
PROGRAMMING
SOFTWARE
BUGS
TESTING
METHODS
DATABASE
USABILITY

METRIKA I KVALITET SOFTVERA

Još početkom'80-ih počeo se prvi put pojavljavitati ideje da se na neki način ustanove ili procene karakteristika softvera koje bi se mogle na neki način izmeriti.

Na taj način mogla bi se izvršiti komaparacija za odredeni softver, analizirati njegovi atributi, ali i proces njegovog razvoja i rada – Tu je Životni ciklus softvera,

METRIKA I KVALITET SOFTVERA

Prilikom izgradnje softvera, veoma je važno znati da se greške u softveru mogu naći u bilo kojoj fazi njegovog razvoja.

Zato je bitno da se greške pronađu sto je pre moguće. Da se MORA pronaći način procene broja potencijalnih grešaka u sistemu.

Ne možete kontrolisati ono što ne možete meriti" - Tome DeMarco (još davne 1982 godine)

METRIKA I KVALITET SOFTVERA

Metrike za pouzdanost dizajna i koda se uglavnom odnose na veličinu siftvera izraženu u broju linija koda (LOC-Lines Of Code), kompleksnost koda izraženu preko ciklomatskog broja (McCabe) i modularnost softvera.

Što je modul kompleksniji veće su poteškoće u negovom razumevanju i veća je verovatnoća defekata nego u manje kompleksnim modulima .

Iako su veličina i kompleksnost korisne metrike često se koristi i njihova korelacija da bi se dobile dodatne informacije.

Za kvalitet objektno orijentisanih struktura koriste se OO metrike.

METRIKA I KVALITET SOFTVERA

Broj linija koda može biti izbrojan samo po završetku programa, što se dešava u veoma kasnoj fazi projekta. /Myers – još davne 1982 godine

Alternativna metrika – metoda funkcionalnih tačaka meri veličinu projekta na osnovu funkcionalnosti, datih u klijentovoj ili tenderskoj specifikaciji zahteva.

- Metoda je prezentovana davne 1979 godine.

METRIKA I KVALITET SOFTVERA

Metrički sistem:

- skup kriterijuma, parametara, mernih uređaja,podataka i jedinica za generisanje i prikazivanje rezultata merenja
- podrazumeva procese evaluacije i monitoringa performansi

Rezultati merenja:

- objektivni i sirovi podaci koji se mogu automatski generisati

METRIKA I KVALITET SOFTVERA

Softversko inženjerstvo se bavi razvojem efikasnog, pouzdanog i bezbednog softvera kao proizvoda namenjenog tržištu. ANSI/IEEE729/1983).

Testiranje softvera obuhvata različite vrste testiranja kako bi se osiguralo da softverski proizvod neće imati funkcionalne i nefunkcionalne nedostatke.

Metrike i testovi su najbolji način sagladevanja arhitekture tj. ispunjenosti nefunkcionalnih zahteva.

Metrika za verifikaciju i validaciju softvera

Metrika softvera

Sa rastom kompleksnosti realizovanih funkcija i primena posebno je narastao zahtev za kvalitetom softvera u pogledu pouzdanosti - posebno kod kritičnih softvera, pogodnosti za testiranje i održavanje, ponovne upotrebljivosti, otpornosti na greške i drugih faktora kvaliteta softvera.

Ima mnogo metoda za testiranje sofvera, ali treba izabrati pravu metodu.

Za ostvarivanje potrebnog kvaliteta softvera od velikog je značaja određivanje metrike softvera i sprovođenje adekvatnog procesa testiranja.

Metrika softvera

Upotrebljivost softvera je u velikoj meri određena načinom na koji se meri.

Metrika je „sveobuhvatni termin koji opisuje metod koji se koristi za merenje nečega, rezultirajuće vrednosti dobijene merenjem, kao i izračunati ili kombinovani skup mera" (Labate, 2016).

Metrika se definiše i kao „sistem ili standard merenja predstavljen u jedinicama koje se mogu koristiti za opisivanje više od jednog atributa" (Mifsud, 2015)

Metrika softvera

- ✓ Razvoj softvera je proces koji je veoma specifičan i složen.
- ✓ Ideja svakog poslovanja je zadovoljan korisnik.

- ✓ Kada se radi neka od vrsta testiranja softvera, moraju se kreirati neki izveštaji o potencijalnim propustima (Defect reports) i naravno na kraju kao važan factor svega jesu i neke metrike.

- ✓ Ti podaci su veoma korisni koji mogu da budu sastavni deo baze znanja Project i Product Menadžerima za neke nove projekte.

Metrika softvera

Metrika softvera obično se deli na:

- metrike procesa (process metrics)
- metrike proizvoda (product metrics)
- metrike projekta (project metrics)

Potrebno je eliminsati zavisnost metrike od metodologije razvoja softvera.

Postoje:

- Tehničke metrike
- Human metrike

Metrika softvera

Proces metrike su vezane za aktivnosti povezane sa produkcijom softvera.

Koriste za poboljšanje razvoja i održavanja softvera.

