše stranice može ukazivati na to da je sudionik ili grupa sudionika usmjerena na ovaj dio stranice na duže vremensko razdoblje (Usability, 2017). Primjer toplinske mape je prikazan na slici 4.



Slika 4. Toplinska mapa (Usability, 2017)

5.2.1.2. „Saccade" putanje

Saccade putanja prati kretanje oka između područja fokusiranja. Putanja kreće od prvog mjesta koje oko zapazi prilikom ulaza na web stranicu i kreće se onako kako oko prelazi po sadržajima web stranica a prilikom toga se gleda razdoblje pozornosti, a zatim brzo kretanje. Crveni krug je područje fokusa, dok crvena crta označava put (Salvucci i Goldberg, 2000). Primjer takve putanje je prikazan na slici 5.



Slika 5. Primjer Saccade putanje (Usability, 2017)

5.2.1.3. Prednosti i nedostaci metode praćenja oka

Prilikom odabira metode praćenja oka, važne su i sljedeće prednosti (Usability, 2017):

- Može se razaznati da li korisnici čitaju ili pretražuju sadržaj.
- Metoda kao rezultat daje relativni intenzitet pozornosti korisnika na različite dijelove web stranice.
- Metodom se može odrediti hoće li korisnik tražiti nešto. To je posebno vidljivo kada korisnici nisu sigurni koje riječi traže. To se može dogoditi kada se korisnici nalaze na novim web stranicama ili stranicama s grupama širokih kategorija.
- Metoda omogućuje da se usporede obrasci skeniranja različitih korisničkih skupina.

Sa druge strane, prilikom odabira metode praćenja oka treba obratiti pozornosti i na njene nedostatke (Usability, 2017):

- Ne može se sa sigurnošću da su korisnici vidjeli nešto savjesno. Korisnici mogu kratko vrijeme usmjeriti svoje oči na neko područje bez svijesti.
- Ne može se reći sa sigurnošću da korisnici nisu vidjeli nešto jer praćenje oka ne obuhvaća periferni vid.
- Ne može se reći zašto korisnici gledaju nešto.
- Ne može se sve učinkovito testirati. Problemi su mogu dogoditi kada neki korisnici nose naočale ili tvrde kontaktne leće ili ako lutaju očima ili imaju bezizražajno lice.

5.2.2. Udaljeno testiranje

Daljinski testovi (eng. remote testing) upotrebljivosti omogućuju da se provedu korisnička istraživanja s sudionicima u njihovom prirodnom okruženju korištenjem softvera za dijeljenje zaslona ili mrežnih usluga korisnicima usluga daljinskog korištenja. Takvi testovi traju oko 15-30 minuta, a sastoje se od oko 3-5 zadataka.

Daljinsko ispitivanje se provodi kada postoje određeni uvjeti i kada ima smisla razmotriti daljinsko testiranje upotrebljivosti. Neki od tih uvjeta uključuju (Usability, 2017):

- Vremenske rokovi koji mogu spriječiti testiranje osoba zbog problema s raspoređivanjem
- Ciljana publika / sudionici geografski su rasprostranjeni, što može postavljati problem za njih ili eksperimente
- Sudionici trebaju koristiti određeni radni stroj zbog softverskih ili sigurnosnih zahtjeva
- Sudionici imaju prava pristupačnosti koja zahtijevaju korištenje vlastitog softvera ili opreme
- Potencijalno pokretanje više testova odjednom,
- Iako je ova metoda fleksibilna treba uzeti u obzir i troškove ove metodologije.

5.2.2.1. Prednosti i izazovi udaljenog testiranja

Prednosti udaljenog testiranja su (Usability, 2017):

- Uklanja se potreba za laboratorijskim okruženjem i njegov učinak na sudionike,
- Osposobljavaju se različite skupine sudionika,
- Općenito je jeftinije od tradicionalnih testiranja u laboratoriju,
- U slučaju neograničenog testiranja, omogućuje se produljenje testnog dana, što omogućava pristup većem broju sudionika,
- Predstavlja priliku da se test provede na većoj skupini ljudi nego u laboratorijskom okruženju.

Izazovi na koje udaljeno testiranje može naići su (Usability, 2017):

- Sigurnost može biti ugrožena ako se testiraju osjetljiva, povlaštena ili intelektualna svojstva,
- Ograničeni ili nikakvi pogledi na korisnikov govor tijela možda bi spriječili neke od njihovih reakcija na materijal koji se testira,
- Tehničke poteškoće su vjerojatne ako korisnici nisu zadovoljni s uključenom tehnologijom, imaju softver ili opremu koji ometa funkciju testiranog softvera, nisu u mogućnosti dijeliti svoj zaslon preko interneta, imaju nepouzdane ili spore brzine veze ili ako test zahtijeva posebnu opremu ili preuzimanja softvera ili dodatke, sudionici možda neće voljni ili ga ne mogu preuzeti.

5.2.3. Razmišljanje naglas

Razmišljanje naglas (eng. thinking-aloud protocol) je tehnika testiranja gdje se od korisnika traži da govore svoje misli dok obavljaju zadatak. Dok je fokus na testiranju korisnika prvenstveno na tome koliko učinkovito korisnik obavlja potrebne zadatke (a ne o tome kako korisnici vjeruju da su oni uspješni), verbalizacije su vrlo korisne u razumijevanju pogrešaka koje su napravljene i dobivanja ideja o tome što bi uzroci mogli biti i kako sučelje se može poboljšati kako bi se izbjegli ti problemi (Plantak Vukovac i Orehovački, 2010).

