UNIVERZITET METROPOLITAN BEOGRAD

# SE101 - RAZVOJ SOFTVERA I INŽENJERA SOFTVERA

# Izazovi u softverskom inzenjerstvu

Lekcija 13

PRIRUČNIK ZA STUDENTE

# SE101 - RAZVOJ SOFTVERA I INŽENJERA SOFTVERA

## Lekcija 13

## *IZAZOVI U SOFTVERSKOM INZENJERSTVU*

# UVOD

## *Cilj i sadržaj lekcije*

U okviru lekcije imaćete priliku da naučite nešto više o izazovima koji se javljaju u softverskom inženjerstvu kroz:

- objašnjenje kakve to izazove donose novi trendovi u mobilnom računarstvu, računarstvu u oblaku i zajedničkog razvoju
- odgovor na pitanje zašto zahtevi koji se često menjaju predstavljaju izazov u softverskom inženjerstvu
- razumevanje kojih su to pet osnovnih izazova prilikom upravljanja promenama zahteva
- informacije o izazovima koji se odnose na korisnike i projektni tim
- objašnjenje izazova u okviru održavanja softvera
- definisanje kategorija izazova upravljanja kvalitetom i objašnjenje svake od kategorija
- razumevanje značaja komunikacije i uticaja geografske, vremenske i socio-kulturalne udaljenosti na kvalitet komunikacije

# Poglavlje 1

# Biti u toku sa trendovima

## IZAZOVI PRILIKOM PRAĆENJA TRENDOVA U SOFTVERSKOM INŽENJERSTVU

*Mobilno računarstvo i računarstvo u oblaku kao dva nova trenda u softverskom inženjerstvu menjaju ponašanje i tokove rada potrošača i IT odeljenja preduzeća*

Nalazimo se usred dve važne promene u IT industriji koje izazivaju talasne efekte u svim aspektima razvoja softvera, isporuke i potrošnje. Ove dve promene se obično označavaju kao „ulazak u post-PC“ i „ulazak na post-server“ ili era računarstva u oblaku. Era nakon PC računara manifestuje se pomeranjem ka sve većoj upotrebi mobilnih računarskih uređaja kao što su mobilni telefoni, tableti i drugi prenosivi uređaji. Ovi mobilni računari imaju isti računarski kapacitet kao i laptop računari početnog nivoa, ali imaju određena ograničenja, a istovremeno nude i nove funkcije. Ključna ograničenja su:

- manji ekran za izlaz,
- dodir ili glasovni interfejs kao ulaz.

Glavne nove karakteristike koje ovi uređaji nude, pored prenosivosti, su stalna veza sa internetom i činjenica da su oni takođe komunikacioni uređaji sa mogućnostima sličnim telefonu.

Post-serverska era manifestuje se potezom kompanija da koriste računarske resurse koji nisu lokalni, već im se pristupa preko interneta ili "oblaka". To znači da IT odeljenja kompanija više ne grade namenske centre podataka ili obezbeđuju servere, već iznajmljuju potrebne računarske kapacitete od dobavljača računarstva u oblaku. Prednosti su niži troškovi i plaćanje samo za ono što je potrebno u odnosu na plaćanje unapred kapitalnih troškova i rizik od nedovoljne ili prevelike nabavke računarskih resursa.

Ova dva trenda menjaju ponašanja i tokove rada potrošača i IT odeljenja preduzeća. Međutim, postavlja se pitanje da li oni imaju druge uticaje na softversku industriju. Konkretno na to kakav je uticaj na životni ciklus softvera i kako post-PC i postserverska era menjaju način na koji je softver dizajniran i razvijen. Takođe, pitanje je da li bi prilagođene verzije postojećih razvojnih procesa bile prikladnije za razvoj oblaka.

# TRENDOVI U MOBILNOM RAČUNARSTVU

## *Testiranje mobilnih aplikacija je sve teže kada se ima u vidu da povezanost i ambijentalna lokacija korisnika značajno utiču na njihovo iskustvo korišćenja*

Trend mobilnog računarstva ima tri glavne realnosti koje utiču na upotrebu i tipove softvera kreiranog za ove uređaje.

1. Uređaji imaju ograničenu površinu ekrana. To znači da je izlaz za ove uređaje ograničen i da se mora uštedeti. Kod novijih telefona ceo uređaj čini ekran, ali su i dalje ograničeni na oblik koji lako stane u džep košulje.
2. Uz mobilnost dolazi i stalna povezanost. Ovi uređaji su često komunikacioni uređaji, ali su i uređaji sve veće potrošnje sa nekom vezom na internet. Kombinacija mobilnosti i povezanosti znači da usluge i aplikacije zasnovane na lokaciji konačno postaju stvarnost.
3. Konačna realnost trenda mobilnog računarstva su nove vrste mehanizama za unos za ove uređaje. Pre svega postoje dve vrste unosa: dodirni i glasovni. Ova dva načina unosa su prirodnija za mobilne uređaje zbog toga što se koriste u pokretu, pa je malo verovatno da će tastature i miševi biti priključeni na ove uređaje. Dodir je bogat i prirodan korisnički unos, dok glas omogućava način unosa koji zahteva manje vizuelnog fokusa od korisnika. Na taj način postiže se mogućnost obavljanja operacija dok korisnik obavlja druge zadatke, na primer, vožnju, hodanje i tako dalje.