Primeri ovih metrika su

- efektivnost uklanjanja defekata za vreme razvoja,
- vreme odziva za fiksiranje problema i slično.

Ove metrike vode dugoročnijem poboljšanju procesa.

Metrike projekta uglavnom se odnose na resurse projekta kao što su ljudi, znanje, hardver.

Primeri ovih metrika su broj developera, budžet, vremenski raspored i produktivnost

Metrika softvera

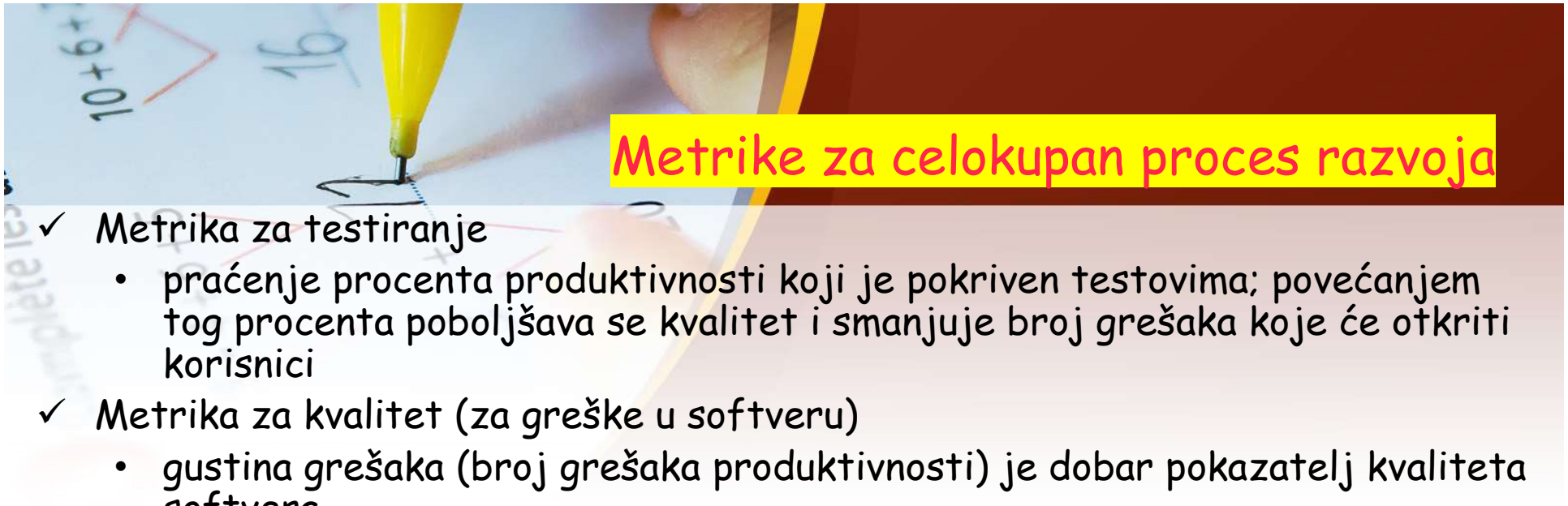
Metrika softvera obično se deli na:

- Metrika za rokove
  - broj taskova koji su uspešno završeni na vreme
  - broj taskova koji nisu završeni na vreme
  - broj taskova čiji su rokovi promenjeni
  - broj taskova koji su odloženi za kasnije
- Metrika za definisanje korisničkih zahteva
  - broj taskova ili zahteva koji su vezani za samu specifikaciju
  - broj novih zahteva koji su naknadno pristigli
  - RFC dijagram (Requests For Change)

Metrika softvera

Metrika softvera obično se deli na:

- Metrika za testiranje
  - praćenje najčešće procenta SLOC (Source Lines of Code)koji je pokriven testovima;
  - Posebno se prati taj procenat jer povećanjem tog procenta poboljšava se kvalitet i smanjuje broj grešaka koje će otkriti korisnici
- Metrika za ukupni rizik projekta
  - Metrika za kvalitet (za greške u softveru)
    - gustina grešaka
    - broj otkrivenih i otklonjenih grešaka



Metrike za celokupan proces razvoja

- ✓ Metrika za testiranje
  - praćenje procenta produktivnosti koji je pokriven testovima; povećanjem tog procenta poboljšava se kvalitet i smanjuje broj grešaka koje će otkriti korisnici
- ✓ Metrika za kvalitet (za greške u softveru)
  - gustina grešaka (broj grešaka produktivnosti) je dobar pokazatelj kvaliteta softvera
  - broj otkrivenih i otklonjenih grešaka (fault arrival and closing rates); proizvod je spreman za isporuku kada ove mere padnu na ≈0
- ✓ Metrika za ukupni rizik projekta
  - stepen spremnosti proizvoda za instaliranje i rad (zahteva veće angažovanje IT i product menadžera)

