

TESTIRANJE SOFTVERA



A close-up photograph of a yellow pen tip pointing at a math problem on a piece of paper. The problem is $10 + 6 =$ followed by a large red checkmark. The number 16 is written next to it. The background is a dark red gradient.

Testiranje softvera

Kada bi se iz ŽIVOTNOG ciklusa softvera nestalo Testiranje softvera, krajnji proizvod bi imao puno više nepravilnosti i grešaka što bi naravno rezultovalo velikim nezadovoljstvom samog korisnika, pa je upravo jedan od razloga što TESTIRANJE ima veliku primenu (svrhu) u procesu razvoja softvera.

Testiranje je kombinacija višestrukih aktivnosti životnog ciklusa softvera povezanih sa planiranjem, dizajniranjem i procenom softverskog proizvoda kako bi se što pre otkrile greške i utvrdilo da li softver ispunjava navedene zahteve ili ne.



Testiranje softvera

Jedna od Definicija za Testiranje softvera je:

“Testiranje je proces ocenjivanja celokupnog sistema ili njegovog dela tj. komponente sa namerom da se utvrdi da li su definisani korisnički zahtevi zadovoljeni ili nisu.”

- ❑ Testiranje softvera takođe uključuje implementaciju modula delova sistema za procenu svih (ili nekih) osobina kroz testiranje softvera.



Testiranje softvera

Testiranje softvera predstavlja važnu osobinu za kvalitetan softverski proizvoda koji zadovoljava potrebe, očekivanja i zahteve samog korisnika.

Testiranje treba da započne još u ranoj fazi razvoja softvera i da se obavlja veoma često.

Treba da se na neki način integriše razvoj aplikacije i životni ciklus testiranja softvera.

Treba formalizovati metodologiju testiranja- a to znači treba sve testirati na identičan način i tada se može očekivati da će se dobiti uniformisani rezultat.



Testiranje softvera

Prema istraživanjima pojedinih autora, najviše grešaka nastane u fazi implementacije, čak 50%, zatim 40% grešaka se dogodi u fazi dizajna, dok samo 10% u fazi specifikacije.

Po velikom broju autora i članaka u naučnim časopisima najviše grešaka dogodi ce u fazi implementacije.

Sugestija/Pravilo

Vrlo važno je Testirati i samo Testirati u svakoj fazi razvoja softvera.



Testiranje softvera

"Testiranje softvera" je mnogo više od običnog traženja grešaka (bugova)."

Kontrolu kvaliteta softvera vrše i testeri.

Tester je u velikoj meri zaslušan za zadovoljstvo klijenta i krajnjeg korisnika

Zadatak testera je da pronade što veći broj bugova koje će zatim programmer ili programerski tim ispraviti.



Testiranje softvera

Bitno je uzeti sledeće aspekte u obzir, koje bi mogle označiti kraj procesa testiranja (NAPOMENA: Nikad nema kraju testiranju softvera):

- Potrebno je ispoštovati **rokove testiranja**,
- Završetak izvršavanja **test slučaja**
- Dovršavanje funkcionalnog i kodnog testiranja(do određene tačke/po specifikaciji)
- Stopa grešaka pada ispod određenog nivoa i nisu identifikovane greške visokog prioriteta
- **Odluka** uprave ili menadžera projekta.
- Vremenski rokovi za implementaciju softvera su istekli



Testiranje softvera

Preduslov da bi programer mogao da preda kod na glavnu razvojnu granu je da se njegov kod uspešno bilduje.

Ako određeni zahtev može biti izvršen i dobijen je zadovoljavajući rezultat, test slučaj je u redu.

Ako su test slučajevi odabrani tako da je svakom zahtevu dodeljen bar jedan test slučaj, program se smatra testiranim i trebalo bi da radi kada svi test slučajevi pokažu zadovoljavajuće rezultate.



Testiranje softvera

Testiranje softvera se može podeliti na :

❖ Dinamičko

❖ Statičko

Podela se vrši na osnovu toga da li se u toku testiranja kod izvršava ili ne.

Statičko testiranje je testiranje bez izvršavanja koda.

Ono se obično vrši kroz razne vrste pregleda i revizija.

Predmet tih pregleda i revizija može biti dokumentacija ili sam kod.



STATIČKO testiranje softvera

Statičko testiranje se svodi na testiranje softvera bez njegovog izvršavanja i pokretanja, bilo ručno bilo putem alata. Moguće je da se izvršava vrlo rano u samom razvoju softvera.

Statičko testiranje nije alternativa dinamičkom testiranju. Ako koristite obe vrste testiranja možete pronaći različite greške.

Statičko testiranje je tehnika koja nam omogućava da prepoznamo nedostatke u softveru, a da ovaj softver zapravo ne pokrećemo.

Statičkim testiranjem možemo pronaći greške koje ne možemo pronaći kod dinamičkog testiranja.

U literature se statičko testiranje može pronaći pod nazivom statična analiza.