Kao posledica buma mobilnog računarstva, softverske aplikacije su napravljene tako da imaju ne samo novije interfejse, sa različitim modernim režimima unosa, već su postale manje i specijalizovanije. Aplikacije za mobilne računarske uređaje su male i specijalizovane i prodaju se po veoma niskim cenama i ponekad su potpuno besplatne. Mobilne aplikacije se dopunjuju veb uslugama i integrišu jedna sa drugom radi proširenja funkcija.

Izazovi u razvoju softvera su na prvom mestu u činjenici da se ove mobilne aplikacije brzo prave sa namerom da popune specijalizovani i nišni prostor. Pošto se aplikacije prodaju po niskim troškovima, programeri žele da nadoknade troškove razvoja obimom prodaje, kao i unakrsnim promocijama i unakrsnom prodajom reklama ili dodatnih funkcija koje proširuju aplikaciju. Uticaj na procese razvoja softvera je da se agilne tehnike potiskuju do njihovih granica, kao i da se dovedu u pitanje neka dugo držana načela kao što je uključivanje korisnika i razvoj zasnovan na testovima. Uključivanje korisnika ima svoj skup izazova koji se mogu prevazići, ali da bi se to uradilo potrebno je prvenstveno pokušati da replicira kontekst u kome kupac namerava da koristi softver. Takva replikacija konteksta je posebno teška kada se ima u vidu da povezanost i ambijentalna lokacija korisnika značajno utiču na njihovo iskustvo korišćenja. Samim tim, testiranje je sve teže, pošto su kontekstualne promene toliko raznolike i česte da ih je teško u potpunosti ponoviti.

# TREND RAČUNARSTVA U OBLAKU

## *Izazovi koji utiču na razvoj softvera u oblaku su sigurnost i privatnost osetljivih podataka i uparvljanje pristupom*

Drugi trend koji bi trebalo razmotriti je kretanje ka iznajmljivanju računarskih resursa na internetu. Dobro je poznato da dobavljači računarstva u oblaku mogu ponuditi ove računarske resurse za cene koje je teško dostići ukoliko bi se stvarali sopstveni centri podataka. Pored toga, pošto korisnici mogu da povećavaju i smanjuju veličinu prostora na osnovu svojih potreba, ekonomiju računarstva u oblaku je teško uporediti sa tradicionalnim načinom rada. Međutim, vredi napomenuti da, iako računarstvo u oblaku utiče na ekonomiju razvoja i primene softvera, ono takođe utiče na način na koji se softver dizajnira i održava:

- Pošto se serveri računarstva u oblaku iznajmljuju po potrebi, cena prebacivanja računarstva na servere nikada nije bila pristupačnija. To znači da arhitekture aplikacija dizajnirane za oblak guraju podatke i računanje na servere. Ovo zaista dobro dopunjuje trend mobilnih uređaja, koji imaju ograničen ekran i kapacitet i stoga su savršeno prilagođeni da deluju kao portovi za pregled do usluga u oblaku.
- Drugi aspekti razvoja softvera zasnovanog na oblaku imaju veze sa testiranjem i eksperimentisanjem. Prvo, pošto su troškovi servera marginalni, kontinuirana integracija i kontinuiranotestiranje su više pretpostavljena stvarnost. Pored toga, mogućnost eksperimentisanja sa visokim računarskim opterećenjem je pristupačnija. Sada su moguće funkcije koje zahtevaju integraciju sa velikim bazama podataka ili funkcije koje zahtevaju teška računanja po korisniku.

Naravno, računarstvo u oblaku takođe dolazi sa sopstvenim skupom izazova koji takođe utiču na razvoj softvera. U principu, to se odnosi na upravljanje aplikacijama u oblaku:

- Pošto se podaci aplikacije nalaze na udaljenim serverima, mora se voditi računa o tome koliko su ovi podaci sigurni i povezanim problemima privatnosti za osetljive podatke.
- Upravljanje pristupom resursima u oblaku i upravljanje načinom na koji se verzije softvera guraju i vraćaju u prethodno stanje postaje ozbiljnija briga jer se ovi zadaci razvoja i održavanja lako izvode i stoga je veća verovatnoća da se zloupotrebljavaju.

# ZAJEDNIČKI RAZVOJ

## *Neke kompanije sve više koriste svoje korisnike za beta testiranje i za pružanje izveštaja o greškama za nove funkcije*

Dva navedena trenda (mobilni uređaji i računarstvo u oblaku) postavljaju neka zanimljiva pitanja o tome kako se procesi razvoja softvera mogu poboljšati i mogu pomoći programerima da kreiraju bolji softver. Jedan takav pristup je kroz *zajednički razvoj*.

Ekstremna verzija zajedničkog razvoja je dobro poznata i korišćena u razvoju otvorenog koda (OSS) koji sve više proizvodi kritičan softver i neke od najčešće korišćenih softvera danas, na primer, Linuks i Apache server.

Međutim, zajednički razvoj se takođe može obaviti u manjem obimu i imati više društvenih karakteristika. Usluge kao što su GitHub, Google Code i Jazz.net su sjajni primeri alata i usluga za zajedničkii razvoj koji prihvataju male i velike timove i pružaju funkcije koje mogu pomoći da proces razvoja bude dinamičniji i u potpunosti se obavlja u oblaku.

