

*Prolećni semestar, 2021/22*

*SE325 - Upravljanje projektima razvoja softvera*

**Razvoj inovativnog rasporeda časova**

Projektni zadatak 2

Profesor: **Nemanja Zdravković**

Asistent: **Tamara Vukadinović**

Student: **Aleksa Cekić 4173**

Niš, 2022.

# Uvod

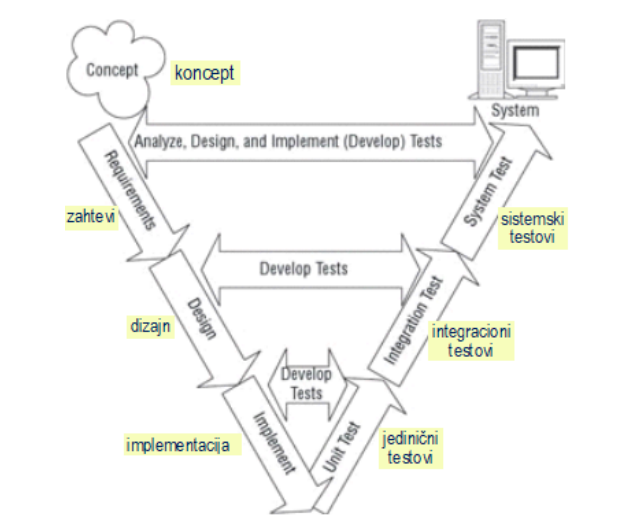
Testiranje je jedna najvažniji faza razvoja bilo kog projekta jer u toj fazi imam najbolju priliku da rešimo sve potencijalne probleme na koje softver može da naiđe u produkciji i tako izbegnemo neprilike. Testiranje treba da bude u prirodi svakog pojedinačnog developera makar na osnovnom nivou. Na nivou ovog projekta testiranje treba da bude iscrpno i plansko kako se očekuje velika komercijalizacija ovog sistema.

# Apstrakt sistema

Krajnji cilj aplikacije jeste sistem koji omogućava pravljenje rasporeda časova na lep, organizovan način, sa raznim alatima koji će pomoći korisniku da vodi računa o vremenu. Jedan od problema koji je inače zastupljen u društvu je taj da su ljudi, pogotovo studenti, uglavnom neorganizovani što im otežava prilikom studiranja i spremanju ispita. Aplikacija je namenjena da korisniku omogući pregled svih potrebnih informacija o predmetu, kao i vremenu početka i završetka predavanja i vežbi tih predmeta koje korisnik unosi u sistem.

Postojećih rešenja već ima u vidu informacionih web sajtova gde korisnici pored ostalog imaju i pristup rasporedu časova, ali sa manjim fokusom na raspored. Sistem koji se razvija bi pružio set interaktivnih alata kao što su alarmi za početak nastava, pristup online predavanjima preko video strima (stream) sa četom (chat) na nivou unetih predmeta. Ovakav vid aplikacije, obzirom na to da se radi o multi-funkcionalnoj aplikaciji, pruža to da se korisnici na organizuju na bolji i efektivniji način.

# Test Plan



Slika prikazuje V model razvoja softvera koji će se privenjivati. Pokazuje ispitne aktivnosti koje započinju paralelno sa odgovarajućim razvojnim aktivnostima. Ovaj model ukazuje da su testovi prihvatanja izvedeni iz potreba korisnika, testovi sistema zasnovani su na funkcionalnim zahtevima, a testovi integracije zasnovani su na arhitekturi sistema. Ovaj model je primenljiv bez obzira da li su aktivnosti razvoja softvera koji se testiraju odnose na proizvod u celini, posebno izdanje, ili na pojedinačni priraštaj razvoja.

***Metode verifikacije i validacije***

Metode verifikacije i validacije mogu biti:

- **Dinamičke** koje su zasnovane na izvršavanju koda na platformama za testiranje (test sistemu). Dinamičke metode se primenjuju tek od trenutka kad je dostupan kod se završe prvi delovi koda (units).

- **Statičke**, bez izvršavanja koda, koje čine pregledi i provere rezultata aktivnosti.

Statičke metode se koriste od početka projekta i imaju raznovrsnu formu pregleda (Ove metode su u manjoj ili većoj meri formalizovane, tj. propisane su

procedure (ko kako i kada ih vrši).