Statičko testiranje

Statičko testiranje je skup tehnika kojim se poboljšava kvaliteta softvera, kao i efikasnost i produktivnost samog procesa razvoja softvera.

Ovaj tip testiranja se bazira na procesu statičkog pregleda (review), sa ciljem da se defekti pronađu što ranije u procesu razvoja softvera.

CILJ: Svako testiranje tako i statičkog je detektovanje što većeg broja grešaka.



Statičko testiranje

- je testiranje softvera bez njegovog pokretanja ili izvršavanja. Statičko testiranje se može vršiti na kodu, dizajnu, modelima, funkcionalnim zahtevima, specifikaciji zahteva i arhitekturi samog softvera

Tipovi statičkog testiranja su:

- ✓ Neformalni pregled
- ✓ Formalni pregled
- ✓ Walkthrough
- ✓ Tehnički pregled
- ✓ Inspekcija



Neformalni pregled

Pregled sam po sebi može biti formalan ili neformalan.

Pregled je generalni termin koji se koristi za sve tehnike manuelnog statičkog testiranja

- u praksi se najčešće koristi *review* ili *inspection*
- u pregledu najčešće učestvuje više ljudi
- *peer review*

Ako se govori o neformalnom pregledu tada se najčešće radi o prezentaciji dokumenata projektnom timu (specifikacije zahteva, detaljni dizajn, tehnička /e specifikacije).

- Svrha pregleda je davanje informacija svim članovima tima o napretku projekta.



Neformalni i formalni pregled

Neke od Faza tokom formalnog pregleda su planiranje, kick-off, priprema, formalni sastanak, korekcije i follow-up. Za razliku od neformalnog pregleda, formalin pregledi se obavezno moraju dokumentovati u obliku zapisnika.

- **Prednosti pregleda**
- Poboljšava se dalji proces razvoja softvera,
- Ideja je da ako je to moguće (a jeste) da se ti nedostaci otkriju, što pre
 - ✓ u razvoj rešenja uključeno više ljudi a samim time je kvalitetnije rešenje
 - učesnici uče jedni od drugih
 - ✓ formalizacija i dokumentovanje rešenja
 - ✓ autor mora da iznese rešenje tako da drugim učesnicima bude čitljivo i razumljivo, kao i da obrazloži odluke donete pri razvoju
 - ✓ odgovornost se raspoređuje na sve učesnike
- Iako troši resurse, pregled je neizostavan u razvoju jer značajno smanjuje broj kasnijih nedostataka i troškove otklanjanja



Neformalni i formalni pregled

- Na generalnom nivou

- ☐ na nivou projekta se mora doneti odluka koji dokumenti se pregledaju
- ☐ mora se u plan projekta uvrstiti i dinamika i troškovi tih pregleda

- Na nivou jednog pregleda

- ☐ pripremiti dokumente koji su predmet pregleda
- ☐ izabrati učesnike u pregledu
- ☐ obezbediti da je dokument spreman za pregled
 - ☐ da je završen na nivou predviđenom za tu fazu projekta
- ☐ odrediti mesto i vreme sastanka i druge detalje potrebne za održavanje pregleda



TEHNIČKI pregled

Tehnički pregled je fokusiran na tehničke aspekte sistema

- ☐ usklađenost sa specifikacijom
- ☐ ispunjenost svrhe zbog koje se razvija
- ☐ usklađenost sa standardima
- Osnov za pregled je isključivo zvanična specifikacija zahteva
- Recenzenti poseduju tehničko znanje
- Procedura može biti više ili manje formalno sprovedena
 - ☐ kod formalnog tehničkog pregleda, svi rezultati sve formalno evidentiraju



Inspekcija softvera

- Uključuje ljude koji proveravaju source code aplikacije sa ciljem utvrđivanja anomalija i defekata.
- ✓ Ne zahteva izvršenje aplikacije, pa se može koristiti i pre implementacije.
- ✓ Može se primeniti na bilo koji element sistema - zahteve, dizajn konfiguracione podatke,.....
- ✓ Pokazalo se da su vrlo efikasna tehnika za utvrđivanje programskih grešaka



Prednosti inspekcija

- U toku testiranja, greške mogu maskirati (sakriti) druge greške. Zbog toga što je inspekcija statički proces, druge greške koje se pojave ne mogu uticati ili sakriti greške.
- Nekompletne verzije sistema se mogu proveravati bez dodatnih troškova. Ako je aplikacija nekompletna potrebno je razviti specijalizovan test koji će testirati dostupne delove
- Osim pronalaženja programskih defekata, inspekcija može posmatrati i šire attribute kvaliteta aplikacije, kao što su poštovanje standarda, portabilnost i mogućnost održavanja.



PROVERA SOFTVERA (SOFTWARE INSPECTIONS)

Odnosi se na „statičku“ proveru, pre implementacije kompletnog softvera, za razliku od testiranja koje podrazumeva „dinamičku“ proveru prilikom izvršavanja.