Pored toga, da bi se pozabavile nekim od ciklusa brzog testiranja za ovu novu klasu razvijenog softvera, neke kompanije, kao što je Facebook, sve više koriste svoje korisnike za beta testiranje i za pružanje izveštaja o greškama za nove funkcije. Ova vrsta obezbeđenja kvaliteta nije bez rizika, ali omogućava učešće najaktivnijih korisnika i omogućava da ciklusi razvoja, primene i proizvodnje softvera budu praktično trenutni.

Naravno, korišćenje više zajedničkog razvojnog procesa predstavlja sopstveni skup izazova. U ekosistemima OSS obično postoji mnogo projekata bez vlasnika. Mnogi projekti su pokrenuti ali ne uspevaju da se materijalizuju u nešto konkretno i korisno. Ovo je takođe posledica stvarnih izazova za stvaranje i negovanje razvojnih zajednica. Bez pravih podsticajnih struktura, razvoj saradnje mogao bi biti više izazov nego rešenje.

# Poglavlje 2

# Zahtevi koji se stalno menjaju

## RAZLOZI STALNE PROMENE ZAHTEVA

*Prepoznavanje trenutnih problema i savremenih izazova upravljanja promenama softvera ima značajnu vrednost za razvoj potpuno funkcionalnog softvera*

Danas se projektni zahtevi menjaju i evoluiraju od samog početka razvoja, a zatim tokom celog životnog ciklusa sistema. Generalno, promena zahteva je rezultat dodavanja, uklanjanja ili ažuriranja neke od sledećih komponenti:

- proizvod,
- usluga,
- uloga stejkholdera,
- poslovno pravilo,
- bilo koje ograničenje koje reguliše prethodne elemente.

Postoji više razloga zbog kojih softver odgovara promenama zahteva. Na primer, u ranim fazama razvoja softvera, zahtevi su obično nepotpuni zbog nedovoljno jasne vizije potrebnih poslovnih ciljeva. Dodatni faktori, kao što su novi propisi vlada, promena cene akcija itd. podstiču poslovanje na razvoj novih zahteva. Do promena zahteva takođe dolazi i zbog internih promena, poput nestabilnosti poslovanja, želje da se ostane konkurentan, i sl. Shodno tome, nedostatak odražavanja promena na softveru ometa zadovoljstvo korisnika i blokira kontinuirani napredak funkcionalnosti softvera.

U poslovnom svetu, sposobnost brzog prilagođavanja informacionog sistema promenljivim zahtevima je kritična. Ovo se zauzvrat odražava na potrebu usklađivanja između tekućih poslovnih zahteva i softvera koji ima ključnu ulogu za postizanje poslovnih ciljeva. Važno je napomenuti da je ukupan uspeh projekta izuzetno pogođen promenama zahteva. U stvari, bilo je mnogo slučajeva da delovi poslednjeg proizvoda nisu ispunjavali potrebne uslove kupaca pošto zahtevana promena nije bila precizno sprovedena. U tom pogledu, nepravilno upravljanje promenama softvera dovodi do potpunog otkaza sistema i može biti uzrok poslovnog gubitka. Shodno tome, prepoznavanje trenutnih problema i savremenih izazova upravljanja promenama softvera ima značajnu vrednost za razvoj potpuno funkcionalnog softvera.

# UPRAVLJANJE PROMENAMA U ZAHTEVIMA

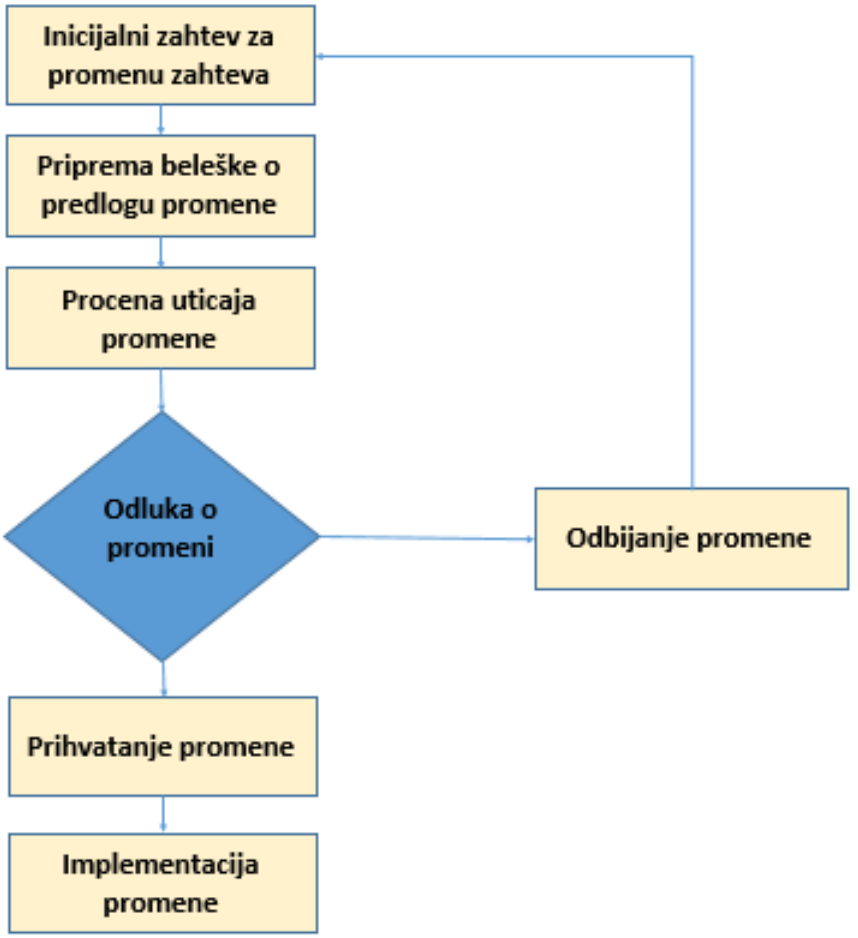
## *RCM je definisan kao "procedura upravljanja promenama u zahtevima tokom procesa inženjeringa zahteva i razvoja sistema"*

