TESTIRANJE SOFTVERA

TESTIRANJE SOFTVERA

PROBLEM: Kada softver nije adekvatno testiran.

"Testiranje softvera je proces izvršavanja programa/funkcije sistema sa ciljem da se otkrije što više grešaka."

Naglašena je posvećenost aktivnosti otkrivanja bigova (grešaka).

Jedan od ciljeva testiranja softvera je da se izbegnu bilo kakvi incidenti i problemi prilikom korišćenja softvera i njihove posledice.

TESTIRANJE SOFTVERA

Testiranje softvera je već više godina, pa i decenija, ustanovljeno kao naučna, odnosno inženjerska disciplina.

"Testiranje softvera se sastoji od aktivnosti dinamičke verifikacije ponašanja programa na bazi konačnog skupa testova, odabranih na pogodan način iz beskonačnog skupa mogućih načina izvršavanja programa, a prema specificiranom očekivanom ponašanju softvera u razvijanoj aplikaciji tj. zahtevanog kvaliteta softvera".

TESTIRANJE SOFTVERA

Bitna razlika :

- ✓ Da li program ili
- ✓ Sistem radi.

Ne postoji savršen softver.

Zato što je jasno da je nemoguće otkriti sve bugove (greške) u softveru.

Od testera se očekuje da proces testiranja softvera učine što efikasnijim i uz što manje troškova obezbede normalan rad određenog softvera.

TESTIRANJE SOFTVERA

U radovima DeMilla - kompleksne greške su povezane sa jednostavnijim greškama.

Kako testiranje skoro nikada ne može da bude kompletno urađeno, ono ne može da pokaže nepostojanje grešaka.

Odatle se nameće zaključak da je cilj testiranja detektovanje što većeg broja grešaka.

Neke greške su kompleksnije od drugih i njihov efekat može biti ozbiljniji.

Testiranje obično ne mora da se fokusira na neki određenu vrstu grešaka zbog postojanja takozvanog grupisanja defekata.

TESTIRANJE SOFTVERA

- **Veoma je važno detaljno isplanirati ceo proces testiranja**, kako bi se slučajno ne bi prikrio neki veliki bag koji može da ugrozi krajnjeg korisnika.
- U mnogim knjigama i časopisima se tvrdi da testiranje softvera odavno nije samo faza u procesu razvoja softvera već paralelni pod-proces.
- U bilo kom softveru u praksi postoji beskonačan broj mogućih testova koje je praktično nemoguće sve izvesti, te je nemoguće u određenom vremenskom periodu, sa ograničenim resursima (ljudi, oprema i alati) izvršiti totalno testiranje koje bi otkrilo sve greške u sofveru.

VRSTE IT PROJEKTA

Prema Snedakeru , postoje sledeće vrste IT projekata:

- Strategijski ili operativni
- Dugoročni ili kratkoročni
- Hardver ili softver
- Razvojni ili implementacioni

Postoje još neke podele vezano za IT projekte:

Kriterijum vremena: dugoročni i kratkoročni

Karakter izvođača: interni, eksterni i kombinovani

VRSTE IT PROJEKTA

- ✓ Pionirski projekti,
- ✓ Repetitivni projekti,
- ✓ Standardni projekti
- ✓ Potencijalni projekti

Prema karakteru ciljeva :

- ❑ (otvoreni, zatvoreni) i
- ❑ složenosti ciljeva (niska, visoka)

KOMPONENTE SOFTVERA

Neki od razlozoga potencijalnog (nastanka) neuspešnog softvera se mogu podeliti na :

- ✓ Složenost problema koji se rešavaju + vremenski okvir koji utiču na razvoja softvera
- ✓ Saradnja među članovima razvojnog tima
- ✓ Česte promene koje "odudaraju od zadate specifikacije".

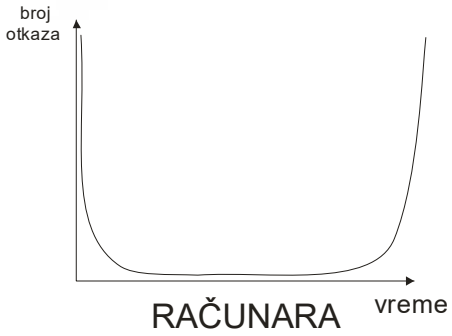
PROGRAM KAO PROIZVOD treba da zadovoljava:

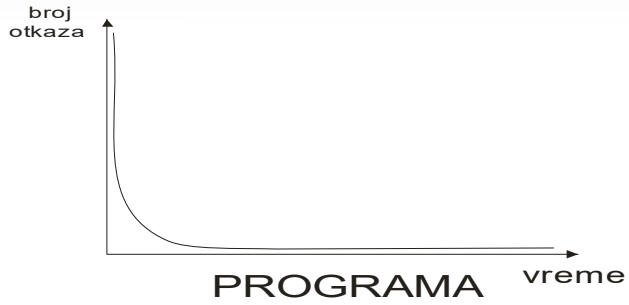
FUNKCIONALNOST

Podrazumeva da program mora odgovarati zahtevima koji proističu iz prirode problema za koji je pisan program. Program je funkcionalan ako zadovoljava razumna očekivanja korisnika

POUZDANOST

Pod pouzdanošću se podrazumeva broj otkaza u jedinici vremena.