Metrika softvera

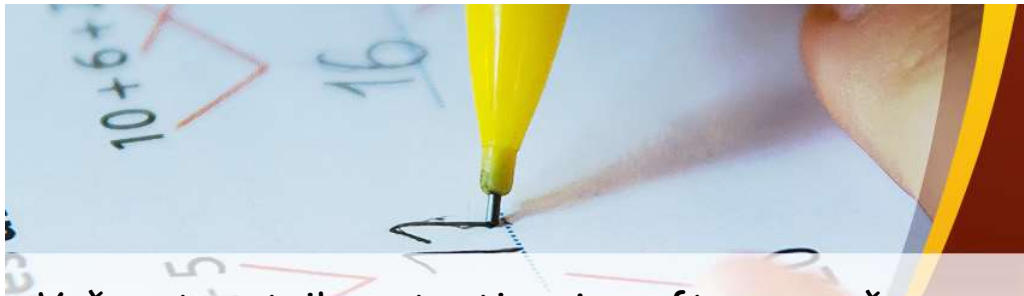
"Upotrebljivost softvera je u velikoj meri određena načinom na koji se meri."- Myers

Metrika je :

„Sveobuhvatni termin koji opisuje metod koji se koristi za merenje nečega, rezultirajuće vrednosti dobijene merenjem, kao i izračunati ili kombinovani skup mera" (Labaté, 2016).

Metrika se definiše i kao :

„Sistem ili standard merenja predstavljen u jedinicama koje se mogu koristiti za opisivanje više od jednog atributa" (Mifsud, 2015)



Metrika softvera

Važnost metrike u testiranju softvera može se navesti kroz nekoliko koraka:
- One pomažu u donošenju odluka za sledeću fazu aktivnosti
- One pomažu u razumevanju potrebe za poboljšanjem softvera
- Olakšava postupak odlučivanja ili promene tehnologije.

To je skup aktivnosti kojima se na osnovu određenih metrika i indikatora performansi utvrđuje kojim tempom se testira zadati sistem.

METRIKA SOFTVERA je:

- kontinualni proces
- podrazumeva podešavanje mera i dobijanje informacija i preduzimanje nekih mera iz iskustva ako je potrebno

Metrika softvera

Nakon definisanja šta su softverske metrike potrebno je navesti i kategorije podataka koje se mere - to su takozvani merni podaci.
Merni podaci omogućuju procenu uspeha neke aktivnosti, a uspeh se svakako može definisati kao postizanje određenog cilja.

- Cilj metrika koje se definisu kroz testiranje softvera su:
- ✓ Postizanje određenog kvalitete softvera,
- ✓ Zadovoljstvo korisnika,
- ✓ Isporuka pouzdanog softvera,
- ✓ Mali troškovi razvoja.
- ✓ Mali troškovi održavanja softvera

Jedan od mernih podataka je pokrivenost testom. Pokrivenost testom može se razjasniti koji su to delovi softvera testirani.

Metrika softvera

Postavlja se pitanje u suštini kome koristi metrika softvera ?

- Korisnicima
  - Mogu da ispitaju funkcionalnost i kvalitet samog softvera
- Product i Project menadžerima
  - Daje potrebne, tačne informaciju o tome kako softver napreduje
  - Daje potrebne, tačne informaciju o tome kako je softver završen
  - Daje potrebne, tačne informaciju o tome da li je softver uspešan ili nije
  - Daje potrebne, tačne informaciju o održavanju softvera (postoji tim)
- Softverski tim koji razvija softver
  - Daje potrebne, tačne informaciju o aktivnosti koje vode ka uspešnom završetku softvera
- Softverski tim koji održava softver

METRIKA I KVALITET SOFTVERA

Softversko inženjerstvo zahteva upotrebu kako analitičkih, tako i deskriptivnih alatki koje su razvijene u okviru računarskih nauka, zajedno sa rigoroznim pristupom koji donose inženjerske disipline u cilju postizanja odgovarajuće pouzdanosti i bezbednosti, a sve to kroz timski rad softverskih inženjera koji funkcionišu u jednom kohezionom okruženju.

Ako se pravilno upotrebi, test metrika može značajno da doprinese inicijativi unapređenja postojećeg procesa razvoja softvera

Metrika softvera

Slaganje sa eksplicitnim izraženim funkcionalnim zahtevima i zahtevima za performanse, eksplicitno dokumentovanim razvojnim standardima, i implicitnim karakteristikama koje se očekuju od svakog stručno razvijenog softvera" (Pressman, davne 2001 godine)

Bilo kakve aktivnosti merenja moraju imati jasan cilj.

Prilikom testiranja softvera, kreiraju se izveštaji o propustima (Defect reports) i druge metrike.

Metrike kvaliteta softverskog procesa zasnovane na metrikama kompleksnosti programa

Metrika softvera

Kako se prikupljaju neki podaci koji su neophodni za metriku softvera:

- Analiza prethodno završenih projekata
- Analizom iz sličnih projekata koji je već neko radio
- Pravljenje baze podataka za buduće projekte (broj SLOC (Source Lines of Code) iz prethodnih projekata, produktivnost programera, rokovi, greške,…)

Kako izabrati metriku za konkretan projekat?