*Upravljanje promenama u zahtevima, RCM* ( Request Change Management) je primarni faktor za uspešno sprovođenje promene zahteva. RCM je definisan kao „procedura upravljanja promenama u zahtevima tokom procesa inženjeringa zahteva i razvoja sistema“. Upravljanje podrazumeva „dodavanje, brisanje ili ažuriranje zahteva i ispravljanje grešaka“. Glavni koraci RCM procesa su identifikovani na sledeći način:

- početni zahtev za promenu,
- priprema napomene o predlogu promene,
- procena uticaja promene,
- odluka o promeni da li da se prihvati ili odbije,
- implementacija promene u sistemu.

Postoji mnogo problema vezanih za RCM koje treba razjasniti, a glavni su:

- neadekvatnost u opisivanju i identifikaciji promena zahteva,
- dvosmislenost zahteva,
- sledljivost zahteva,
- vreme i cena.




Slika 2.1 RCM proces [Hussin Ahmed, Azham Hussain, Fauziah Baharom, Current Challenges of Requirement Change Management, School of Computing, Universiti Utara Malaysia, 06010 UUM, Kedah, Malaysia, 2016. ]

# IZAZOVI ZA UPRAVLJANJE PROMENAMA ZAHTEVA

## *Pet osnovnih izazova prilikom upravljanja promenama*

U literaturi su ključni izazovi identifikovani na sledeći način:

- Ponovna upotreba - U današnje vreme brza evolucija poslovnih zahteva i potreba za fleksibilnošću podstiču potrebu ponovnog korišćenja funkcionalnih i prethodno testiranih poslovnih delova u aktuelnim informacionim sistemima. Ponovna upotreba se u početku mora koristiti tokom razvoja zahteva, a zatim će sredstva koja se mogu ponovo koristiti biti replicirana u RCM.
- Merenje aktivnosti promene ima veliku vrednost za beleženje istorije promena svakog zahteva. To je način za procenu stabilnosti zahteva i otkrivanje mogućnosti za poboljšanja procesa koje mogu dovesti do manje promena u budućnosti. Štaviše, pomaže u saopštavanju promena i održavanju istorije revizija. Merenje aktivnosti promena će pomoći da se stekne uvid u budućnost u svrhu optimalnog donošenja odluka i idealne implementacije. Merenje aktivnosti promena uzima u obzir tip promene kao što je dodavanje, ažuriranje/brisanje, datum promene, tip zahteva.

- Povezanost sa softverskim artefaktima - Razvoj zahteva i softverskog koda su međusobno usko povezani. Uspostavljanje veze između njih doprinosi rešavanju mnogih problema, na primer, promena zahteva, a zatim i koda, znači da tim održavanja mora da uradi analizu uticaja promene dva puta: jednom na nivou zahteva gde postoji potreba da se utvrdi uticaj promene, a zatim i na nivou koda.
- Predviđanje promena - Predviđanje promena u RCM-u je od velike vrednosti u trenutnom konkurentnom svetu. Neki zahtevi se pojavljuju u budućnosti i oni su nepoznati u trenutku kada je informacioni sistem projektovan ili izgrađen. Predviđanje promena tržišta i zahteva kupaca i brzo reagovanje dobija veliku pažnju u okviru novih veb aplikacija. Praktično je i značajno identifikovati potencijalne promene, a ne dozvoliti da se one odlažu za kasnije faze proizvodnje.
- Automatizacija upravljanja promenama - Jedan od ključnih problema u savremenim metodologijama upravljanja promenama je prevelika zavisnost od ljudske uloge koja ne garantuje ponovljivost rezultata promene. U stvari, mnoga IT odeljenja i poslovne jedinice troše resurse i većinu vremena na održavanje postojećeg sistema i nemaju preostalu snagu da ostvare nove zadatke. Stoga je poželjno definisati softverski proces sa dovoljnom preciznošću kako bi se mnogi od rutinskih zadataka mogli automatizovati. Evidentno je da postoji krucijalna potreba za modelom koji je orjentisan na agenta za automatizaciju zadataka u RCM-u kako bi se rešili aktuelni problemi u cilju uštede vremena, smanjenja troškova i što je najvažnije uspešne implementacije promena. Korišćenje agenata u RCM-u bi olakšalo proces rukovanja ogromnom količinom informacija sa velikim nivoom tačnosti i doslednosti.