PRENOSIVOST

Sposobnost izvršavanja na što više različitih sistema.

TESTIRANJE SOFTVERA

Zašto se onda tolika pažnja u poslednje vreme posvećuje testiranju softvera kada ima toliko ograničavajućih faktora?

Zato je prvenstveni zadatak testera (ili testerskog tima) otkrivanje problema u softveru sa ciljem da se oni otklone i pronađu pre produkcije.

DOKUMENTACIJA

Prema nameni dokumenta se mogu podeliti na :

- **Interna dokumenta** - koji nastaju u kompanijama i za potrebe istih
- **Eksterna dokumenta** - namenjeni za razmenu sa drugima

Razvojna dokumentacija (izveštaji o specifikaciji i dizajnu, opisi programa,…)

Podržava efikasnu **kooperaciju i koordinaciju** između članova razvojnog tima i **efikasan pregled i inspekciju dizajna i programskog proizvoda**.

Korisnička dokumentacija daje jasan opis kako da se koristi zadatana aplikacija.

Dokumentacija za održavanje:

- ✓ Prikuplja potrebne informacije o kodu, strukturi i zadacima svakog člana tima
- ✓ Veoma je bitna za lociranje uzroka softverskih grešaka, izmene i korekcije postojećeg softvera.

- ✓ DNEVNIK TESTIRANJA!

Problem kod projekt menadžera

Neki softverski timovi ne shvataju proces testiranja i složenost uloga koje postoje u procesu testiranja.

Nekada projekt menadžer postaje i test menadžer i tako kontroliše ceo tim.

U nekim kompanijama postoji i product menadžer.

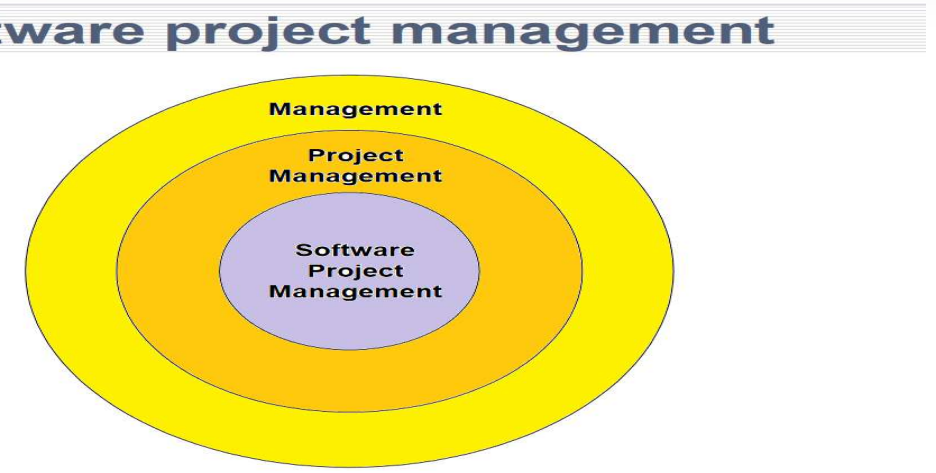
Zbog vremenskih rokova, resursa i samog budžeta, često postoje situacije da nemate dovoljno vremena za testiranje softvera ako projekat kasni.

Problem kod projekt menadžera

- Potencijalni problem -mali broj project menadžera adekvatno tretira informacije dobijene tokom procesa Testiranja Softvera.

Neki od zadataka project menadžera su:

- Planirajanje projekta
- Propisuju pravila
- Određuju rokove
- Raspored obavljanja posla
- Kontrolišu rad



TESTIRANJE SOFTVERA

Dosadašnji pristup problemu testiranja softvera je bio takav da se kroz čitav repertoar međusobno nezavisnih tehnika i strategija testiranja softvera oceni nivo kvaliteta softverskog proizvoda. Takav pristup u literaturi se naziva pasivan pristup.

Pasivan pristup je kao rezultat davao samo konstatataciju da softverski proizvod nije korektan jer su detektovane (otkrivene) greške i to najčešće u kasnijim fazama razvoja softvera t.j. u fazi testiranja dok su one pravljene mnogo ranije. Može se reći da je život softverskih grešaka bio dug od momenta generisanja do momenta otkrivanja.

TESTIRANJE SOFTVERA

Najveće softverske kompanije i grupe svakodnevno objavljuju sve bolje i bolje arhitekture i komponente koje omogućavaju da se napravi odličan softver.

Kompleksan kod se lako generiše brojnim alatima i generatorima koda

U dobrim kompanijama testeri su:

Zaduženi za kontrolu kvaliteta.