Uključuje ljude koji mogu da pregledaju izvorni kod kako bi našli neke anomalije u softveru, greške i delove koda koji bi mogli izazvati neželjeno ponašanje prilikom samog izvršavanja softvera.



PREDNOSTI inspekcija

Osim pronalaženja programskih defekata, inspekcija može posmatrati i šire attribute kvaliteta aplikacije, kao

što su :

- ✓ poštovanje standarda,
- ✓ portabilnost i
- ✓ mogućnost održavanja.



PROVERA SOFTVERA (SOFTWARE INSPECTIONS)

- Može biti primenjeno na bilo koji segment sistema (na primer na zahteve, na dizajn softvera)
- Koristi se znanje o čitavom sistemu, UML modeli, šeme baze podataka, domen aplikacije.
- Ne može zameniti testiranje softvera.



PROVERA SOFTVERA (SOFTWARE INSPECTIONS)

Tokom testiranja softvera greške koje se pronalaze mogu da "sakriju" druge (velike) greške (bugove). Kako je inspekcija statički proces, često se ističe da možda nema razloga za brigu o „vezanim“ greškama.

(UVEK SE MOGU NAĆI bugovi i nekad jedan bug kad se ispavi prouzrokuje drugi bag, to su takozvani vezani bagovi).

Može se primeniti na takozvane nepotpune ili nedovršene sisteme, pre konačne implementacije softvera.



PROVERA SOFTVERA (SOFTWARE INSPECTIONS)

Pored pronalaženja grešaka, tokom inspekcije mogu biti analizirane i druge važne karakteristike sistema, kao što je na primer saglasnost(cookies), prenosivosti, održivost samog softvera.

Inspekcija ne može da proveriti nefunkcionalne karakteristike sistema kao sto su na primer performance (nekog sistema, uređaja), ..



Faze pregleda

- Postoje različiti tipovi pregleda, obično postoje sledeće faze (implicitno ili eksplicitno izražene)
 - ✓ Planiranje
 - ✓ Informisanje
 - ✓ Samostalna priprema
 - ✓ Sastanak
 - ✓ Ispravke
 - ✓ Revizija



Walkthrough

Walkthrough spada u neformalni proces. Autor/i sa svim učesnicima sastanka prolazi kroz ceo dokument. Ideja je da se da se prikupe dodatne povratne informacije.

Cilj walkthrough procesa je:

- ✓ Prezentacija dokumenta da bi se dobile povratne informacije o dokumentaciji
- ✓ Prenošenje znanja
- ✓ Evaluacija sadržaja dokumenta
- ✓ Diskusija o predloženim rešenjima



Revizija

- Nakon ispravki neophodno je izvršiti reviziju nove verzije dokumenta.
- Mora se obaviti ŠTO PRE novi pregled
 - procedura drugog pregleda može biti nešto kraća sa fokusom samo na modifikovane delove (koji su promenjeni)
- Potrebno je evaluirati sam proces pregleda kako bi naredni pregled ispravio nedostatke u proceduri
 - Treba da se identifikuju novi faktori koji mogu dovesti do toga da neki naredni pregled bude efikan.



Uloge učesnika

Mogu se identifikovati nekoliko klasa učesnika pregleda:

- Menadžer
- Moderator
- Autor
- Recenzenti
- Beležnik
 - zadužen da evidentira informacije iz pregleda (probleme, stavove, odluke, zaključke)



Uloge učesnika

U praksi se može videti da isti ljudi preuzimaju više različitih tipova uloga

- ❖ npr. menadžer je istovremeno i moderator, ali i recenzent koji aktivno učestvuje u donošenju odluka u pregledu
- ❖ takođe, za vođenje beležaka može biti zadužen neko od članova tima
 - Veoma je bitno da svi imaju uvid u zaključke pregleda



Tipovi pregleda

Postoje dve grupe pregleda i to:

- pregledi koji su namenjeni analizi karakteristika samog projekta
 - da li su zahtevi (funkcionalni i nefunkcionalni) ispunjeni
- pregledi koji su namenjeni analizi samog procesa razvoja projekta
 - Na primer analiziraju se kvalitet upravljanja projektom, ispunjenost plana projekta i drugo.



Izbor tipa pregleda

To zavisi od konkretnog slučaja

- Preporuke i ideje:

- ❖ Formalno dokumentovani rezultati pregleda trebaju biti
- ❖ Koliko je komplikovano organizovati grupni lični sastanak sa učesnicima
- ❖ Koliko tehničkog znanja je potrebno da učesnici poseduju
- ❖ Kakav je odnos resursa potrebnih za pripremu pregleda i koristi od rezultata pregleda
- ❖ Da li je predmet pregleda formalno specificiran da bi mogao biti analiziran programski
- ❖ Koliko je menadžment projekta obezbedio resursa za preglede



STATIČKO testiranje softvera

Vrste grešaka koje je lakše pronaći tokom statičkog testa:

- ✓ Odstupanja od standarda
- ✓ Loš kod
- ✓ Greške u dizajnu
- ✓ Probelemi sa specifikacijom

IDEJA: Koristiti statičko testiranje da biste rano pronašli greške (bagove).