5.2.4. Postavljanje pitanja

Za razliku od protokola za razmišljanje naglas, u kojemu se od korisnika traži da slobodno govore o svojim mislima, ocjenjivač eksplicitno postavlja pitanja korisniku (eng. question asking protocol) tijekom testiranja kako bi vidio kako se zasloni tumače se i zamolite korisnika da reagira na dijelove dizajna koji korisnik inače zanemaruje (Plantak Vukovac i Orehovački, 2010).

5.2.5. Automatsko zapisivanje postupaka

Prijava uključuje automatsko sakupljanje podataka (eng. logging actual use) o detaljnoj upotrebi sustava. Korisno je jer pokazuje kako korisnici obavljaju svoj stvarni posao i zato što je jednostavno automatsko prikupljanje podataka od velikog broja korisnika koji rade pod različitim okolnostima. Uobičajeno, dnevnik sučelja sadržavat će statistiku o učestalosti kojom svaki korisnik koristi svaku značajku u programu i učestalost kojom su se dogodili razni događaji od interesa (kao što su poruke o pogreškama). Statistike o učestalosti korištenja naredbi i drugih značajki sustava mogu se koristiti za optimizaciju često korištenih značajki i za prepoznavanje značajki koje se rijetko koriste ili se ne koriste. Statistika o učestalosti različitih situacija u pogrešci i upotrebi internetske pomoći može se koristiti za poboljšanje upotrebljivosti budućih izdanja sustava redizajniranjem značajki koje uzrokuju najviše pogrešaka i većine pristupa za on-line pomoć. Ova se tehnika može koristiti pri testiranju ili fazi razvoja softvera (Usabilityhome, 2016).

Od ostalih metoda testiranja, najčešće su (Plantak Vukovac i Orehovački, 2010):

- Izravan prijenos (eng. shadowing method)
- Stručno podučavanje (eng. coaching method)
- Učenje (eng. teaching method)
- Konstruktivna interakcija (eng. codiscovery learning)
- Odgođeno emitiranje (eng. retrospective testing)
- Analiza dnevnika (eng. log file analysis)

5.3. Metode ispitivanja

Metoda ispitivanja (eng. inquiry methods) koriste se pri vrednovanju web aplikacije na kraju njenog razvojnog ciklusa i pokazuju cjelokupno zadovoljstvo tom aplikacijom. Slijede neke od najvažnijih metoda (Plantak Vukovac i Orehovački, 2010).

5.3.1. Upitnici

Upitnici (eng. questionnaires) se sastoje od unaprijed definiranih pitanja i skupa otvorenih ili zatvorenih odgovora. Upitnici se temelje na prikupljanje mišljenja korisnika. Kao metodu, moguće ih je primijeniti u svim razvojnim fazama, ali najčešće se koriste nakon što se korisnik prvi puta sretne sa web aplikacijom. Upitnike se može razvrstati na one koji su namijenjeni za procjenu upotrebljivosti programskih proizvoda i na one koji služe za procjenu zadovoljstva korisnika i upotrebljivost web stranica. Sljedeće dvije metode su odabrane za svaku kategoriju: SUS i WUS (Usability, 2017).

5.3.1.1. SUS

Skala upotrebljivosti sustava (eng. SUS – System Usability Scale) osigurava brzo i pouzdano sredstvo za mjerenje upotrebljivosti. Sastoji se od 10 pitanja, svaki sa pet opcija odgovora: od „Čvrsto se slažem“ pa do „U potpunosti se ne slažem“. Izvorno ga je stvorio John Brooke 1986. g, a omogućuje procjenu široke palete proizvoda i usluga, uključujući hardver, softver, mobilne uređaje, web stranice i aplikacije.

SUS je postao industrijski standard, s referencama u više od 1300 članaka i publikacija. Istaknute koristi od korištenja SUS-a uključuju sljedeće (Usability, 2017):

- Jako je lagan za administraciju sudionicima,
- Može se koristiti na malim uzorcima s pouzdanim rezultatima,
- Može učinkovito razlikovati korisne i neupotrebljive sustave.

Kada se koristi SUS, od sudionika se traži da ocjenjuju sljedećih 10 stavki s jednim od pet odgovora koji se kreću od „Čvrsto se slažem“ do „U potpunosti se ne slažem“ (Usability, 2017):

1. Mislim da bih želio koristiti ovaj sustav često,
2. Otkrio sam da je sustav nepotrebno složen,
3. Mislim da je sustav jednostavan za korištenje,
4. Mislim da bih trebala pomoć tehničke osobe da bi mogla koristiti ovaj sustav,
5. Otkrio sam da su razne funkcije u ovom sustavu dobro integrirane,
6. Mislim da postoji previše nedosljednosti u ovom sustavu,
7. Smatram da će većina ljudi naučiti koristiti ovaj sustav vrlo brzo,
8. Otkrio sam da je sustav vrlo težak za upotrebu,
9. Osjećala sam se vrlo sigurno u korištenju sustava,
10. Morao sam naučiti mnogo stvari prije nego što sam mogao raditi s ovim sustavom.

Tumačenje bodovanja SUS-a može biti složeno. Bodovi sudionika za svako pitanje pretvaraju se u novi broj, dodaju se zajedno, a zatim pomnože sa 2,5 za pretvaranje izvornih rezultata od 0-40 na 0-100. Iako su rezultati 0-100, to nisu postotci i trebali bi se uzeti u obzir samo u smislu njihovog prosječnog rangiranja. Na osnovi istraživanja, ocjena SUS iznad 68 bila bi iznad prosjeka, a sve ispod 68 je ispod prosjeka, no najbolji način tumačenja rezultata uključuje "normaliziranje" rezultata kako bi se postigao rang prosjeka (Usability, 2017).