Takve kompanije se lako prepoznaju po tome što imaju stabilne softverske proizvode koje koristi veliki broj zadovoljnih korisnika.

TESTIRANJE SOFTVERA

- Faza Optimizacije
- Prevencije grešaka i
- Kontrola kvaliteta

Optimizacija

Šta optimizujemo?

Pod pojmom optimizacijom koda se obično misli na:

- Poboljšanje čitljivosti koda i kultura programiranja

(da neko može da nastavi brzo da radi posle Vas - komentari, dokumentacija)

- Optimizaciju koda radi ubrzanja rada
- Optimizaciju koda radi smanjenja veličine programa

TESTIRANJE SOFTVERA

Testiranje softvera je aktivnost za koju se vezuje negativan prizvuk jer je operativno usmerena na "slabe tačke" procesa razvoja kao i ka kvalitetu samog softverskog proizvoda

Drugi problem je, da skoro nikad nema dovoljno vremena za testiranje softvera

Zato treba da menjamo odnos prema aktivnosti testiranja softvera i vremena koje je planirano za testiranje,

Testiranje softvera u ranim fazama razvoja softvera

ISTORIJAT TESTIRANJA SOFVERA

Testiranje se menjalo kroz istoriju:

- ✓ Testiranje radi pronalaženja bugova (grešaka)
- ✓ Demonstraciono testiranje
- ✓ Destrukciono testiranje
- ✓ Evaluaciono testiranje
- ✓ Preventivno testiranje

Testiranje rad pronalaženja grešaka

- nisu postojali projektni timovi – analitičar, programer, testeri, administratori
- Često je ista osoba i programiral i testirala softver, menjala softver,
- Znači da su testiranje vršili programeri i oni su ispravljali greške (bugove) koje su našli.
- Problem realnog testiranja, jer programer nije realan tester jer on sam vrši implemetaciju softvera.

Demonstraciono testiranje

- Testiranje je bilo proces kojim se demonstriralo da softver radi upravo ono za šta je originalno namenjen.

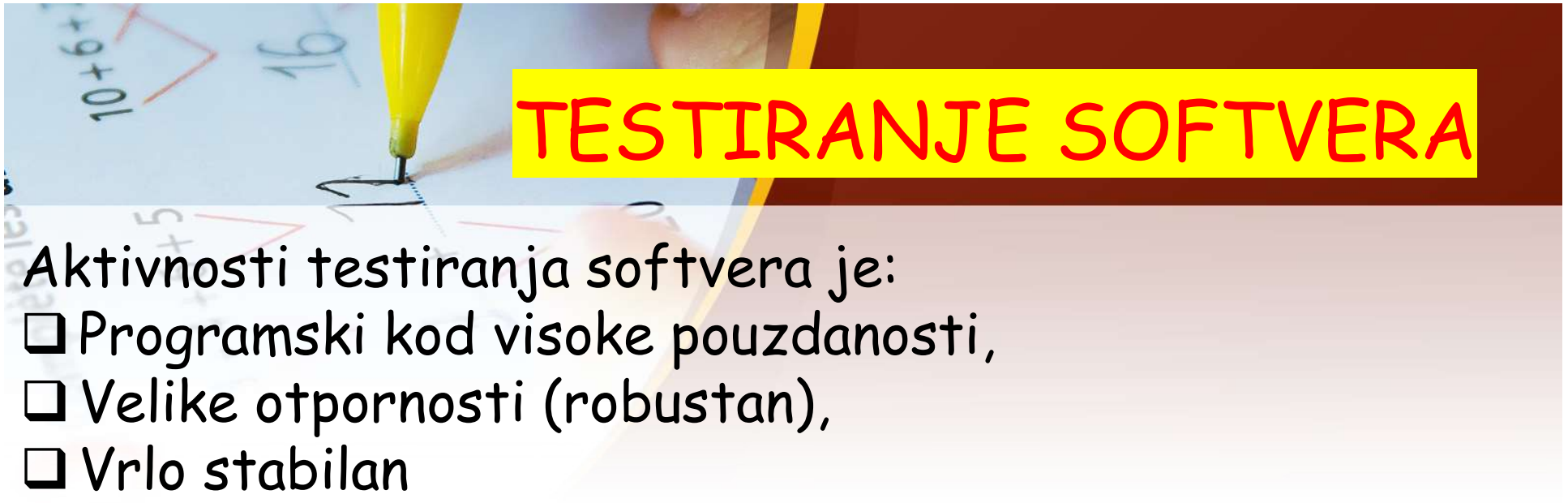
Svako odstupanje je predstavljalo grešku koju je trebalo ispraviti.

- Pronalaženje grešaka je predstavljalo proces koji je bio povezan samo sa funkcionalnosti softvera.

Destrukciono testiranje

- Testiranje je proces otkrivanja grešaka u programu.
- Pronalaženje grešaka je proces otkrivanja gde se greške nalaze u programskom kodu

Testiranje Softvera: Pronalaze se test slučajevi kojim se najlakše dokazuje da stvarno postoje bugovi (greške) u programu.