- ✓ Povezati metriku sa poslovnim ciljevima
- ✓ Odabrati metriku koju razumeju i menadžeri, i programeri
- ✓ Odabrati merenja koja se mogu sprovesti
- ✓ Odabrati metriku koja je stvarno bitna za aktivnosti datog projekta
- ✓ Odabrati konzistentnu metriku za više projekata odabrati metriku koja može da ukaže na načine za povećanje produktivnosti i/ili kvaliteta
- ✓ Odabrati metriku za koju se mogu definisati odgovarajuće akcije
- ✓ Najbolje metrike su one koje prirodno proističu iz radnog procesa

METRIKA I KVALITET SOFTVERA

Pouzdanost je jedan od ključnih faktora kvaliteta i zato pouzdanost isporučenog koda zavisi od kvaliteta svih procesa

Podaci o pouzdanosti softvera pomažu u povećanju produktivnosti, redukciji isporučenih defekata, redukciji frekvecija prekida i redukciji padova sistema.

Da bi se dobili podaci u tim fazama treba upotrebiti softverske metrike za pomoć u vrednovanju pouzdanosti.

Na taj način se povećava kvalitet a sa samim tim i pouzdanost.

METRIKA I KVALITET SOFTVERA

Kod testnih metrika postoje dva pristupa za vrednovanje pouzdanosti:

- Prvi se odnosi na vrednovanje test plana da se obezbedi da sistem ima funkcionalnost predviđenu specifikacijom zahteva.
- Drugi pristup vezan za pouzdanost se sastoji u vrednovanju broja grešaka u kodu i i otklanjanju istih.

Podaci o greškama se dobijaju za vreme testiranja i stvarne upotrebe softverskog proizvoda

Ovi podaci se koriste za izgradnju modela pouzdanosti na osnovu koga se može izvršiti predikcija vremena sledeće greške na osnovu istorijata prethodnih grešaka.

STATISTIKA!TESTIRANJA SOFTVERA

Metrika softvera

Standard ISO/IES 9126 se bavi aspektima koji udvrđuju kvalitet softverske aplikacije:
-Model kvaliteta,
-Eksterna metrika
-Interna metrika
-Pokazatelj kvaliteta u upotrebi.

Pokrivenost koda je metrika koja se koristi za izražavanje koje delove softvera testovi ne koriste, koliki deo izvornog koda je testiran.

Prijavljivanje grešaka

- Svaka greška se prijavljuje u dogovorenom formatu:
  - Ko je grešku pronašao
    -podaci o testeru
  - Opis greške
    - sadrži i opis kako greška može biti reprodukovana (to je posebno izazovno jer ponekad ne možemo tako lako da reprodujujemo grešku)
  - Test okruženje u kojem se greška desila
  - Prioritet
  - Tip greške
  - status
  - Komentar

Po velikom broju literature postoji tvrdnja da je ;

- Korisnike je teže obučiti da prijavljuju potencijalne greške po unapred dogovorenoj proceduri

	Attribute	Meaning
Identification	Id / Number	Unique identifier/number for each report
	Test object	Identifier or name of the test object
	Version	Identification of the exact version of the test object
	Platform	Identification of the HW/SW platform or the test environment where the problem occurs
	Reporting person	Identification of the reporting tester (possibly with test level)
	Responsible developer	Name of the developer or the team responsible for the test object
	Reporting date	Date and possibly time when the problem was observed
Classification	Status	The current state (and complete history) of processing for the report (section 6.6.4)
	Severity	Classification of the severity of the problem (section 6.6.3)
	Priority	Classification of the priority of correction (section 6.6.3)
	Requirement	Pointer to the (customer-) requirements which are not fulfilled due to the problem
	Problem source	The project phase, where the defect was introduced (analysis, design, programming); useful for planning process improvement measures
Problem description	Test case	Description of the test case (name, number) or the steps necessary to reproduce the problem
	Problem description	Description of the problem or failure that occurred; expected vs. actual observed results or behavior
	Comments	List of comments on the report from developers and other staff involved
	Defect correction	Description of the changes made to correct the defect
	References	Reference to other related reports

Metrika softvera

Softverski timovi se organiziraju u skladu sa nekom od metodologija, a metodologije slede jedan ili više modela razvoja softvera.

Potrebno je pratiti različite parametre i obezbediti kvalitet softvera, kreirati planove vezane za kvalitet i ostvariti potrebnu kontrolu.

Ono što je karakteristično za današnji razvoj softvera je činjenica da se vreme od ideje do razvoja softvera, sve više skraćuje.

MANJAK VREMENA ZA TESTIRANJE SOFTVERA

Metrika softvera

Statički atributi koda mogu omogućiti da se nauči puno o strukturi sistema, kao i o područjima u programu koji mogu biti potencijalni uzročnici bagova (grešaka).

Što je kod kompleksniji, program je teže testirati, održavati a to prouzrokuje i mnoge greške.