5.3.1.2. WUS

Web upotrebljivost sustava (eng. WUS – Web Usability Scale) je Likertova ljestvica temeljena na upitniku za ocjenu upotrebljivosti web stranica, a temelji na SUS-u. Za svaki test web stranica, korisnici ispunjavaju kratki upitnik i na temelju odgovora se izračunava WUS rezultat (Whatusersdo, 2017).

Kao i kod SUS-a, rezultat se kreće u rasponu 0-100, gdje rezultat 100 označava najbolju upotrebljivost. Neke studije su pokazale da prosječna ocjena web stranica iznosi 72 na WUS skali. Kao i kod SUS-a, prilikom uporabe ove metode korisnici dobiju 10 pitanja na koja odgovaraju (Whatusersdo, 2017):

1. Mislim da bih često koristio ovu web stranicu,
2. Našao sam nepotrebno složenu web stranicu,
3. Mislim da je web stranica jednostavna za korištenje,

4. Mislim da bih trebao pomoć nekog tehničara kako bi mogao koristiti ovu web stranicu,
5. Otkrio sam da su različite funkcije na web stranici dobro integrirane,
6. Mislim da postoji previše nedosljednosti na ovoj web stranici,
7. Smatram da će većina ljudi vrlo brzo naučiti koristiti ovu web stranicu,
8. Otkrio sam da je web stranicu vrlo neugodno koristiti,
9. Osjećala sam se vrlo sigurno upotrebom web stranice,
10. Trebao sam naučiti mnogo stvari prije nego što sam se mogao snaći s ovom web stranicom.

5.3.2.Kontekstualno ispitivanje

Kontekstualno ispitivanje (eng. contextual inquiry) je metoda za dobivanje informacija o kontekstu upotrebe, pri čemu ocjenjivači prvo postavljaju korisnicima skup standardnih pitanja, a zatim ih promatraju i propituju dok rade u vlastitom okruženju. Budući da se korisnici intervjuiraju u vlastitom okruženju, podaci analize su realniji. Ova se tehnika općenito koristi na početku procesa dizajna i dobra je za dobivanje informacija o radnoj praksi, društvenim, tehničkim i fizičkim okruženjima i korisničkim alatima.

Četiri načela kontekstualnog istraživanja su (Usability Body of Knowledge, 2017):

- Fokus: Plan za upit, zasnovan na jasnom razumijevanju njegove svrhe,
- Kontekst: Ocjenjivači idu na radno mjesto klijenta i gledaju ih dok rade svoj posao,
- Partnerstvo: Razgovor s klijentima o radu i njihovo uključivanje u otkrivanje nerazriješenih aspekata rada,
- Tumačenje: Razvijanje zajedničkog razumijevanja s klijentom o aspektima posla koji su važni.

Rezultati kontekstualnog istraživanja mogu se upotrijebiti za definiranje zahtjeva, poboljšanje procesa, saznanje što je korisnicima i korisnicima važno i samo saznanje više o novoj domeni za informiranje budućih projekata.

5.3.3. Ciljane skupine

Ciljane skupine (eng. focus groups) su moderirane rasprave koje obično uključuju 5 do 10 sudionika. Pomoću ciljane skupine moći ćete saznati o stavovima korisnika, uvjerenjima, željama i reakcijama na koncepte. Ciljane skupine su tradicionalne tehnike istraživanja tržišta. U tipičnoj ciljanoj skupini sudionici razgovaraju. U tipičnom provođenju testa upotrebljivosti ili kontekstualnom ispitivanju, korisnici djeluju, a kao rezultat njihovog promatranja se mogu izvući zaključci. Pri određivanju ciljane skupine, istraživači obično odabiru sudionike na temelju specifičnih osobina ili karakteristika, uključujući: dob, okupaciju sudionika, iskustvo, obrazovanje i etničku pripadnost (Usability, 2017).

5.3.4. Terensko ispitivanje

Terensko ispitivanje (eng. field observation) je metoda koja uključuje posjećivanje jednog ili više korisnika u njihovoj radnoj okolini te praćenje izvršavanja pojedinih zadataka putem određene web aplikacije. Kao i kod kontekstualnog ispitivanja, cilj metode je promatranje korisnika u njihovom prirodnom okružju i poželjno je da ocjenjivač prikuplja podatke neprimjetno kako ne bi ometao korisnike. Kako bi se to postiglo, ova metoda se često primjenjuje zajedno sa metodom odgođenog emitiranja (eng. retrospective testing) gdje se korisnika snima kamerom, a podaci prikupljaju analizom dobivenog video zapisa (Plantak Vukovac i Orehovački, 2010).

Od ostalih metoda testiranja, najčešće se koriste (Plantak Vukovac i Orehovački, 2010):

- Ankete (eng. surveys),
- Razgovori (eng. interviews),
- Vođenje dnevnika (eng. self-reporting logs),
- Spremanje izgleda ekrana (eng. screen snapshots),
- Povratna informacija od korisnika (eng. user feedback).

6. KVALITETA MOBILNIH APLIKACIJA

Mobilni uređaji postali su prevladavajući standard za komunikaciju, za potrošače i za poslovanje širom svijeta. Danas mobilne aplikacije vode većinu osobnih i profesionalnih interakcija. Stoga se postavljaju pitanja: što određuje uvjerljivost mobilne aplikacije, a što tjera ljude da koriste baš određene aplikacije? Jedan od zajedničkih odgovora u razvoju aplikacija jest osigurati da mobilna aplikacija pokriva očekivanja klijenata i poslovne ciljeve putem strategije testiranja mobilnih aplikacija (Singh, 2015).