STATIČKO *testiranje softvera*

Statičko testiranje obuhvata testiranje softverskog zahteva. Planirane aplikativne funkcionalnosti se analiziraju na primer da li postoje neke nejasnoće, neka dvosmislenosti, kao i da li se uopšte može testirati softver.

Svaki zahtevani detalj koji može dovesti do pogrešnog tumačenja mora biti nedvosmisleno razjašnjen.

Statičko testiranje podrazumeva verifikaciju test objekta analizom njegovog sadržaja.



Testiranje softvera

Testiranje softver može da se izvrši na različitim nivoima.

U zavisnosti od samog sistema, kao i od raspoloživih resursa, bitno je odabrati odgovarajuću metodologiju testiranja softvera.

-Metodologiju testiranja softvera -

Softverski testeri igraju ključnu ulogu u IT industriji jer je njihov zadatak da utvrde da li softverski proizvod odgovara unapred postavljenim zahtevima.



Testiranje softvera

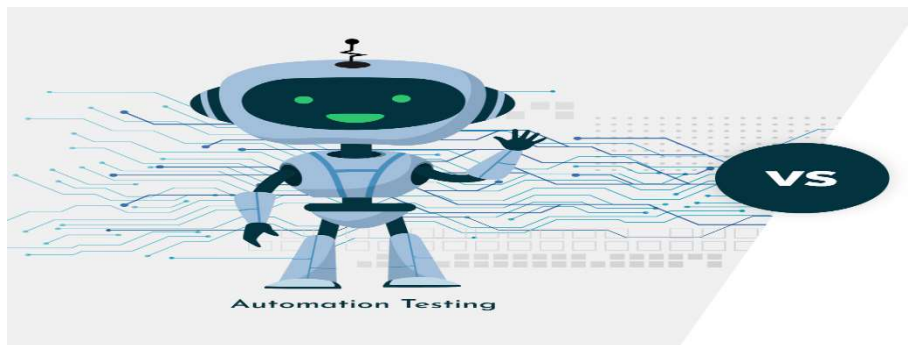
Tester je član tima koji mora da se postavi u ulogu samog klijenta ili ciljne grupe korisnika, ali isto tako da se postavi u ulogu programera koji kreira neku softversku aplikaciju.

Testiranje u bilo kojoj od faza razvojnog životnog ciklusa softvera se obavlja na specifičan način, na primer u fazi prikupljanja se razmatraju analize i verifikacija zahteva testiranja.



Testiranje softvera

Testiranje može da se radi manuelno ili automatski uz pomoć alata i tehnologija za pisanje automatskih testova. Zbog toga postoji podjela na automatsko i manuelno testiranje. manuelno testiranje obavlja čovek, dok automatsko izvršava program.



VS





Testiranje softvera

Da bi garantovao kompletnost testiranja, tester često sledi pisani plan ispitivanja koji ih vodi kroz niz važnih test slučajeva.

Svrha manuelnog testiranja je da pokrije što više scenarija koje **nije moguće automatizovati** i na taj način da se dobije na kvalitetu samog softvera.



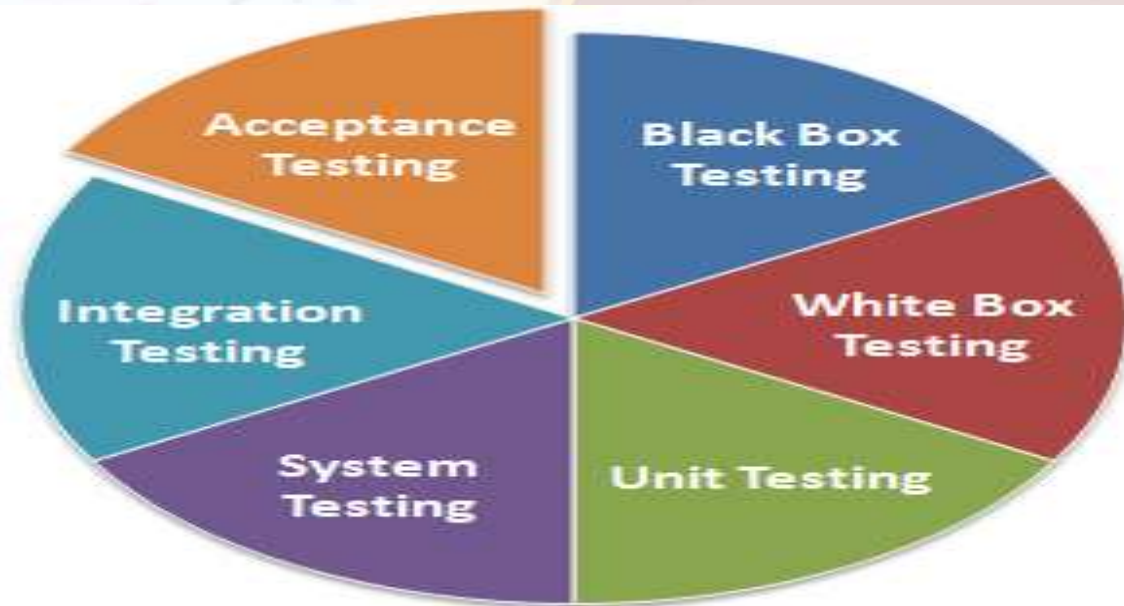
Ručno (manuelno) testiranje softvera

Glavni koncept ručnog testiranja je da aplikacija radi bez grešaka.