POJMOVI:

- ✓ **Veličina** (size): Tipična merenja linija koda (LOC) , Metrike linije koda (Lines of Code – LOC)
- ✓ **Cyclomatic Complexity**: Mere kontrole toka unutar modula.
- ✓ **Halstead kompleksnost**: Mere broja operatora i operanada u kodu.
- ✓ **Informacijski tok** (Information Flow): Merenja toka podataka u i iz modula.
- ✓ **Kompleksnost sistema** (System Complexity): Merenja kompleksnosti celokupnog sistema u terminima **održavanja i/ili dizajna**.
- ✓ ***Objektno-orijentirane strukturalne metrike***: Merenje različitih struktura objektno orijentiranih programa (u odnosu na funkcionalno dizajnirane programe).

METRIKA I KVALITET SOFTVERA

Veličina je jedan od **najčešćih atributa softvera**.

Neka od ključnih pitanja o kojima korinici žele da znaju uključuje veličinu projekta.

- Koliko je velik softver ?
- Koliko je vremena trebalo da se uradi dati projekat?
- Koliko je velik neki softver u odnosu na druge projekte?
- Koliko napora je uloženo za imlementaciju softvera?
- Kako se kvalitet (softvera koji sada radi) poredi sa drugim projektima?
- Kako se produktivnost koja je uložena u dati projekat poredi sa drugim projektima?

Veličina (size) je osnova svih ovih metrika.

- STANDARDIZACIJA

Metrika softvera

- Alternativna metrika – metoda funkcionalnih tačaka meri veličinu projekta na osnovu funkcionalnosti, datih u klijentovoj ili tenderskoj specifikaciji zahteva.

Metrike kvaliteta softverskog procesa koje se računaju zasnovano na metrikama veličine su:

-Metrike gustine grešaka

-Metrike ozbiljnosti grešaka

-Metrike efektivnosti uklanjanja grešaka

METRIKA I KVALITET SOFTVERA

Softverske metrike mere su uspešnosti softverskog procesa.

Sama metrika se može podeliti na metriku kontrole i metriku predviđanja.

Potrebno je eliminsati zavisnost metrike od metodologije razvoja softvera.

Ako prijavljeni problem uspemo da reprodukujemo, onda ćemo ga sigurno i rešiti. (Reproduction= solution)

Metrika softvera

- ✓ Probabilistički model
- ✓ Modeli rasta pouzdanosti
- ✓ Modeli stope neuspeha
- ✓ Mek-Kejbova ciklomatična složenost

$C(G) = E - V + 2P$,

gde je:

E = broj grana grafa

V = broj čvorova grafa

P = broj povezanih komponenti grafa

Metrika softvera

Izgradnja visoko pouzdanog softvera zavisi od učešća atributa kvaliteta u svakoj fazi životnog ciklusa razvoja sa naglaskom na prevenciju grešaka, specijalno u ranim fazama.

Potrebe korisnika: određeni nivo kvaliteta, a ne samo funkcionalnost

Da bi se ovi atributi merili sa ciljem poboljšanja kvaliteta i pouzdanosti potrebno je definisati softverske metrike za svaku razvojnu fazu (dokumentaciju, kod, planove testiranja i samo testiranje na samo kraju, kao i održavanje.

METRIKA I KVALITET SOFTVERA

U razvojnom ciklusu softvera sve je značajniji zadatak Procesa Testiranja Softvera ili Verifikacije i Validacije koji treba da obezbedi zahtevani nivo poverenja u ispravnost (korektnost) softvera kao i obezbeđenja ostalih zahtevanih karakteristika softvera.

Optimizacija koda i brzina

METRIKA I KVALITET SOFTVERA

Određivanje mera koje reflektuju bitne karakteristike svakog softverskog projekta

- obim projekta i kvalitet izlaznog softvera
- isporučivanje proizvoda i vremenski rokovi koji su potrebni za izradu softvera
- napor potreban da se projekat završi
- Testiranje softvera
- Pravljenje kvalitetne dokumentacije

METRIKA I KVALITET SOFTVERA

- **Održavanje softvera** je
  - Proces modifikacije softverskog sistema ili komponente **nakon isporuke** radi ispravljanja grešaka, poboljšanja performansi ili drugih atrubuta ili prilagodjenja novoj sredini [IEEE Std 610.12-1999]
  - Softverski proizvod podleže modifikaciji koda i odgovarajuće dokumentacije usled nekog problema ili potrebe za poboljšanjem.
  - Cilj toga je da se modifikuje postojeći softverski proizvod a da se u isto vreme sačuva njegov integritet. [ISO Std 12217]

METRIKA I KVALITET SOFTVERA – Evolucija softvera

- **Evolucija softvera** je
  - Niz aktivnosti, tehničkih koje obezbeđuju da softver nastavlja da ispunjava organizacione i poslovne ciljeve pritom iskorišćavajući poverena sredstva na najbolji način (Institut za istraživanja na polju evolucije softvera)*[Research Inst. On Sw. Evolution]*
  - Svaka programerska delatnost koja je namenjena stvaranju nove verzije softvera od neke ranije verzije (Lehman i Ramil 2000 godine)
  - Primena mera (aktivnosti) i procesa održavanja softvera koji stvaraju novu radnu verziju softvera sa promenjenom funkcionalnošću (prema iskustvima korisnika) ili sa svojstvima prethodne radne verzije, zajedno sa odgovarajućim merama i procesima koje osiguravaju kvalitet, i rukovođenjem tim merama i procesima (Ned Chappin davne 1999 godine)

METRIKA I KVALITET SOFTVERA

Koji su razlozi zašto neko menja neki softver koji mu i dalje radi?
Neki od razloga su!