# Poglavlje 3

# Izazovi vezani za korisnike i projektni tim

## IZAZOVI VEZANI ZA KORISNIKE

*Dva osnovna izazova vezana za korisnike su nedovoljna uključenost i nejasni zahtevi od strane klijenata*

Učešće korisnika u procesu razvoja softvera je veoma važan aspekt za uspeh projekta.

Programeri imaju poteškoća u radu sa klijentima na softverskim projektima pretežno zbog nedovoljne uključenosti korisnika, odnosno njihovog odsustva do kraja projekta. Takođe, navodi se da su korisnici često zauzeti drugim poslom i nemaju vremena da po ceo dan razgovaraju sa programerima. Ovo na kraju smanjuje komunikaciju sa klijentom sa jednom nedeljno na samo dva puta po sprintu.

Kao sledeći veliki izazov pominje se tehnička specifikacija koja nije na vreme dostavljena programerima što podrazumeva dodatno produženje vremena projekta i prenošenje odgovornosti za specifikaciju na programere. Takođe, klijenti najčešće ne znaju šta zaista žele u svom budućem sistemu, a to postaje prepreka za uključivanje korisnika u proces razvoja projekta. Pokazalo se da klijenti misle da imaju jasnu predstavu o softveru ali se često ispostavi da nemaju, da pružaju nedostatak informacija, što se na kraju može svesti na nejasne zahteve korisnika. Ovo postavlja izazov programerima da shvate šta tačno korisnici traže.

Han i Huang identifikovali su nekoliko izazova vezanih za korisnike prilikom upravljanja projektima razvoja softvera. To su:

- korisnici su često otporni na promene,
- pojava sukoba među korisnicima,
- neki korisnici imaju negativne stavove prema projektu,
- nedostatak posvećenosti projektu.

## IZAZOVI VEZANI ZA PROJEKTNI TIM

*Osnovni izazovi vezani za projektni tim odnose se na komunikaciju, koordinaciju i probleme poverenja u okviru projektnog tima*

U nastavku su navedeni osnovni izazovi koji se odnose na funkcionisanje projektnog tima.

Opšte je poznato da je nedostatak komunikacije među programerima koren neuspeha u većini softverskih projekata i da se veliki deo vremena programera troši na komunikaciju. Postoje mnoge tehnologije predložene za ublažavanje problema u komunikaciji, uključujući e-poštu, veb-demo sa interfejsom, razmenu trenutnih poruka, virtuelnu privatnu mrežu ( VPN), telefonske konferencije i video konferencije. S obzirom da je problem komunikacije izazuzetno značajan i sveobuhvatan izazov, celokupan sledeći objekat učenja posvećen je ovoj temi.

Problemi koordinacije i kontrole javljaju se prilikom podele i dodeljivanja posla. Ako obe lokacije (projektni tim i klijent) pokušaju da kontrolišu projekte doći će do sukoba u upravljanju projektima. Jedan od predloga je da se postavi ambasador na lokaciji dobavljača i koordinator na lokaciji klijenta i dozvoli im da deluju kao predstavnici svake lokacije kako bi se smanjili problemi koordinacije i kontrole. Takođe, ambasador na lokaciji dobavljača treba da bude lokalno lice, a da obrazovana osoba na lokaciji klijenta bude dugo vremena.

Problemi sa poverenjem nastaju ako programeri na strani projektnog tima ne završe zadatke u određenom vremenskom periodu ili ne shvate zahteve klijentske lokacije i završe zadatak na drugačiji način. Ako se ovo često dešava, programeri na lokaciji klijenta imaju poteškoća da uspostave osećaj poverenja kod programera na strani dobavljača, a nedostatak poverenja između dve lokacije značajno otežava saradnju.

# Poglavlje 4

# Komunikacija u globalnom razvoju softvera

## ZNAČAJ KOMUNIKACIJE

### *U globalnom razvoju softvera komunikacija je glavni problem zbog geografske, vremenske i socio-kulturalne udaljenosti*

U globalnom razvoju softvera (Global Software Development, GSD) komunikacija je glavni problem zbog geografske, vremenske i socio-kulturalne udaljenosti. Kvalitet i učestalost komunikacije u globalnom razvoju softvera su na nižem nivou u poređenju sa timovima koji su smešteni zajedno. Zbog problema komunikacije, distribuirani projekti traju i do dva ipo puta duže u odnosu na iste projekte realizovane od strane timova koji su smešteni na istoj lokaciji. Barstou je naveo da „pojedinačni softverski inženjer troši više vremena na razmenu informacija od bilo koje druge aktivnosti“.

- *Geografska udaljenost* - nedostatak blizine kolega, nedostatak transparentnosti zadataka i nedostatak timskog rada.