Ovaj način testiranja softvera predstavlja osnovnu tehniku svih tipova testiranja i pomaže u pronalaženju kritičnih grešaka u softverskoj aplikaciji.

Svrha manualnog testiranja jeste da se osigura da aplikacija nema grešaka i da radi u skladu sa specificiranim funkcionalnim zahtevima

Testiranje softvera





Ručno (manuelno) testiranje softvera

Manuelno testiranje je takav način testiranja gde se ne koristi ni jedan automatizovan alat ili bilo koja skripta. U ovom načinu testiranja ispitivač preuzima ulogu krajnjeg korisnika i testira softver sve dok ne indentifikuje neko neočekivano ponašanje ili grešku. Prilikom testiranja, obično su na raspolaganju ograničeni resursi, čime se onemogućava efikasno izvršavanje manualnih testova

Postoje različite faze manualnog testiranja :

- Intergraciono testiranje i
- Prihvatljivo testiranje

Postoje i različite tehnike analize

- razlikuju se po intenzitetu, formalnosti, potrebnim resursima i ciljevima
- ne postoji jedinstvena terminologija za ove različite tehnike analize

Tester koristi plan testiranja, testove ili scenarija za testiranje softvera i time oni osiguravaju da proces testiranja bude kompletan.



Ručno (manuelno) testiranje softvera

Manuelno testiranje takođe uključuje i istraživačko testiranje za cilj da treba istražiti softver kako bise indentifikovala greška u njemu.

Manuelno testiranje zahteva angažovanje većeg broja ljudi unutar test timova za izvršenje testa.

Za manuelan način testiranja nije potrebno poznavanje nijednog alata za testiranje



Istraživačko testiranje softvera

Istraživačko testiranje (exploratory testing), gde bez unapred definisanog postupka tester koristi svoju intuiciju i iskustvo u pronalaženju grešaka

ILI

Istraživačko testiranje je vežbanje testiranja, u kojem se testerima dodeljuje neprecizno definisan zadatak, uz korišćenje softvera koji se testira.

Istraživačko testiranje nije nasumično, ali nije ni skriptovano kao manuelno testiranje.



Istraživačko *testiranje softvera*

Istraživačko testiranje je metoda testiranja u kojoj kao i u skriptovanim testovima postoje test procedure ali ne moraju striktno da se prate.

Tokom ove metode testiranja tester i otkrivaju i proveravaju alternativne pravce korišćenja sistema, što znači da se u istraživačkom testiranju objedinjene aktivnosti identifikacije, dizajna i izvršavanja test slučajeva.