TESTIRANJE SOFTVERA

Aktivnosti testiranja softvera je:

- Programski kod visoke pouzdanosti,
- Velike otpornosti (robustan),
- Vrlo stabilan
- da potvrdi da softver zadovoljava skoro sve zahteve krajnjeg korisnika ili one koje je obavezno zahtevao.

Zato se aktivnost testiranja softvera smatra destruktivnom prema programskom kodu, mada se na kraju pokazuje da je ta aktivnost vrlo konstruktivna

Evaluaciono testiranje

Testiranje je postalo evaluaciono – proces testiranja se vrši kroz ceo proces razvoja softvera.

Zavisno od faze projekta svaki dokument ili zahtev se proveravao kako bi se što je moguće ranije pronašli.

U agilnim metodama testiranje je istovremeno i preventivno i evaluaciono.

Na početku svake iteracije testeri evaluiraju zahteve pre nego što se počne sa implementacijom kako bi se obezbedilo da se samo validan zahtev preda timu na implementaciju.

Preventivno testiranje

Testiranje softvera je aktivnost ili proces kojim se pokazuje ili demonstrira da program ili sistem izvršava sve predviđene funkcije korektno.

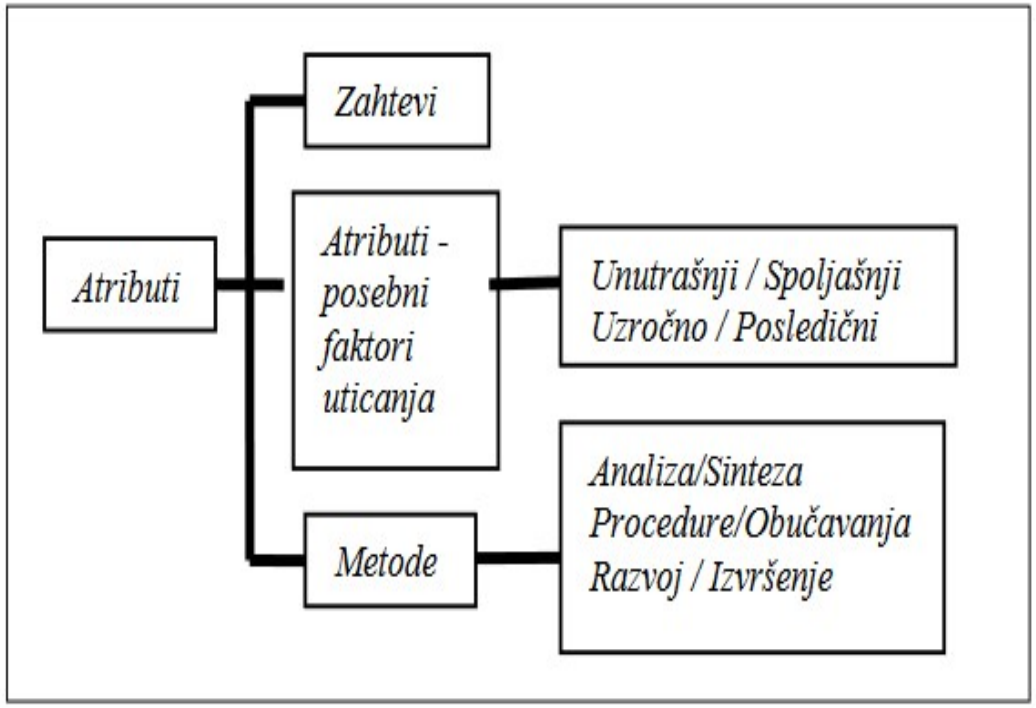
Za razliku od evaluacionog testiranja kojim se testira svaka komponenta pošto se napravi, u prevencionom testiranju se testovi prave i pre nego što se komponente naprave.

Stav prema kvalitetu softvera je prevencija, mnogo je bolje izbeći probleme nego ih ispravljati.

Troškovi proizvodnje razvoja softvera se smanjuju sa ranim otkrivanjem i ispravkom grešaka. ODNOSNO:

Preventivni troškovi sastoje se od akcija koje se preduzimaju kako bi se sprečili defekti.