- ✓ Poboljšanja (Enhancements)
- ✓ Popravak (Repairs)
- ✓ Performanse (Performance improvements)
- ✓ Menjanje, dodavanje, sa ciljem da softver brže bolje i sigurnije radi
- ✓ Zaštita sistema
- ✓ Konfiguracija hardvera (Configuration changes)
- ✓ (Environment upgrades)

METRIKA I KVALITET SOFTVERA - Tipovi održavanja softvera

- Prema standardu ISO/IEC 14674 za softverski inženjering - održavanje softvera

  Adaptivno održavanje

  - "Modifikacija softverskog proizvoda koja se izvodi nakon isporuke, a sa ciljem da se tom softverskom proizvodu sačuva upotrebna vrednost u promenjenoj sredini ili sredini koja se upravo menja."

  Korektivno održavanje

  - "Reaktivna modifikacija softverskog proizvoda koja se vrši nakon isporuke, radi popravke otkrivenih grešaka."

Softver ima atribute koji ga opisuju:

- ✓ CORRECTNESS (korektnost)
- ✓ MAINTAINABILITY- (mogućnost održavanja)
- ✓ RELIABILITY
- ✓ (pouzdanost)
- ✓ EFFICIENCY (efikasnost)
- ✓ INTEGRITY
- ✓ USABILITY (primenljivost)

- TESTABILITY - (mogućnost testiranja)
- FLEXIBILITY (prilagodljivost)
- PORTABILITY (prenosivost)
- REUSABILITY (obnovljivost)
- INTEROPERABILITY

- Prema standardu ISO/IEC 14674

  Perfektivno održavanje

  - "Modifikacija softverskog proizvoda nakon isporuke radi unapređenja performansi ili održivosti."(takođe uključuje i dodavanje novih karakteristika)

  Preventivno održavanje

  - "Modifikacija softverskog proizvoda nakon isporuke, sa ciljem da se detektuju i isprave skrivene greške u tom softverskom proizvodu pre nego da postanu delotvorne."

METRIKA I KVALITET SOTVERA Tipovi održavanja softvera

Klasifikacija zasnovna na objektivnom dokazu [Chapin i drugi 2001]

- Klasifikuje evoluciju softvera i mere i procese održavanja unutar softvera (uključujući dokumentaciju)
  - koda
  - funkcionalnosti za korisnika
- poredeći softver pre i posle evolucije

METRIKA I KVALITET SOFTVERA Tipovi održavanja softvera

PROBLEMI ODRŽAVANJA - Lienzi Swanson1977 -NosekPalvia1990

- Velika količina zahteva za održavanje različitih softvera
- Neefikasno korištenje programerskih resursa u timovima za održavanje
- Povećan angažman nakon implementacije
- Određivanje prioriteta uvođenja novih funkcionalnosti
- Povećan broj grešaka u novim isporukama softvera,
- Česta promena programera u timovima za održavanje,
- Dokumentacija ako postoji je često nepotpuna i nedosledne,
- Ne-metodološki pristup razvoju softvera koji otežava održavanje
- Manja produktivnost programerskog tima zbog specifičnosti poslova u održavanju

PROBLEMI ODRŽAVANJA

- ✓ Različiti kvalitete programera u timovima za održavanje
- ✓ Sporo uključivanje novih programera u proces održavanja,
- ✓ Manje iskusni programeri koji se često zapošljavaju u timovima za održavanje
- ✓ Nerealna očekivanja korisnika vezano uz rokove za isporuku novih zahteva
- ✓ Osiguravanje i povećanje kvaliteta proizvoda
- ✓ Osiguravanje i povećanje kvaliteta usluga
- ✓ Osiguravanja kontinuiteta održavanja
- ✓ Povećanje efikasnosti procesa održavanja

Lienzi Swanson 1977 –NosekPalvia 1990

Neki od najčešćih problema su:

- Specifikacija
- Česta promena specifikacije
- Česta promena tima programera ili pojedinaca (programera) može da puno utiče na motivaciju i produktivnost
- Kvalitetno napisana dokumentacije
- Nerealna očekivanja korisnika
- Kratko vreme za testiranje softvera

REDIZAJN SOFTVERA

Mada je održavanje jedan neprekidan proces, ozbiljno se mora posmatrati situaija kod redizajniranja softverskog sistema.

Glavno pitanje je da li se sistem može održavati ili ne može.