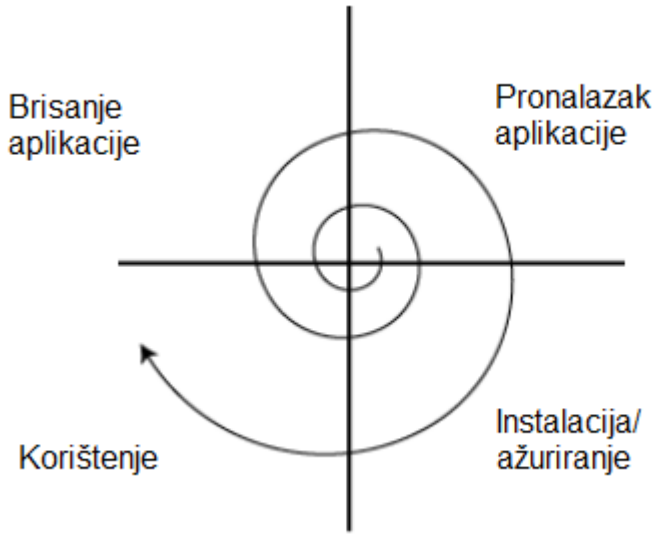
Kao i svako testiranje na radnoj površini ili web aplikacija, testiranje mobilnih aplikacija naglašava kvalitetu i performanse krajnjeg proizvoda. Ipak, to je teško, zbog ovih aspekata (Singh, 2015):

- Fragmentacija uređaja: Za razliku od konvencionalnih web ili desktop aplikacija, mobilne aplikacije upotrebljavaju se na više uređaja i platformi, kao što su iPhone uređaji, Androidi, Windows telefoni i drugi. Postoje brojne verzije operacijskih sustava za svaki uređaj. Razdvajanje mobilnog uređaja može biti problem za programere softvera koji stvaraju različite verzije iste aplikacije kako bi se osiguralo ispravno funkcioniranje s različitim verzijama zadanog operacijskog sustava. To je također izazov za odjele za provjeru kvalitete jer različite operacijske verzije imaju različite sposobnosti, što ih čini teže upravljati i osigurati (Moumane et al., 2016).
- Vanjsko protiv unutarnjeg testiranja: Zbog slabih proračuna i kratkih rokova, donositelji odluka često imaju dovoljno razloga za provođenje vanjskih aktivnosti testiranja. Za razliku od unutarnjeg testiranja, vanjskim će se smanjiti troškovi. Također će omogućiti usredotočenje na osnovne aktivnosti, posvetivši više vremena marketinškim kampanjama, korisničkim uslugama i tako dalje.
- Dostupnost mobilnih alata za testiranje: Kada tvrtka upotrebljava unutarnje testiranje, obično nedostaje alat za testiranje ili učinkovite metode. Postoji povećana razina sofisticiranosti potrebna za usklađenost s više uređaja. Tvrtke obično nemaju pristup najboljim praksama, smjernicama i industrijskim standardima testiranja za mobilne uređaje. Ovaj nedostatak dostupnosti je razlog zašto organizacije odlučuju dobavljati alate i talent preko partnera.

- Testiranje životnog ciklusa aplikacije: Metode testiranja mobilnih aplikacija zahtijevaju česte nadogradnje zbog sljedećeg: Krajnji korisnici očekuju približavanje bugova u stvarnom vremenu, redovne promjene ažuriraju programere kako bi zadržale kompatibilnost, agilne razvojne metodologije pružaju mnoge prednosti i ažuriranja značajki Ovi aspekti povećavaju raspon potreba testiranja mobilnih aplikacija, a testni ciklus raste za svaki uređaj, operativni sustav i nadogradnju softvera.

6.1. Model mobilne aplikacije

Životni vijek aplikacije se može podijeliti na 4 glavna dijela (Flood et al., 2012): pronalazak aplikacije, instalacija ili ažuriranje ako se aplikacija već koristi, korištenje aplikacije i brisanje aplikacije kada ona više korisniku nije potrebna ili ako je nezadovoljan aplikacijom. Prikaz tog životnog vijeka je ilustriran slikom 6.



Slika 6. Model životnog vijeka mobilne aplikacije (Flood et al., 2012, str. 6)

6.2. Karakteristike mobilne domene

Kao i kod programskih proizvoda te web orijentiranog softvera, kvaliteta mobilnih aplikacija se također najviše odnosi na njihovu upotrebljivost. Problemi koji se mogu pojaviti, a vezani su na upotrebljivost su sljedeći (Ljubić, 2011):

- Veličina zaslona: zadnjih nekoliko godina proizvođači mobilnih uređaja se služe logikom „veći zaslon, bolja upotrebljivost“. Naravno, u usporedbi sa stolnim računalima, zasloni mobilnih uređaja su mali i sami time prikaz ograničen. Stoga se pojavljuju sve napredniji zasloni raznih dimenzija putem kojih se želi omogućiti bolji prikaz, a i lakša interakcija sa aplikacijama. Dakle, aplikacije se dizajniraju na način da budu prilagodljive raznim dimenzijama zaslona.
- Veličina tipkovnice: budući da su dimenzije tipki (bilo one fizički dostupne ili virtualno prikazane na zaslonu) vrlo male, lako se mogu dogoditi pogreške tijekom unosa, što se najviše može odraziti prilikom uporabe aplikacija bankarstva ili neke druge aplikacije gdje je važna preciznost unosa.
- Razina procesorske snage i radne memorije: predstavlja čak veći problem od problema prikaza i unosa jer ako se aplikacija izradi na način da joj za pokretanje treba jači procesor ili veća količina radne memorije, tada će tu aplikaciju moći koristiti samo oni korisnici koji imaju zadovoljene te uvjete. Dakle, prilikom izrade aplikacije važno je usmjeriti se na njenu kompatibilnost kako bi se mogla pokrenuti na mobilnim uređajima sa raznim razinama dostupne radne memorije i sa različitim procesorima.
- Trajnost baterije: veće korištenje procesora i radne memorije jače crpi bateriju, te se programeri aplikacija trebaju osvrnuti na njihovu „težinu“ kako bi dobili aplikaciju koja će imati manji učinak na crpljenje baterije.
- Raznolikost tehnologije i standarda: Kao što je spomenuto i kod testiranja, postoje razne mobilne platforme. Kada je riječ o aplikacijama namijenjenima za Windows Phone ili Android, mnoge su napravljene na način da se mogu pokrenuti na obje platforme, ali kada je riječ o aplikacijama za iPhone, one su dostupne samo na Apple uređajima.