KVALITET SOFTVERA



OPŠTA KLASIFIKACIJA ATRIBUTA KVALITETA

KVALITET SOFTVERA

Ključni aspekti kvaliteta softvera odnose se na:
- KVALITET PROCESA RAZVOJA
- STRUKTURNI KVALITET SOFTVERA
- FUNKCIONALNI KVALITET SOFTVERA

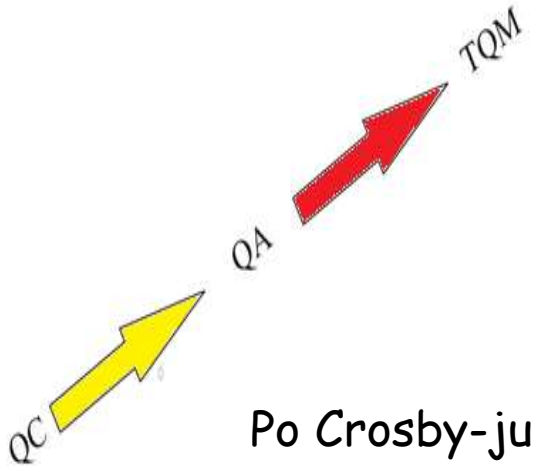
PROBLEM kod KVALITETA SOFTVERA:

Reputacije kompanije

Gubitka značajnih klijenata

KVALITET SOFTVERA

- ✓ Kontrola kvalitete – QC (**Q**uality **C**ontrol)
- ✓ Osiguranje kvaliteta – QA (**Q**uality **A**ssurance)
- ✓ Potpuno upravljanje kvalitetom – TQM (**T**otal **Q**uality **M**anagement),



Po Crosby-ju kvalitet je "prilagođavanje zahtevima

KVALITET SOFTVERA

Testiranje softvera može se posmatrati kao jedan od metoda koji se koriste za kontrolu kvaliteta softvera.

Kontrola kvaliteta je process koji se sprovodi na kraju izrade softvera i kontrolom se upoređuje kvalitet urađenog softvera sa polaznom specifikacijama.

Greške koje se prave tokom rada su normalna i svakodnevna stvar i nisu direktno vezane za izradu softvera.

Greške se dešavaju u svim oblastima rada i uzrokovane su kako ljudskim, tako i drugim faktorima.

Imajući u vidu da su greške realna pojava, u svim oblastima rada se uvodi sistem kontrole kvaliteta sotfvera kojim se greške identifikuju i otklanjaju pre nego što izazovu veće probleme u radu.

KVALITET SOFTVERA

Kvalitet softvera podrazumeva usklađenost sa operativnim i efektivnim potrebama",

Kvalitet softvera se može definisati na različite načine :

- ✓ Usaglašenost sa zahtevima i potrebama korisnika
- ✓ Dobri atributi proizvoda (brzina rada, pokretanja...)
- ✓ Lakoća održavanja i promena u samom softveru
- ✓ Usaglašenost sa standardima
- ✓ Rad sa ogromnom količinom podataka

KVALITET SOFTVERA

" Upravljanje kvalitetom softvera u najjednostavnijoj mogućoj formi je jedan proces kroz koji možemo definisati kvalitet softvera, ali i planirati, i i proveriti koliko je softver u razvoju u skladu sa inicijalnim zahtevima. "

KVALITET SOFTVERA

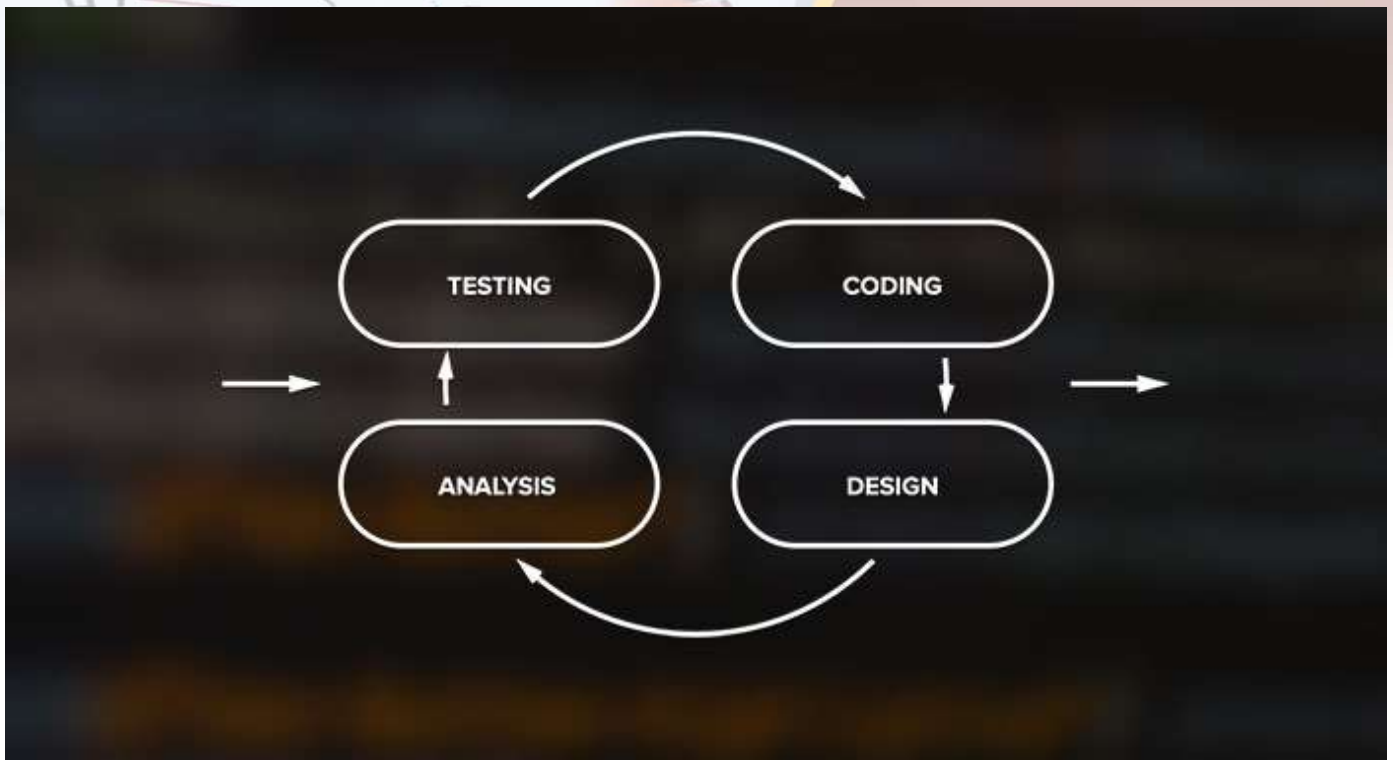
Šest generalnih karateristika kvalitetnog softvera:

- ✓ 1. **funkcionalnost** - postojanost funkcija softvera ili njenih naznačenih osobina
- ✓ 2. **pouzdanost** - sposobnost softvera da održava nivo performansi pod navedenim uslovima, u toku navedenog vremenskog perioda
- ✓ 3. **upotrebljivost** - pogodnost za korišćenje softvera, odnosno kompleksnost za krajnjeg korisnika
- ✓ 4. **efikasnost** - odnos između nivoa performansi softvera i količine resursa koji se koriste, pod navedenim uslovima
- ✓ 5. **održivost** - sposobnost softvera da jednostavno podrži buduće modifikacije
- ✓ 6. **prenosivost** - sposobnost softvera da bude prenesen iz jednog u drugo radno okruženje

TESTIRANJE SOFVERA

- Kao što se vidi od strategije rešavanja zavisi kako će teći projekat

TESTIRANJE SOFTVERA



TESTIRANJE SOFTVERA

ZAŠTO TESTIRAMO PROGRAM?

- Da bi lakše pokazali korisnicima da softver zadovoljava željene zahteve i funkcionalnosti.
  - Koristimo testove validacije
  - Tada očekujemo od sistema da za skup test primera radi korektno i da daje isti rezultat
  - Uspešni testovi pokazuju da sistem pruža željene funkcionalnosti.

TESTIRANJE SOFTVERA

ZAŠTO TESTIRAMO SOFTVER?

- ✓ Da bi lakse primetili ili locirali greške i probleme u samom programu.
- ✓ Koristimo defect testiranje
- ✓ Test primeri korišćeni pri ovakvom testiranju su pravljeni kako bi izazvali pojavljivanje grešaka (ukoliko one postoje) i pokazali eventualno neželjeno ili neočekivano ponašanja sistema.

NEKI KORACI U PROCESU TESTIRANJA PROGRAMA

- ✓ Ponovite testove za iste ulazne podatke ili niz ulaznih podataka veliki broj puta.
- ✓ Pronaći ako je moguće da softver generiše neispravane izlazne vrednosti.
- ✓ Treba postići da rezultati izračunavanja budu preveliki ili premali.

Razvoj softvera je podeljen u nekoliko faza

- Analiza i definisanje zahteva

U ovoj fazi se utvrđuju zahtevi koje sistem treba da zadovolji,što se postiže saradnjom između razvojnog tima i korisnika sistema. Rezulat ove faze je lista korisničkih zahteva.

- Projektovanje sistema (dizajn)

U ovoj fazi se generiše projekat sistema koji daje plan rešenja odnosno arhitekturu sistema.

- Implementacija softvera

Faza u kojoj developeri pišu kod softvera.

Razvoj softvera je podeljen u nekoliko faza

- Testiranje softvera

Faza u kojoj se otkrivaju i ispravljaju greške u sistemu.

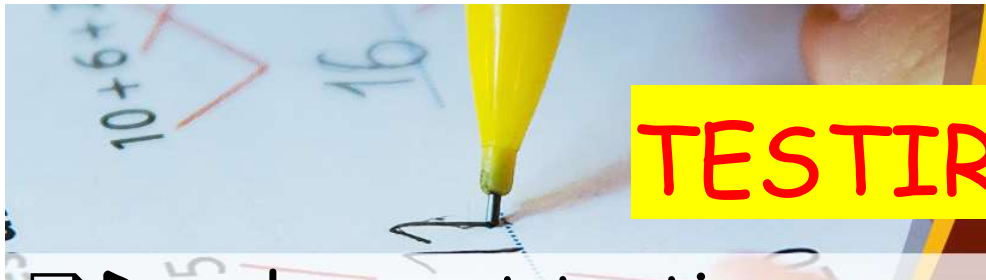
- Isporuka sistema

Sistem se isporučuje naručiocu, softver se instalira u radnom okruženju i vrši se obuka korisnika.