Postoje različiti rizici koji nastaju zbog geografske udaljenosti među članovima tima. Ograničena vidljivost udaljenog člana tima zbog putnih troškova je jedan primer takvih rizika. Zbog troškova putovanja, komunikacija licem u lice postaje prilično teška. Nedostatak sastanka licem u lice ili ograničena vidljivost udaljenih članova tima može dovesti do nezadovoljstva među članovima što direktno smanjuje moral tima. Dugoročno, to dovodi do problema u koordinaciji jer članovi tima nisu spremni da međusobno sarađuju. U geografski razuđenim timovima takođe je veoma teško imati neformalnu komunikaciju između članova tima što dovodi do manje motivisanosti zbog ograničenih zajedničkih aktivnosti.

- *Vremenska udaljenost* - kašnjenje u odgovoru se povećava što dalje povećava frustraciju.

Globalno raspoređeni timovi nisu samo geografski udaljeni već se nalaze i u različitim vremenskim zonama poznatim kao vremenska udaljenost. Zajedno sa rizicima povezanim sa geografskom distancom i društveno-kulturološkim distancama, vremenska udaljenost takođe dodaje više komunikacijskih rizika. U slučaju manjeg broja sati koji se preklapaju, postoji ograničena mogućnost komunikacije sa udaljenim kolegama tako da su članovi tima prinuđeni da se oslanjaju na asinhronu komunikaciju koja nije efikasna. Kašnjenja se obično dešavaju u pogledu odgovora i nema čestih povratnih informacija između članova tima zbog velike vremenske udaljenosti. Kontinuirano kašnjenje u odgovoru može dovesti do kašnjenja u rešavanju zadataka, kao i do gubitka poverenja među distribuiranim timovima. To takođe može dovesti do problema u koordinaciji između timova. Vremenska udaljenost se takođe može videti ako postoje različiti radni dani. Zavisnosti od „informacionih i komunikacionih”

alata se povećavaju zbog geografske i vremenske udaljenosti, što rezultira povećanjem napora prilikom prenosa znanja.

# SOCIO-KULTURALNA DISTANCA

## *Socio-kulturalna distanca povećava nedostatak međusobnog razumevanja, povećava mogućnost pogrešne komunikacije i ističe različito ponašanje ljudi u različitim situacijama*

- *Socio-kulturalna distanca* povećava nedostatak međusobnog razumevanja, povećava mogućnost pogrešne komunikacije i ističe različito ponašanje ljudi u različitim situacijama zbog drugačijih političkih i verskih načela.

U globalnom razvoju softvera ljudi različitog kulturnog porekla, nacije koje govore različite jezike rade jedni sa drugima. Ove razlike se nazivaju socio-kulturalna distanca. Socio-kulturalna distanca se može meriti načinom na koji ljudi iz različitih kulturalnih sredina (kao što su jezici i etika) razumeju i reaguju na određene situacije. Postoje različiti rizici koji mogu nastati zbog ove društvene kulturalne distance.

Kulturalna distanca postaje jedan od najvećih rizika zbog slabe svesti o društveno-kulturološkoj usklađenosti. Ne podrazumeva se da kulturalna distanca postoji samo ako su članovi tima iz različitih nacija, može se desiti i da su članovi tima iz iste nacije ali iz različitih organizacija koji rade zajedno kao zajedničko ulaganje i koji takođe mogu imati kulturološke razlike.

Još jedan rizik koji se javlja uglavnom zbog loše socio-kulturalne usklađenosti je jezička barijera ili razlika. Ovaj nedostatak zajedničkih ili poslovnih jezičkih veština može dovesti do nerazumevanja među članovima tima. Zbog toga što je teško izraziti osećanja ili ideje na stranom jeziku kao što se to može izraziti na maternjem jeziku može doći do nesporazuma među članovima tima.

U lošem društvenom kulturalnom uklapanju, različiti istraživači su raspravljali o razlikama u terminologiji i smatraju to glavnim razlogom za pogrešno tumačenje ili nerazumevanje poruka ili zahteva. Ovi nesporazumi mogu dalje dovesti do gubitka poverenja među članovima tima.

Nedostatak međusobnog razumevanja loše utiče na projekat i može rezultirati lošim kvalitetom proizvoda i lošim ponašanjem timova među sobom. Zbog kulturološke raznolikosti i loših komunikacionih veština, razmena znanja postaje teška jer ljudi ili nisu u stanju ili ne žele da dele znanje među distribuiranim timovima što ponovo može da rezultira gubitkom međusobnog poverenja.

# Poglavlje 5

# „Borba” sa održavanjem softvera

## ODRŽAVANJE SOFTVERA

*Termin „održavanje softvera" se koristi za razumevanje radnji softverskog inženjeringa koje se dešavaju tokom napredovanja softvera*

Održavanje je poslednja faza životnog ciklusa razvoja softvera.

Termin „*održavanje softvera*" se koristi za razumevanje radnji softverskog inženjeringa koje se dešavaju tokom napredovanja softvera. Proces održavanja softvera je veoma komprimovan i obično ponovo obuhvata više od polovine procesa razvoja. Tipično, razvoj softvera traje 1 do 2 godine, dok faza održavanja traje 5 do 10 godina. Kada kompanija izda uspešan projekat svom klijentu u određeno vreme, tada počinje stvarni rad na održavanju. Mnogo puta se videlo da troškovi održavanja premašuju troškove razvoja projekta. U osnovi, faza održavanja softvera održava softver u toku sa promenama okruženja, ispravlja greške i poboljšava performanse softverskog proizvoda nakon isporuke.