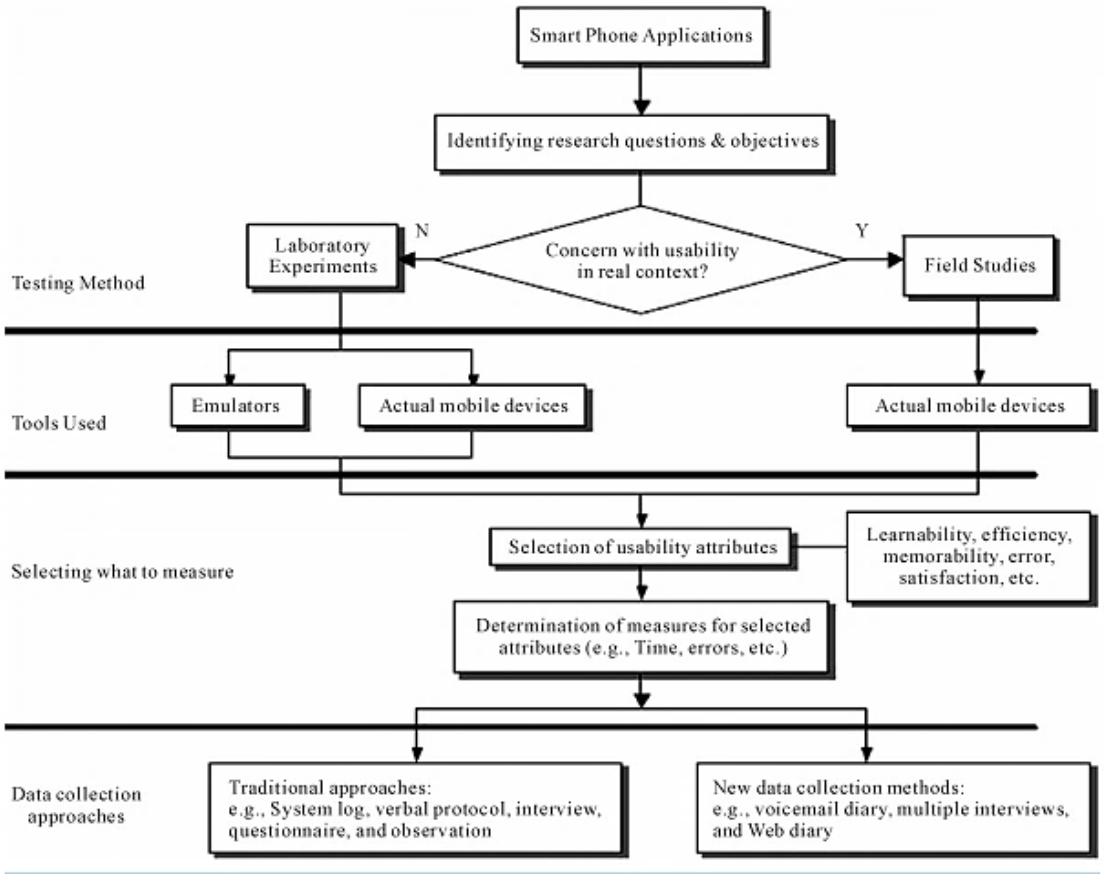
6.3. Vrednovanje mobilnih aplikacija

Ispitivanje upotrebljivosti pametnog telefona uglavnom se provodi pomoću testova upotrebljivosti alata u razvojnoj praksi. Za testiranje pametnog telefonskog programa pomoću alata za testiranje upotrebljivosti, testovi se obično provode uz pomoć protokola za razmišljanje (eng. thinking-aloud), koji se temelji na radu Ericssona i Simona. Kao i kod vrednovanja web-a, u ovom se protokolu određeni poslovi dodjeljuju korisnicima u ispitnoj domeni, a korisnici se potiču da glasno misle kad su pokušavajući izvršiti svoje specifične zadatke (Ahmad et al., 2014).

Tradicionalno, testovi upotrebljivosti za analizu pametnih telefonskih aplikacija provedeni su u ispitnim laboratorijima. Na primjer, laboratorij za testiranje upotrebljivosti sastojao bi se od zajedničkog dnevnog boravka ili ureda koji je priključen na područje praćenja putem jednostranog ogledala. Testiranje se provodi u mirnom okruženju u kojem se ispitivač upotrebljivosti može usredotočiti samo na njegov specifični zadatak. Nadalje, kamera u pametnom telefonu se brzo razvija tijekom posljednjih nekoliko godina, a mobilni korisnici vrše svoje korisničke testove za testiranje ove značajke u polju. Testiranje pametnih telefonskih aplikacija postaje vrlo jednostavno i usredotočeno. Korisnici pametnih telefona mogu bilježiti zaslon pametnih telefona sa priloženom mini kamerom i tako dobivaju informacije za kasniju evaluaciju. To omogućava korisnicima da provode testiranja i izvan laboratorijskih uvjeta (Ahmad et al., 2014).