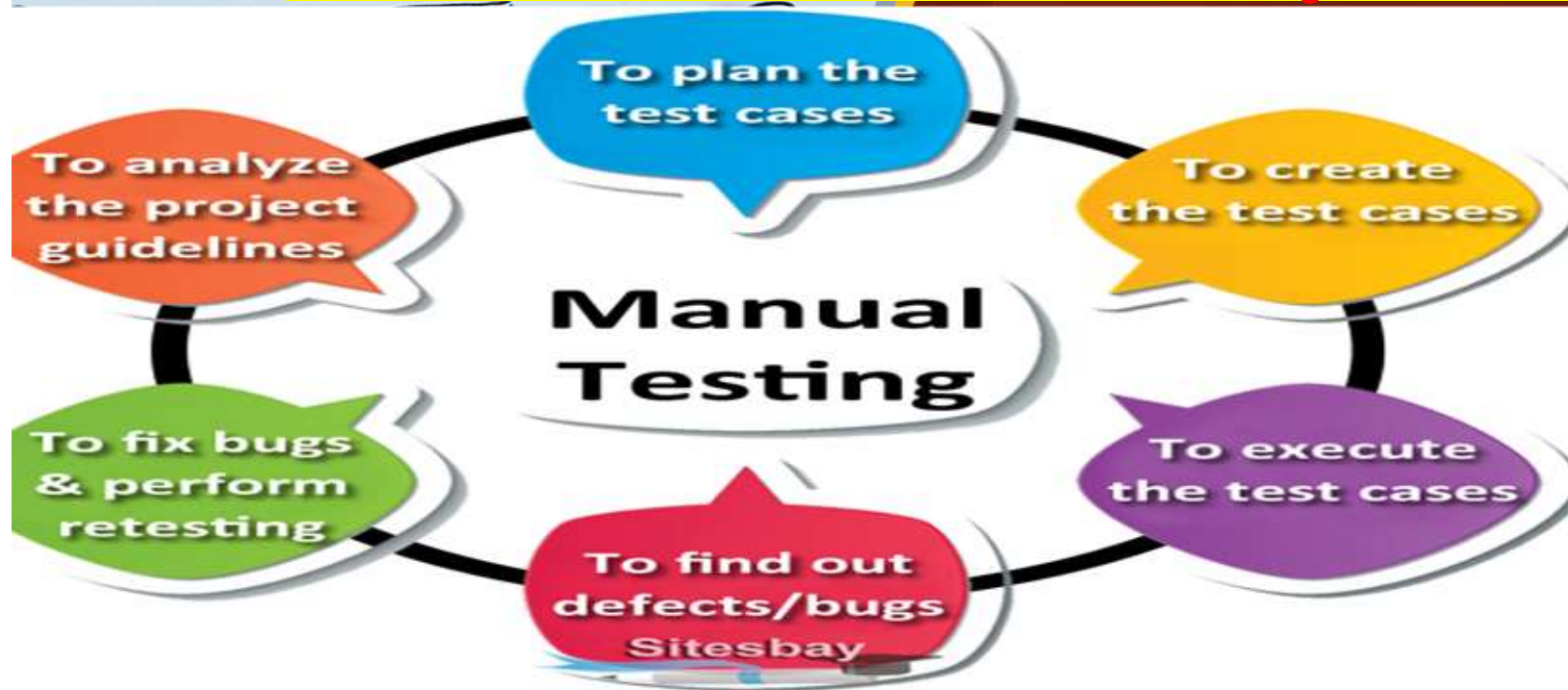
Istraživačko *testiranje softvera*

Jedna od mana ovog testiranja je da je ovo testiranje neregulisano i da nema realnog plana da se odredi koje delove softvera treba da se testira.

Neki od problema koji se javljaju jeste da se neke greške ili defekti ne mogu reprodukovati.

Ovo je neformalni proces testiranja.

Ručno (manuelno) testiranje softvera





Ručno (manuelno) testiranje softvera

Ručno (manuelno) testiranje softvera ponekad skraćuje ukupno vreme potrebno za proveru softvera jer je potrebno da probleme koji se javljaju razumemo na konceptualnom nivou.

Postoje radovi gde neki autori tvrde da se u velikoj kompaniji poput Microsoft pronade 10-20 % grešaka u njihovim projektima (a potom uspešno otkloni) na ovakvom vidu testiranja softvera.



Ručno (manuelno) testiranje softvera

Oni mogu da budu neformalni, gde tester proverava da li je implementirana funkcionalnost u skladu sa dokumentacijom, ili formalni, gde se test sprovodi po unapred definisanom setu koraka, koji simuliraju ponašanje korisnika.



Ručno (manuelno) testiranje softvera

Manuelno testiranje ne zahteva poznavanje programskih jezika i predstavlja samostalnu celinu.

Manuelno testiranje takođe može biti i uvod u automatsko testiranje, jer se najčešće manualni testeri odlučuju da kasnije uče i automatsko testiranje, koje zahteva osnovno poznavanje bar jednog programskog jezika.



Ručno (manuelno) testiranje softvera

- ✓ Manuelno testiranje je proces otkrivanja nedostataka ili grešaka u softverskom programu.
- ✓ Kod ovakvog tipa testiranja softvera, tester ručno izvršava testne slučajeve, bez upotrebe automatizovanih alata.
- ✓ Ručno testiranje je takoreći najprimitivniji oblik testiranja softvera.

Koristimo ga za pronalaženje grešaka (bagova) u aplikacijama i softverskim sistemima.



Ručno (manuelno) testiranje softvera

Odnosi se na analiziranje dokumenata u projektu od strane učesnika

- dokument je najčešće sam kod, ali može biti i specifikacija zahteva, model sistema, model podataka, plan aktivnosti, ...

Različite tehnike analize

- razlikuju se po intenzitetu, formalnosti, potrebnim resursima i ciljevima
- ne postoji jedinstvena terminologija za ove različite tehnike analize



Ručno (manuelno) testiranje softvera

- ✓ Mana ručno (manuelno) testiranje softvera je dugotrajnost celog procesa i mogućnost ljudske greške.
- ✓ Ljudski faktor u ovom načinu testiranja često je ključan kod što bolje evaluacije.
- ✓ U ovakvom vidu testiranja softvera je veoma bitno iskustvo kao jedan od bitnih faktora za sam kvalitet softvera.