Uobičajeno viđenje faze održavanja je da ona uključuje samo otklanjanje kvarova. Međutim, prethodna istraživanja pokazuju da se većina, preko 80% napora na održavanju koristi za nekorektivne aktivnosti.

U osnovi postoji sedam glavnih faza u procesu održavanja, koje su date na sledeći način:

1. *Upravljanje promenama* - Faza u kojoj korisnik apeluje za izmenu. Takođe uključuje aktivnosti da se utvrdi da li da se prihvati ili odbije zahtev.
2. *Analiza* - Osnovni cilj je da se zaključi mogućnost realizacije tražene promene. Analiza se sprovodi na dva nivoa: analiza izvodljivosti i detaljna analiza.
3. *Dizajn* - Izmena sistema je zapravo osmišljena u ovoj fazi.
4. *Implementacija* - Aktivnosti kodiranja i testiranja jedinica, asimilacija prilagođenog koda, integracija i analiza, regresiono testiranje i rizik. Faza takođe uključuje pregled spremnosti za testiranje.
5. *Testiranje sistema* - Kompletan sistem se testira da bi se postigao određeni nivo usaglašenosti sa novim zahtevima.
6. *Testiranje prihvatanja* - Ovaj nivo testiranja uključuje korisnike, kupce ili treću stranu koju imenuje kupac.
7. *Isporuka* - Uključuje aktivnost obaveštavanja korisničke zajednice, izvođenje instalacije i obuke.

# IZAZOVI VEZANI ZA ODRŽAVANJE SOFTVERA

*Jedan od najvažnijih izazova u održavanju softvera je otkriti efekte predložene modifikacije na ostatak sistama*

Većina problema koji su povezani sa održavanjem softvera uzrokovani su nedostacima u procesu razvoja softvera. Postoji nekoliko tehničkih i menadžerskih problema prilikom održavanja softvera:

- *Troškovi* - Različite istraživačke studije su predložile da održavanje softvera odnosi 60% do 80% troškova u toku životnog veka softvera. Ove ankete takođe navode da su troškovi održavanja uglavnom posledica poboljšanja, a ne ispravki.
- *Analiza uticaja* - Jedan od najvažnijih izazova u održavanju softvera je otkriti efekte predložene modifikacije na ostatak sistama. Analiza uticaja je radnja procene verovatnih efekata promene sa planom smanjenja iznenadnih neželjenih efekata. Zadatak uključuje procenu ispravnosti projektovane modifikacije i procenu rizika vezanih za njen završetak, plus procene efekata na imovinu, energiju i razvoj.
- *Korektivne promene* - jedno od ključnih pitanja su korektivne promene jer je teško pronaći pravo mesto za izvršenje promena. Ako se preliminarni dizajn smanji, mala promena može insistirati na promenama arhitekture koje zahtevaju mnogo vremena. Ako postoji potpuno rešenje za jedan problem, onda je sledeći teže rešiti. Greške u dizajnu je teško popraviti jer je potrebno mnogo vremena i razumevanja celokupne baze koda.
- *Prilagodljive promene* često nisu lake zbog nedostatka informacija o tome u šta se softver menja. Takođe je teško analizirati uticaj i otkriti interfejse za nove stvari. Problemi zbog neuravnoteženog idejnog projekta su zabrinjavajući.
- *Razumevanje programa* podrazumeva da inženjeri za održavanje treba da potroše mnogo vremena da pročitaju i razumeju kod i relevantnu dokumentaciju kako bi imali bolju perspektivu o njegovoj logici, svrsi i strukturi kako bi održavali deo softvera i poboljšali kvalitet softvera.

# Poglavlje 6

# Izazovi pri upravljanju kvalitetom i testiranju

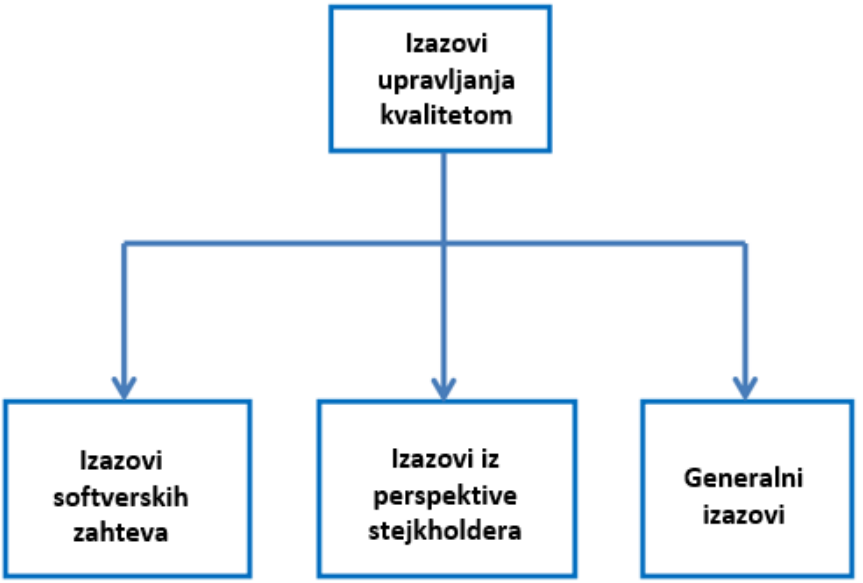
## KATEGORIJE IZAZOVA PRILIKOM UPRAVLJANJA KVALITETOM

*Izazovi softverskih zahteva, zainteresovanih strana i generalni izazovi predstavljaju tri glavne kategorije izazova prilikom upravljanja kvalitetom softvera*

Svi izazovi u softverskom inženjerstvu koji se odnose na upravljanje kvalitetom i testiranje mogu se svrstati u tri glavne kategorije, dok svaka od kategorija sadrži više vrsta izazova.