U posljednjih nekoliko godina, pametni telefoni imaju koristi od implementacije uzastopnih značajki kao što su zaslon osjetljiv na dodir i sposobnosti obavljanja više zadataka odjednom (eng. multitasking). Mehanizam implementacije ostaje različit od mobilnog telefona do drugog mobitela, što se razlikuje po pristupu koji je uređen kako bi zadovoljio zahtjeve korisnika (Ahmad et al., 2014). Razvoj većine izvrsnih proizvoda, aplikacija, dizajna reinženjering pristupa i online trgovinama se kompetencije koje izgrađuju konkurenciju između različitih tvrtki pametnih telefona.

Slika 7. predstavlja okvir za testiranje pametnih telefona u laboratoriju i na terenu. Kao zamjena fizičkom uređaju, kod ispitivanja upotrebljivosti u laboratoriju se koriste i emulatori.



Slika 7. Okvir za testiranje mobilnih aplikacija (Ahmad et al., 2014, str. 1047)

Za evaluaciju odgovarajućih tehnologija provodi se testiranje upotrebljivosti i povezano je s razvojem svojstava sustava, značajkama i sučeljem sustava. U testiranju upotrebljivosti aplikacija koriste se jedinstvene značajke poput dodataka pametnog telefona, povezivanja i veličine zaslona pametnih telefona, različitih rezolucija zaslona i ograničene mogućnosti obrade i snage (Ahmad et al., 2014).

Vrlo je teško procijeniti dva različita najizraženija proizvoda na poslovnom tržištu. U opsegu ponude, jedan je proizvod bolji zbog izdašnosti, a drugi je bolji jer ispunjava zahtjeve korisnika. Kako bi zadovoljili zahtjeve korisnika, provode se ankete i provode se procjene upotrebljivosti od najpopularnijih pametnih telefona kao što su Android i Apple iPhone.

Testiranje upotrebljivosti raspravlja o različitim atributima iz perspektive korisnika. U početku se pretpostavlja metoda istraživanja upotrebljivosti jer pomaže u smanjenju troškova proizvoda i učinkovitom pristupu prikupljanju podataka od različitih korisnika dok su povezani s testom upotrebljivosti (Ahmad et al., 2014).

6.4. Bugfender

Bugfender daje priliku razvijanja kvalitetnih i stabilnih mobilnih aplikacija s lakoćom i učinkovitošću. Mnogo je lakše pronaći i riješiti izvor problema, te omogućuje učinkovitije obavljanje službe za korisnike jer se točno može vidjeti što se dogodilo kada se problem pojavio. Kada je Bugfender uključen u mobilnu aplikaciju, svi zapisi iz aplikacije korisnika bilježe se na konzoli. Dakle, u bilo kojem trenutku je moguće pristupiti tim zapisima da bi se vidjelo što se dogodilo, a sigurnost korisnika nikada nije ugrožena (Ventayol, 2016).

Bugfender je moćan alat kojim tijekom testiranja ne samo prijavljuje probleme, već pomaže da se oni brže pronađu. Sa Bugfenderom se može proći kroz svoje sesije i pogledajte gdje su se korisnici borili s problemima. To pomaže u otkrivanju grešaka u aplikaciji ili ako je problem u korisnikovom uređaju. Bugfender daje najsitnije informacije o izvedbi aplikacije i, još važnije, procjenu iskustva koje korisnici imaju s aplikacijom (Ventayol, 2016).

6.5. Robotium

Robotium je vodeći svjetski okvir otvorenog koda za automatizaciju testiranja na Androidu. Objavljen je u siječnju 2010 i ima punu podršku za izvorne i hibridne Android aplikacije. Robotium Recorder je alat kreiran od strane Robotium tima pomoću kojega programeri mogu testirati svoje aplikacije. Pomoću njega, testni slučajevi koji obično traju tjednima sada mogu se snimiti za nekoliko minuta. Robotium Recorder dostupan je kao dodatak za Android Studio i Eclipse. Paket ADT. Kod korištenja, uređaj koji se koristi za snimanje testnih slučajeva (bilo emulatora ili stvarnog uređaja) mora imati vanjsku pohranu (Robotium).