- Održavanje

Dugotrajna faza u kojoj se ispravljaju greške u sistemu, otkrivene nakon njegove isporuke.

Radi se na daljnjem unapređenju delova sistema prema zahevima korisnika.



TESTIRANJE SOFTVERA

- Development testing

Testovi u toku razvoja

- Test-driven development

Test driven development (TDD) je proces razvoja softvera nastao kao deo Ekstremnog programiranja, da bi ga kasnije usvojile sve agilne metodologije

- Release testing
- User testing

TESTIRANJE SOFTVERA

- Development testing – to je testiranje softvera u fazama razvoja kako bi greške bile otkrivene i otklonjene iz programa.
- Release testing – testiranje čitavog sistema od strane posebnog tima za testiranje pre nego što se sistem konačno pusti u produkciju.
- User testing – kada korisnici ili potencijalni korisnici testiraju sistem u svom okruženju.

DEVELOPMENT TESTING

DEVELOPMENT TESTING Predstavlja sva testiranja izvršena od strane razvojnog tima na sistemu.

Dele se na:

- Unit tests

Testiranje jedinica koda, gde je svaka jedinica koda zasebno testirana, pri čemu jedinica koda može biti funkcija, metod, klasa, operacija, struktura podataka.

DEVELOPMENT TESTING

- Component testing

Testiranje komponenata, pri čemu komponenta predstavlja integrisane jedinice koda ko je zajedno pružaju neku funkcionalnost, tako da se ovo testiranje uglavnom (a ne uvek) bavi testiranjem interfejsa te komponente.

System testing

❑System testing

Testiranje sistema kao celine.

Sistemsko testiranje se zasniva na testiranju sistema kao celine, nakon što je završena kompletna integracija.

Definisano je funkcionalnim zahtevima i specifikacijama sistema.

Po nekim statistikama tvrdi se da najviše grešaka nastane u fazi implementacije.

Tada je neophodno obavljati testiranje sistema tokom i nakon implementacije.

Na ovaj način se smanjuje mogućnost grešaka.

Faze u testiranju softvera

- Prva faza (eng. Red Phase)
- Druga faza (eng. Green Phase)
- Treća faza (eng. RefactorPhase)

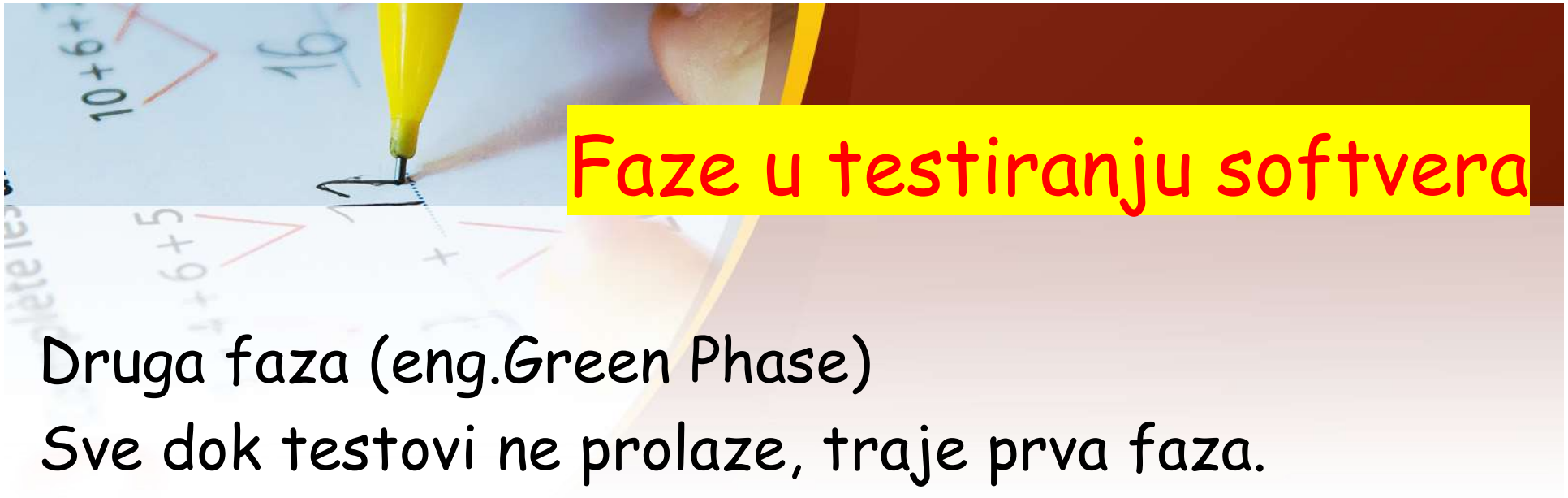
Faze u testiranju softvera

- Prva faza (eng. Red Phase)

To je faza koja programerima koji tek počinju sa primenom ove metode stvara najviše problema.