Ručno (manuelno) testiranje softvera

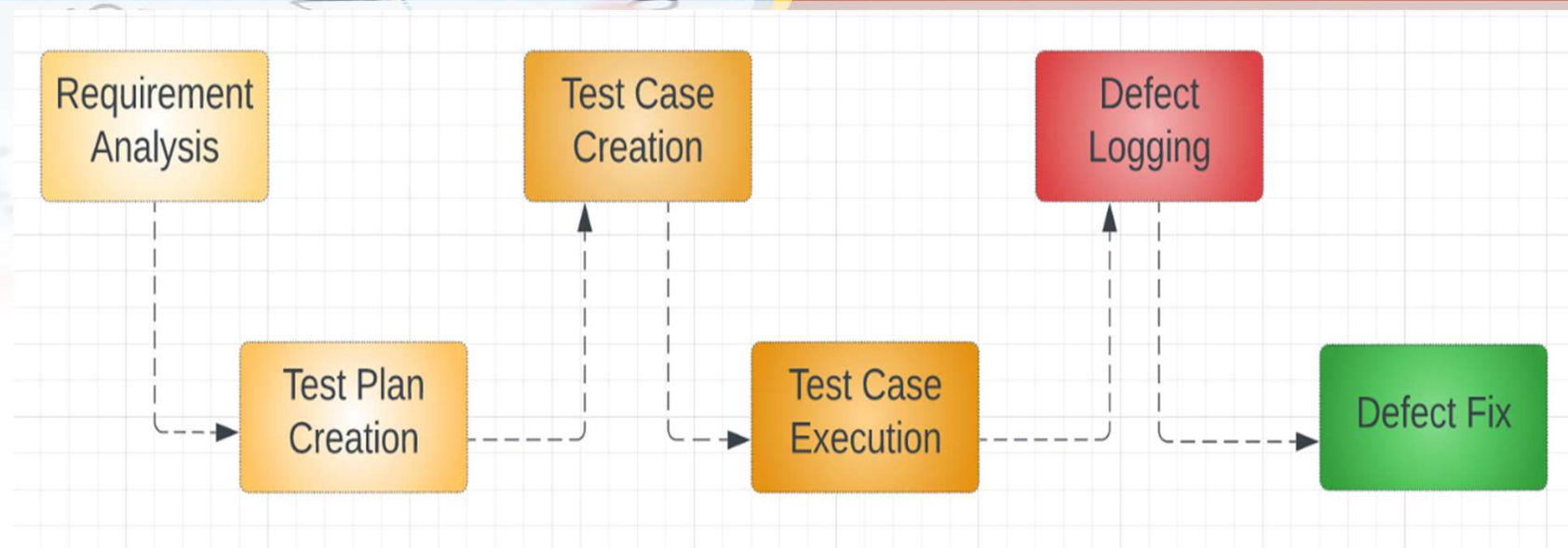
✓ Mana ručno (manuelno) testiranje softvera je:

Puno zavisi od samog testera (njegovog iskustva i njegovog poznavanja programskog jezika, koda i samog domena problema)

Ako se u međuvremu promeni deo koda, neophodno je ponovo testiranje (dugotrajnost celog procesa)