Neke od problema su:

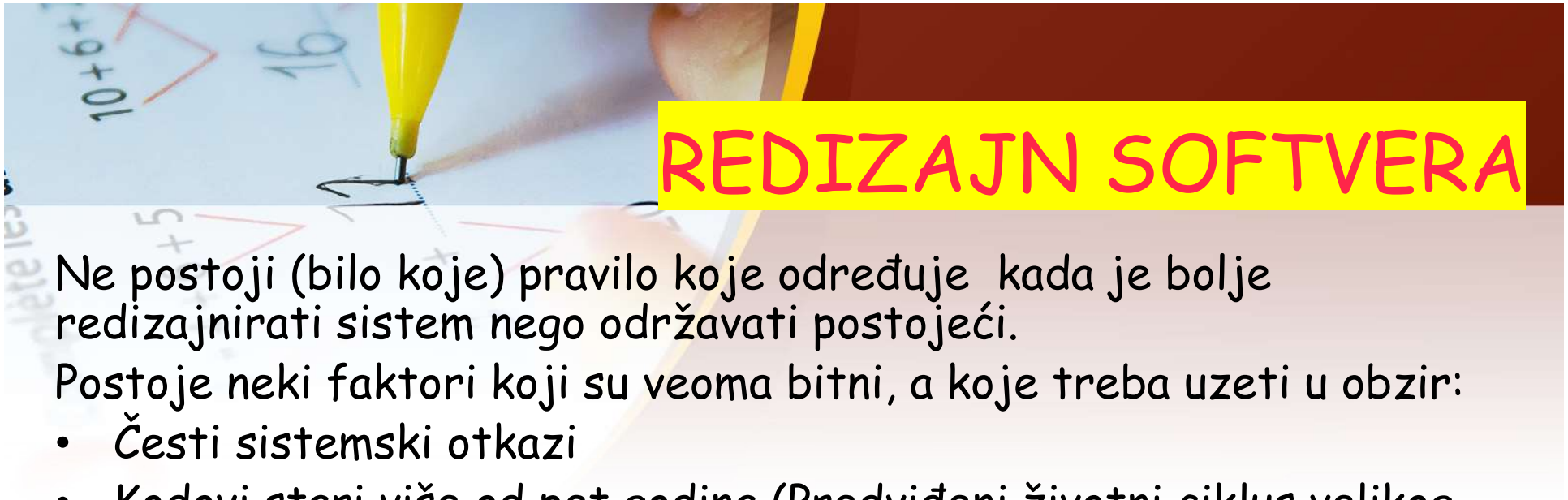
Nove verzije operativih sistema, -

problem sa drajverima.

Drugim rečima treba da postoji sofverski tim koji mogu da analiziraju opšte stanje softvera i kada bude neophodno odlučuju da je softver bolje redizajnirati nego ga nastaviti korektivno održavati.

SUGESTIJA: ANALIZIRATI SLEDEĆE:

Troškovi budžeta i održavanja softvera kada postane sklon bagovima, neefikasan i skup, moraju biti dobro iznalizirani sa onima koji se zalažu za redizajniranje sistema čak i kada redizajniranje nije najbolje rešenje.



REDIZAJN SOFTVERA

Ne postoji (bilo koje) pravilo koje određuje kada je bolje redizajnirati sistem nego održavati postojeći.

Postoje neki faktori koji su veoma bitni, a koje treba uzeti u obzir:

- Česti sistemski otkazi
- Kodovi stari više od pet godina (Predviđeni životni ciklus velikog dela aplikacija je 3- 5 godina. Pogoršanje karakteristika softvera sa godinama je rezultat brojnih u zargonu reči poput " opravki " i "krpljenja".
- Previše kompleksne programske strukture
- Programirano i rađen softver za predhodnu generaciju harvera

METRIKA I KVALITET SOFTVERA Tipovi održavanja softvera

- ✓ Obuka
- ✓ Konsultacija
- ✓ Vrednovanje
- ✓ Redizajn
- ✓ *Ažuriranje*

METRIKA I KVALITET SOFTVERA Tipovi održavanja softvera

- ✓ Testiranje
- ✓ Dokumentacija
- ✓ Performanse
- ✓ Adaptacija
- ✓ Ograničenja u hardveru i operativnom sistemu
- ✓ Izmena na postojećem kodu i ažuriranje
- ✓ Nove verzije

METRIKA I KVALITET SOFTVERA

- Promene u zahtevima korisnika
  - Modifikacije i proširanja na osnovu zahteva od strane korisnika
- Otklanjanje grešaka
  - Redovne popravke
  - Vanredne popravke - koje više koštaju usled velikog pritiska (vreme)
- Promene u formatima podataka
  - Novi standardi: UML, XML, wsdl, json,...
- Promene hardvera
- Unapređenja efikasnosti

Održavanje je težak i složen proces.

Sistem je već u eksploataciji

- ✓ svako modifikovanje mora da se uskladi sa potrebama korisnika
- ✓ neki sistemi dopuštaju, a neki ne da izvesno vreme ne budu aktivni
- ✓ mora se pronaći neko alternativno rešenje koje ne bi ugrozilo korisnika (rezervni sistem)

Održavanje je težak i složen proces.