Ona podrazumeva pisanje testova, za klase i metode koje još nisu implementirane.

Posledica je da napisani testovi u ovoj fazi možda neće proći.



Faze u testiranju softvera

Druga faza (eng.Green Phase)

Sve dok testovi ne prolaze, traje prva faza.

Da bi prešli u drugu fazu potrebno je implementirati kod potreban za testove da bi prošli.

Dobra praksa da se napiše najjednostavniji kod da bi test prošao.

Faze u testiranju softvera

Treća faza (eng. Refactor Phase)

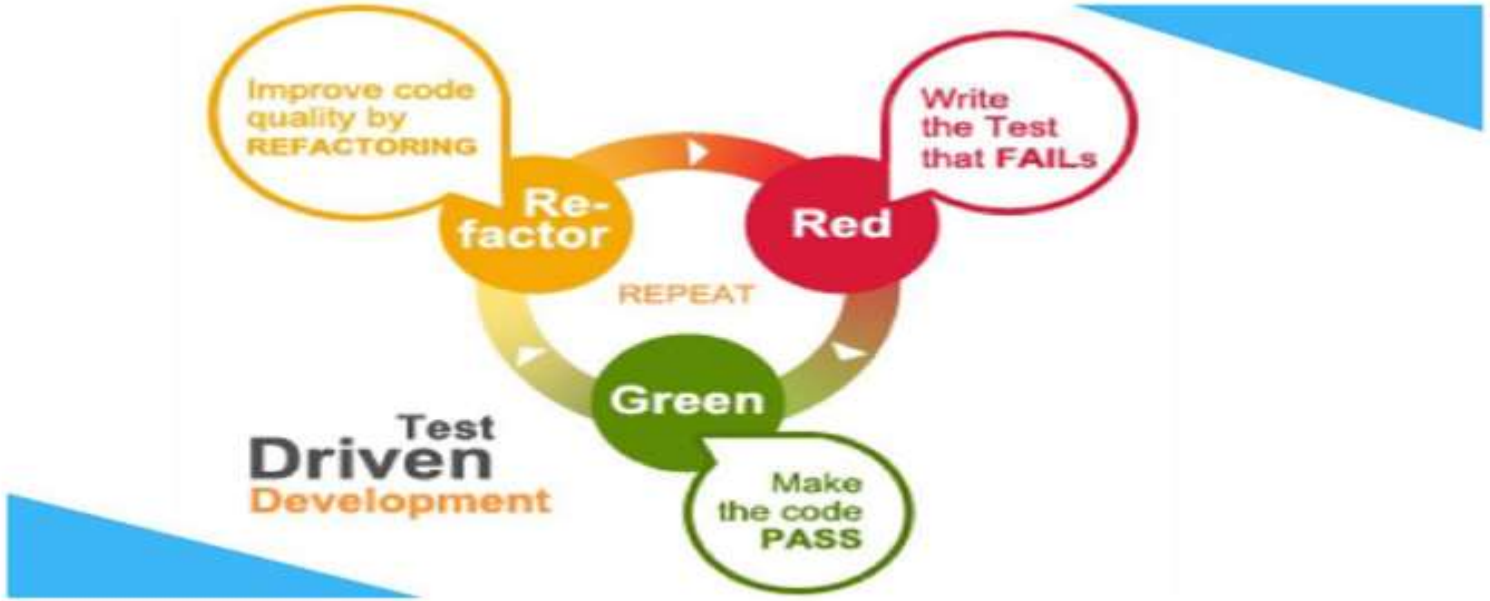
U ovoj fazi se vodi računa o kvalitetitu koda koja je napisan, jednostavnosti održavanja

Refaktorizacija podrazumeva menjanje

Testovi u ovoj fazi služe da bi developeri videli da promene na kodu ne menjaju dosadašnju prolaznost testova.

Dok testovi prolaze znamo da je kod koji je napisan u redu.

Faze u testiranju softvera



Faze u testiranju softvera

- Pišu se testovi koje prolaze.
- Prave se jednostavniji programi i korisnički interfejsi (eng. Application Interface -API).
- Programer koji piše program i pravi korisnički interfejs je prva osoba koja ih koristi, te na osnovu toga može da vidi da li imaju smisla, nerealno testiranje ili je potrebno poboljšanje.

Faze u testiranju softvera

- U finalnim verzijama skoro da nema ne korišćenog koda.
- Testovi koji se pišu omogućavaju da sve (eventualne) promene u kodu ili dodavanje novih metoda neće narušiti već postojeće funkcionalnosti.
- Ova metoda ima vrlo malo defekata.

Ako se uoči neki problem ili bug, kod se ispravlja i piše se novi test za taj bug da bi se omogućilo da se taj problem neće više pojavljivati.