Slabi kriterijumi pokrivenosti

Ručno (manuelno) testiranje softvera



Šest osnovnih faza za izvođenje manualnog testiranja



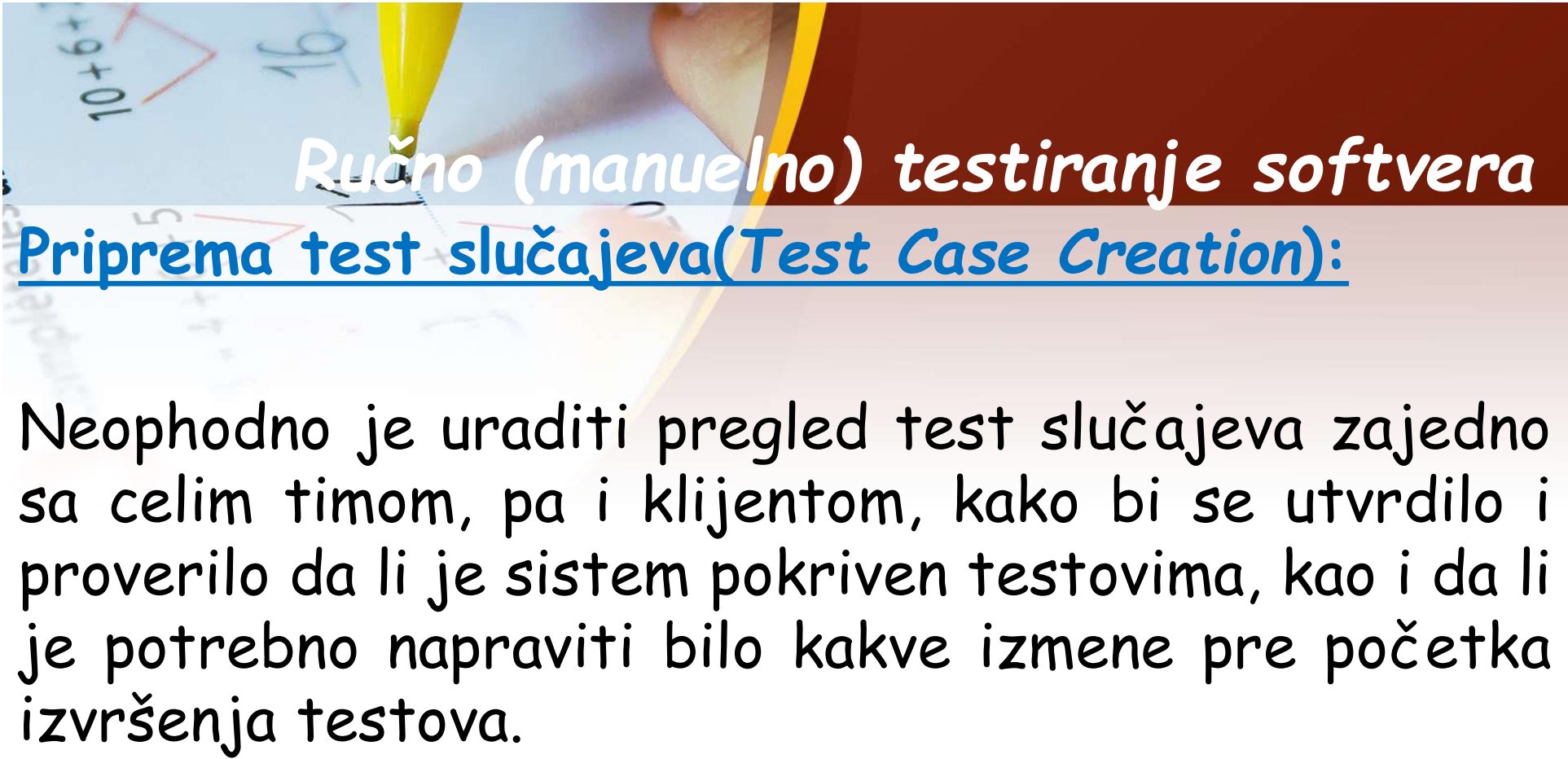
Ručno (manuelno) testiranje softvera

Analiza zahteva (Requirement Analysis):

Test inženjeri moraju analizirati svu dokumentaciju sa zahtevima kako bi prepoznali očekivano ponašanje softvera i ono što je potrebno testirati.

Priprema test plana (Test Plan Creation):

Nakon razumevanja zahteva, prave se test slučajevi u cilju pokrivanja različitih scenarija. Plan testiranja postavlja redosled za testiranje funkcionalnosti i upotrebljivosti za celu aplikaciju, koje se meri u odnosu na očekivane rezultate.



Ručno (manuelno) testiranje softvera

Priprema test slučajeva (Test Case Creation):

Neophodno je uraditi pregled test slučajeva zajedno sa celim timom, pa i klijentom, kako bi se utvrdilo i proverilo da li je sistem pokriven testovima, kao i da li je potrebno napraviti bilo kakve izmene pre početka izvršenja testova.



Ručno (manuelno) testiranje softvera

Izvršavanje test slučajeva (Test Case Execution):

Manuelno testiranje se sada može izvršiti, koristeći bilo koju od gore navedenih tehnika.

Osim pronalaženja grešaka, cilj je identifikovati potencijalne greške do kojih bi mogli da dođu korisnici, kao i „rupe u kodu“ koje bi hakeri mogli iskoristiti.

Test inženjeri izvršavaju test slučajeve jedan po jedan, ponekad koristeći alate za praćenje grešaka kao što je *Jira* alat.



Ručno (manuelno) testiranje softvera

Prijavljivanje grešaka (Defect Logging):

Kada se identifikuju greške, tim za testiranje prosleđuje rezultate testiranja razvojnom timu u obliku izveštaja o testiranju.

Ovo sadrži detalje o tome koliko je nedostataka ili grešaka pronađeno, koliko test slučajeva nije uspelo i koje je potrebno ponovo pokrenuti.



Ručno (manuelno) testiranje softvera

Ispravljanje grešaka (Defect Fix):

Kada se okrije greška u kodu, razvojni tim dobija zadatak da ispravi greške.

Kada razvojni tim popravi greške, softver se ponovo vraća test inženjerima.

Ponovo se pokreće isti test slučaj koji je prvobitno rezultirao greškom, kako bi proverili je li problem rešen.



Ručno (manuelno) testiranje softvera

DA SUBLIMIRAMO!

- Procedura koja se prati prilikom manualnog testiranja:
- Kreiranje plana testa → treba da se Dokumentuje plan testiranja i nove funkcije.
- Stvaranje predmeta testa → Uključuju sve slučajeve vezane za testiranje zadate funkcije.
- Razne kombinacije test scenarija.
- Ispitivanje slučaja → Izvršiti sva testiranja i sve moguće scenarije.
- Logovanje defekta → Kada se nađe greška, treba je dokumentovati u alatu za praćenje i sve zapisati u dnevnik testiranja
- Popravljanje defekata i ponovna verifikacija → Programerski tim ispravlja grešku, testerski tim ponovo testira.
- TESTIRATI I SAMO TESTIRATI