Mogu se javiti personalni i organizacioni problemi

- ✓ nedovoljno razumevanje funkcionisanja sistema i nedostatak potrebnih veština od strane korisnika (rešenje kvalitetna obuka i dokumentacija)
- ✓ nesklad između prioriteta Menadžmenta i korisnika
- ✓ moral tima za održavanje (primer iz prakse-jedno od rešenje je rotacija programera iz razvojnog tima u tim za održavanje)

Održavanje je težak i složen proces.

Mogu se javiti tehnički problemi

- ✓ zbog načina projektovanja i implementacije sistema
- ✓ zbog izabranih hardverskih i softverskih platformi korišćenih u realizaciji (nepouzdane i podložne otkazima)

Projektni budžet održavanja celog sistema-projekta

MODELI PROCESA ZA ODRŽAVANJE

IEEE Model

1.Problem/modication identication, classication, and prioritization Identifikacija problema/promene, klasifikacija i prioretizacija)

2.Analysis (Analiza)

3.Design (Dizajn)

4.Implementation (Implementacija)

5.Regression/system testing (Regresijsko i sistemsko testiranje)

6.Acceptance testing(Testiranje prihvaćenosti)

7.Delivery(Isporuka).

Održavanje je težak i složen proces.

Uputstvo za instalaciju

Uputstvo za instalaciju je tehnički dokument u kome je detaljno, korak po korak, opisan postupak instalacije softvera.

- navode se uslovi neophodni za regularan rad softvera
- hardverska i softverska platforma
- minimalna i maksimalna konfigiracija
- Koji operativni sistem se koristi, (verzije)

Održavanje je težak i složen proces.

Vrste dokumentacije

- ✓ Uputstvo za korisnika
- ✓ Uputstvo za operatera
- ✓ Uputstvo za instalaciju
- ✓ Uputstvo za programere

HALSTEAD metrike kompleksnosti

Halstead metrika kompleksnosti je razvijena od strane Maurice Halsteada sa namenom određivanja kvantitativnih mera kompleksnosti na osnovu operatora i operanada u modulima izvornog (source) koda.

- Pošto se primenjuje na kod, često se koristi i kao metrika održavanja.
- Halstead merenja su uvedena još 1977 godine i predstavljaju najstarija merenja programske kompleksnosti.

HALSTEAD metrike kompleksnosti

- ✓ Halstead volumen (V) opisuje veličinu implementacije algoritma.
- ✓ Računanje V je bazirano na broju operacija koji se izvršavaju i broju operanada koji se pojavljuju u algoritmu.
- ✓ Volumen funkcije trebao bi biti najmanje 20 i najviše 1000.
- ✓ Volumen jedno linijske funkcije bez parametara, koja ima telo funkcije je oko 20.
- ✓ Volumen fajla bi trebao biti između 100 i 8000.

METRIKA I KVALITET SOFTVERA

Problemi sa koji se javljaju u sistemima:

- Često rade na zastarelom hardveru
- Teško ih je održavati, poboljšati i proširiti
- Opšte odsustvo razumevanja sistema:
  - Nema nikoga da objasni kako sistem radi, nedostatak programera, sistema analitičara,
  - Nedostatak dokumentacija i uputstva za upotrebu
  - Sa novim verzijama softvera, problem sa drajverima

METRIKA I KVALITET SOFTVERA

Troškovi kontrole:

Obuhvataju troškove koji se troše za sprečavanje i otkrivanje grešaka softvera u cilju da ih svede na prihvatljiv nivo.

Neuspeli troškovi kontrole:

Obuhvataju troškove koji nisu uspeli da otkriju ili spreče greške koje su se dogodile u softveru.

Ovaj model troškove dalje razlaže na podklase.

Troškovi kontrole se mogu podeliti na interne i eksterne troškove.

METRIKA I KVALITET SOFTVERA

Interni troškovi nedostataka uključuju troškove greške koje su otkrivene u dizajnu, testiranju softvera i testiranju prihvatljivosti, a završen pre nego što je instaliran na opremi kod klijenta, ili pre nego što je klijent preuzeo aplikaciju sa Interneta. (clouda..)

Eksterni troškovi nedostataka uključuju sve troškove vezane za ispravljanje propusta otkrivenih od strane kupaca ili tima za održavanje nakon sto je softverski sistem instalirao dati softver

Metriku za konkretan projekat, šta je najbolje?

U literaturi se kao očigledan primer za potvrdu ne mogućnosti kompletnog testiranja softvera navodi jednostavan misaoni eksperiment koji je zamislio Beizer,a koji se sastoji u tome da ako imamo program koji učitava niz od 10 znakova i izvršava neku od potrebnih operacija nad njim. I ovako jednostavna situacija ima jako veliki broj slučajeva ulaznih vrednosti koje treba testirati i samo testirati.

METRIKA I KVALITET SOFTVERA

- *Postoji i metrika koje se koriste za merenje programske kompleksnosti i koje su izuzetno vredne.*

- *Neke od njih su:*

  Branching complexity (Sneed Metric),

  Data access complexity (Card Metric),

  Data flow complexity (Elshof Metric),