ZAKLJUČAK

Gotovo svaki dan se pojavljuje neki novi softver te se konstantno stvaraju različite vrste softvera za istu svrhu. Bilo da se radi o softverima koji se prodaju u trgovinama ili o softverima koji su dostupni na webu, vrlo je važno znati njihovu kvalitetu prije samog odabira. Upravo ta kvaliteta predstavlja izazov za programere i proizvođače jer ako softver nije kvalitetan, on neće biti nikome od koristi i samim time njegovo stvaranje neće biti profitabilno. Također, kvaliteta određenog softvera može doprinijeti rastu reputacije pojedinih tvrtki koje se bave njihovom proizvodnjom i na taj način one mogu privući korisnike da od njih nabavljaju i ostale proizvode. Također, kvalitetan softver će korisnicima omogućiti da obave ono što trebaju, dok će sa nekvalitetnim softverom ostati nezadovoljni. Upravo zbog toga je važno vrednovanje. Vrednovanjem se dobiva ocjena kvalitete koju će kasnije korisnici gledati pri odabiru odgovarajućeg softvera. Ukoliko je ocjena vrednovanja dobra, korisnici će lakše vjerovati takvom softveru za obavljanje zadataka, a ako je ocjena loša, tada će korisnici vjerojatno potražiti softver sa boljom ocjenom. Upravo zbog te krajnje ocjene su važne metode. Korištenjem metoda vrednovanja proizvođači i programeri mogu dobiti jasan uvid u probleme koji se mogu pojaviti prilikom korištenja softvera i na temelju toga ispraviti te probleme kako bi poboljšali svoj proizvod. Metode bi se trebale primjenjivati bilo da se radi o softveru koji će biti na nekom operativnom sustavu ili će biti dostupan na webu jer korištenje weba je danas čak i važnije od korištenja samog računala. Upravo zbog važnosti weba, važno je i vrednovanje kroz koje se može utvrditi kvaliteta web mjesta i samim time omogućiti korisnicima da nađu kvalitetne softvere. Također, u tu kategoriju se ubrajaju i web stranice jer bez kvalitetne web stranice je teško pristupiti kvalitetnom softveru. Dakle, kvaliteta je definitivno jedna od najvažnijih komponenti pojedinog softvera bilo da se radi o programu, mobilnoj aplikaciji, web aplikaciji ili web mjestu. Naravno, osim kvalitete je važno i da je softver ima dobar programski kod, ali korisnicima je također važno i da koriste softver koji ima dobar dizajn i koji je lako koristiti. Svrha vrednovanja i korištenja raznih metoda je upravo u tome da se ispitaju sve važne komponente i poprave mogući problemi kako bi korisnici mogli imati pristup softveru koji će biti kvalitetan, koji će imati dobro dizajnirano sučelje jer će na taj način biti i lakši za korištenje i koji će moći obaviti zadatke koje korisnik pred njega postavi.

LITERATURA

1. Ahmad, N.; Boota, M. W.; Masoom, A. H. (2014) *Smart Phone Application Evaluation with Usability Testing Approach*, Journal of Software Engineering and Applications, URL: http://file.scirp.org/pdf/JSEA_20141127170002 99.pdf (2017-09-25)
2. AQC Lab, URL: http://www.aqclab.es (2017-01-20)
3. Boehm, B. W.; Brown, H.; Lipow,M. (1978) "*Quantitative Evaluation of Software Quality*," TRW Systems and Energy Group, URL: http://csse.usc.edu/TECHRPTS/1976/usccse76-501/usccse76-501.pdf (2017-07-10)
4. Consortium for IT Software Quality, URL: http://it-cisq.org (2017-07-10)
5. Dubey, S.K.; Soumi G.; Ajay R. (2012) "*Comparison of Software Quality Models: An Analytical Approach*," International Journal of Emerging Technology and Advanced Engineering, Volume 2, Issue 2, URL: http://www.ijetae.com/files/Volume2Issue2/IJETAE_0212_20.pdf (2016-12-15)
6. ECRIM News, URL: https://ercim-news.ercim.eu (2017-01-28)
7. Flood, D.; Harrison, R.; Iacob, C.; Duce, D. (2012*) Evaluating Mobile Applications: A Spreadsheet Case Study*, Oxford Brookes University URL: https://ai2-s2-pdfs.s3.amazonaws.com/0100/af54a68e7a117f4a418de8cd4de0ac4b444d.pdf (2017-09-26)
8. Ghayathri, J.; Priya, E. M. (2013) *"Software Quality Models: A Comparative Study,"* International Journal of Advanced Research in Computer Science and Electronics Engineering (IJARCSEE) ,Volume 2, Issue 1, pp 42-51, URL: http://www.ijarcsee.org/index.php/IJARCSEE/article/view/304/271 (2017-04-10)
9. Hom, J. (1998) *The Usability Methods Toolbox Handbook*, URL: http://www.idemployee.id.tue.nl/g.w.m.rauterberg/lecturenotes/usabilitymethodstoolboxhandbook.pdf (2017-01-25)
10. Leavitt, M.O.; Shneiderman, B. (2006) *Research-Based Web Design and Usability Guidelines*, 2nd edition, Health and Human Services Dept., URL:

https://www.usability.gov/sites/default/files/documents/guidelines_book.pdf?post=yes (2017-04-20)

11. Ljubić, S. (2011) *Upotrebljivost mobilnih aplikacija*, Sveučilište u Rijeci, Tehnički fakultet, URL: https://www.researchgate.net/profile/Sandi_Ljubic/publication/265237014_Mobile_Applications_Usability_Upotrebljivost_mobilnih_aplikacija_PhD_Qualifying_Exam_--_in_Croatian/links/58f0831a458515ff23a8b9d3/Mobile-Applications-Usability-Upotrebljivost-mobilnih-aplikacija-PhD-Qualifying-Exam--in-Croatian.pdf (2017-09-26)
12. ISO/IEC 25010:2011, (2011) *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, International Organization for Standardization, Geneva, Switzerland, URL: https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-1:v1:en (2016-06-05)
13. ISO/IEC 25000:2014, (2014.) *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Guide to SQuaRE*, International Organization for Standardization, Geneva, Switzerland, URL: https://www.iso.org/obp/ui/#iso:std:iso-iec:25000:ed-2:v1:en (2016-06-03)
14. Ivory, M. Y.; Hearst, M. A. (2001.) *The State of the Art in Automating Usability Evaluation*, ACM Computing Surveys, Vol. 33, No. 4, URL: http://acs.ist.psu.edu/ist521/papers/ivoryH01.pdf (2016-11-10)
15. Jackson, M.; Crouch, S.; Baxter, R. (2011) *Software Evaluation: Criteria-based Assessment*, Software Sustainability Institute, URL: https://www.software.ac.uk/sites/default/files/SSI-SoftwareEvaluationCriteria.pdf (2017-06-05)
16. Nielsen Norman Group, URL: https://www.nngroup.com/articles (2017-04-25)
17. Miguel, J. P.; Mauricio, D.; Rodríguez G. (2014.) *A Review of Software Quality Models for the Evaluation of Software Products*, International Journal of Software Engineering & Applications (IJSEA), Vol.5, No.6, URL: https://arxiv.org/ftp/arxiv/papers/1412/1412.2977.pdf (2017-03-14)
18. Moumane, K.; Idri, A; Abran, A.. (2016) *Usability evaluation of mobile applications using ISO 9241 and ISO 25062 standards*, Springer Plus, URL:

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4851667/pdf/40064_2016_Article_2171.pdf (2017-09-25)

19. M2 Scientifics LLC, URL: https://www.m2scientifics.com/ (2017-02-21)
20. Orehovački, T. (2010) *Proposal for a set of quality attributes relevant for Web 2.0 application success*, Information Technology Interfaces (ITI), 32nd International Conference on Cavtat URL: http://www.academia.edu/600847/Proposal_for_a_set_of_quality_attributes_relevant_for_Web_2.0_application_success (2016-10-11)
21. Orehovački, T.; Granić. A.; Kermek, D. (2011) *Exploring the Quality in Use of Web 2.0 Applications: The Case of Mind Mapping Services*, LNCS, volume 7059, URL: http://gplsi.dlsi.ua.es/congresos/qwe11/fitxers/QWE11_Orehovacki.pdf (2017-05-20)
22. Plantak Vukovac, D.; Orehovački, T. (2010.) *Metode vrednovanja web upotrebljivosti*, Conference Paper URL: https://bib.irb.hr/datoteka/473208.PlantakVukovac_Orehovacki.pdf (2016-11-25)
23. Pressman, S. (2010) *Software Engineering: A Practitioner's Approach* (Seventh edition), McGraw-Hill, - Higher Education, URL: http://dinus.ac.id/repository/docs/ajar/RPL-7th_ed_software_engineering_a_practitioners_approach_by_roger_s._pressman_.pdf (2016-05-20)
24. Salvucci, D. D.; Goldberg, J. H. (2000) *Identifying fixations and saccades in eye-tracking protocols. In Proceedings of the Eye Tracking Research and Applications Symposium,* New York: ACM Press, URL: https://pdfs.semanticscholar.org/178d/c4cd8c23f07d10ffd3797d5977c76f3dde15.pdf (2017-07-10)
25. SIG.eu, URL: www.sig.eu (2017-06-05)
26. Singh, J. (2015) *7 Ways To Win At Mobile Application Testing*, URL: https://www.axelerant.com/resources/articles/7-ways-win-mobile-application-testing (2017-09-25)
27. Sommerville, I. (2011) *Software Engineering*, Ninth Edition, Boston: Person, URL: https://edisciplinas.usp.br/pluginfile.php/2150022/mod_resource/content/1/1429431793.203Software%20Engineering%20by%20Somerville.pdf (2016-04-20)

28. Test Institute, URL: http://www.test-institute.org/index.php (2017-02-20)
29. Usability Body of Knowledge, URL: http://www.usabilitybok.org (2017-07-11)
30. Usability.gov, URL: https://www.usability.gov (2017-06-05)
31. Usability First, URL: http://www.usabilityfirst.com (2017-04-28)
32. Usability Home, URL: http://www.usabilityhome.com (2016-12-15)
33. Ventayol, A. (2016) *The 3 Methods for Testing Your Mobile App*, URL: https://bugfender.com/blog/the-3-methods-for-testing-your-mobile-app/ (2017-09-26)
34. What Users Do, URL: https://www.whatusersdo.com/what-website-usability-scale (2017-05-25)

POPIS SLIKA

Slika 1. Modeli kvalitete prema Thapu 14
Slika 2. Model za tijek rada temeljen na snimci sustava 26
Slika 3. Grupacija atributa kvalitete i njihovi pod-atributi 28
Slika 4. Toplinska mapa 32
Slika 5. Primjer Saccade putanje 33
Slika 6. Model životnog vijeka mobilne aplikacije 42
Slika 7. Okvir za testiranje mobilnih aplikacija 45

POPIS TABLICA

Tablica 1. Atributi kvalitete 4

Tablica 2. Grupiranje kriterija ocjenjivanja 21

SAŽETAK

Programski proizvodi, ali i web aplikacije su ključan dio korisnikove interakcije sa računalom ili mobitelom ili nekim drugim uređajem koji ga mogu pokrenuti. Naravno, budući da se radi o nečemu što će korisnici koristiti na dnevnim razinama potrebno je utvrditi hoće li odabrani proizvod biti ono što korisnik traži. Budući da korisnici prvenstveno žele kvalitetne programske proizvode, tu kvalitetu je potrebno i osigurati. Upravo zbog toga javlja se potreba za vrednovanjem proizvoda koje se zasniva na raznim metodama.

Ključne riječi: Programski proizvod, kvaliteta, ISO standardi, vrednovanje kvalitete, metode vrednovanja, web upotrebljivost, vrednovanje web upotrebljivosti, mobilna upotrebljivost

ABSTRACT

Program products and web applications are a key part of user interaction with a computer or cell phone or some other device that can launch it. Of course, since it is something that users will use on daily basis, it is necessary to determine if the selected product is what the user is looking for. Since users primarily want quality program products, this quality needs to be ensured. That need is the core of product valuation based on various methods.

Keywords: Program product, quality, ISO standards, qualitly evaluation, methods of evaluation, web usability, evaluating web usability, mobile usability