**Statičke metode provere (na nivou cele aplikacije)**

Statičke metode provere su tehnike zasnovane na neizvršavanju koda, tj. provere su zasnovane na dokumentima. Ove provere uključuje preglede i razmatranja u cilju razumevanja i provere zahteva, dizajn dokumenata i kodiranja. Svaki dokument prolazi kroz provere i poređenja sa ček-listama, drugim dokumentima, standardima i ustaljenom praksom. Osnovne metode statičke metode verifikacije i validacije su:

• Neformalni pregledi i komentari

• Walkthrough (prolazak)

• Recenzija

***Prolazak kroz proceduru (Walkthrough***) je polu-formalni pregled gde dokument prolazi kroz prezentaciju i diskusiju vođenu od strane autora. Autor upoznaje učesnike sa ciljevima, sadržajem i zaključcima dokumenta, da se postigne zajedničko razumevanje i da se prikupe povratne informacije o mišljenju auditorijuma.

Ova tehnika je veoma korisna za auditorijum koji nije iz discipline softvera, tj. koji ne poznaje i ne koristi terminologiju i metode softverskog inženjerstva, ili ne mogu lako da razumeju proces razvoja softvera.

Takođe, ova tehnika li je posebno korisna za veće dokumente kao što su nivo obavezne specifikacije, itd.

***Recenzija*** (ili peer review ) je formalni postupak u kome postoje tri osnovna aktera : recenzenti, autor i moderator.

Recenzenti (review-eri) vrše pregled i ocenu rezultata. Obično svaki recenzent vrše I nezavisno od ostalih svoju evaluaciju i dostavlja moderatoru.

Autor dostavlja predmet koji treba oceniti (kod, dokumente itd.) i daje potrebna odnosno dodatna objašnjenja. Moderator, koji organizuje recenziju. Ne učestvuje u samoj oceni, ali formuliše konačne zaključke na bazi ocena recenzenata. Izveštaj o recenziji sa ocenom (npr. rezultat prihvaćen, ili prihvatljiv uz dodatne korekcije) i preporukama šta treba uraditi se dostavlja menadžmentu projekta i autoru.

# Integraciono testiranje (integracije sa ostalim sistemima)

Integraciono testiranje se radi posle jediničnog testiranja ( “Unit testing” ). Nakon što su sve individualne jedinice (metode) napravljene i testirane, počinjemo sa kombinovanjem ovih jedinično-iztestiranih modula i tako radimo integraciono testiranje. Integraciono testiranje se obavlja u toku razvoja softvera a ne na kraju razvoja softvera. Koristi se mogu dobiti od ovakve pažljive integacije su lakša dijagnoza defekata, manje defekata, manje vremena do prvog operativnog proizvoda, kraće ukupno vreme razvoja proizvoda,veće šanse da se projekat uradi uspešno i na samom kraju pouzdanije procene rokova i naravno bolji kvalitet programa.

Postupak integracionog testiranja:

1. individualni moduli se prvo testiraju u izolaciji od ostalih modula odnosno jedinično testiraju
2. integrisati tj. kombinovati jedinično-iztestirane module, jedan po jedan, u složenije celine
3. testirati sve dok se ne integrišu svi moduli, i to uključujući kombinovano ponašanje modula
4. testirati svaki ovaj složeni modul kao jednu jedinicu
5. glavni cilj da se testiraju interfejsi između jedinica/modula i da se izvrši validacija da li su zahtevi implementirani korektno

# 

# Zaključak

Radi što većeg stepena zadovoljnih korisnika softvera potrebno je obezbediti u samom startu proizvod visokog kvaliteta koji će garantovati njegovu vrednost i pouzdanost. Kreiranje softvera se može posmatrati i kao poslovni dogovor. Softver treba da iznese klijentske potrebe bez greške. Testiranje softvera nudi dokaz za softver ispunjava poslovne zahteve klijenta.

Iako testiranje može dovesti do povećanja planiranih troškova, kašnjenja… ovo je svakako zanemarljivo u odnosu na vrednost koja se dobija od samog testiranja. Kako su bagovi popravljeni pre objave softvera, troškovi popravke bagova i održavanje softvera su svedeni na minimum.

U razvoju softvera, sve se mora duplo proveriti pre objave proizvoda. Da bi ostali u poslu kao IT kompanija, imperativ je imati tim softver testera. Proizvodi sa kritičnim bagovima, mogu prouzrokovati velike gubitke u poslovanju. Zato danas, svaka ozbiljna IT kompanija investira u svoj tim testera koji imaju zadatak da do toga ne dođe. Otkrivaju greške još u fazi razvoja, kako bi softverske performanse bile adekvatne i kako bi korisnik bio zadovoljan.

# Literatura

* <https://dtc.rs/sr/testiranje-softvera>
* <https://docplayer.rs/116320652-Testiranje-softvera.html>