Izazovi koji će biti navedeni u nastavku direktno ili indirektno utiču na ukupnu produktivnost softverske kompanije. Neki od njih utiču na rast kompanije odmah, dok neki postaju vidljivi nakon prolaska ili nakon implementacije projekta.

S obzirom da je o izazovima koje donose promene zahteva već bilo reči u okviru ove lekcije, fokus ovog objekta učenja biće izazovi iz perspektive zainteresovanih strana (stejkholdera) i generalni izazovi upravljanja kvalitetom softvera.



Slika 6.1 Kategorije izazova upravljanja kvalitetom u softverskom inženjerstvu [Md. Shahadat Hossain, Challenges of software quality assurance and testing, IJSECS, ISSN: 2289-8522, Volume 4 Issue 1, 2018]

# IZAZOVI IZ PERSPEKTIVE ZAINTERESOVANIH STRANA

## *Postoje 4 tipa zainteresovanih strana koje mogu imati uticaj na kvalitet softvera*

Različite zainteresovane strane mogu biti odgovorne za probleme kvaliteta. U nastavku su date 4 vrste ključnih zainteresovanih strana.

- *Izazovi sa aspekta programera*

Kako je osnovna uloga programera da završe kodiranje, oni ponekad smatraju da će kod sigurno funkcionisati, pa ga isporučuju i bez testiranja, ali kasnije dobijaju mnogo izveštaja o greškama. Istina je da programeri ne izdvajaju dovoljno vremena za testiranje jer imaju više zadataka za realizaciju. Dodatno, neki programeri samo prolaze kratak kurs obuke od 3-6 meseci i uče specifični programski jezik a zatim počinju sa programiranjem. S obzirom da oni ne mogu da naprave ispravnu logiku programiranja, kao rezultat kada krajnji korisnik unese izuzetne podatke dolazi do logičke greške. Loše logičko kodiranje programera uzrokuje nedostatak kvaliteta softvera.

- *Rizici sa stanovišta kompanije*

Softverske kompanije žure da ispoštuju rok i ne žele da obezbede dovoljno vremena za testiranje. Kažu da ako radi, onda ćemo isporučiti ostatak testiranja tokom perioda podrške i održavanja ili sledećeg izdanja. Neke od softverskih kompanija nemaju ni jednog inženjera za obezbeđenje kvaliteta.

- *Pretnja od strane korisnika*

Klijenti se vrlo često fokusiraju na troškove smatrajući da će kvalitetniji softver koštati više. S tim u vezi, oni traže softver koji će samo obavljati njihove zahteve prilikom svakodnevnih aktivnosti, ali ne žele kvalitetan softver.

- *Nemar sa strane prodavca*

Nekada zbog nemara dobavljača kvalitet softvera ne uspeva. Prodavci implementacije treće strane ponekad ne održavaju kvalitet softvera, već samo žure da završe implementaciju.

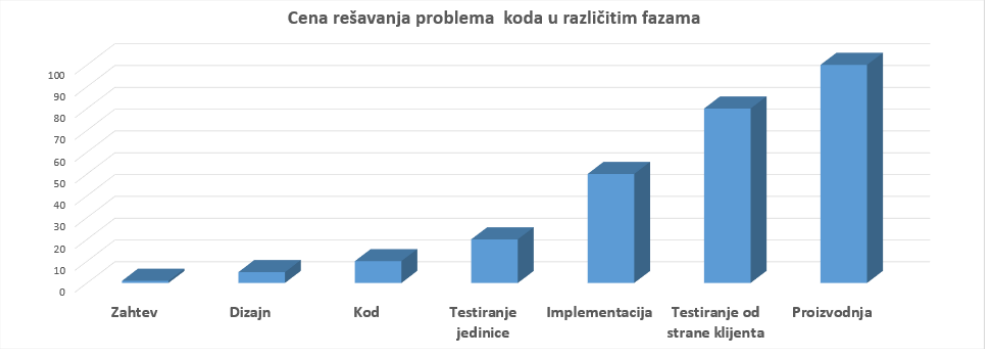
# GENERALNI IZAZOVI

## *Rano otkrivanje grešaka je veoma bitno jer cena njihovog otklanjanja raste kroz faze projekta*

*Ekonomski izazovi*

Ekonomski uticaji na osiguranje kvaliteta softvera su visoki. Cena kvaliteta i otklanjanja grešaka varira od faze do faze. U početnoj fazi cena ispravljanja grešaka je niža, dok kasnije cena raste. Zato je bitno fokusiranje na rano otkrivanje grešaka i hihovo ispravljanje. Na slici ispod vidi se promena cene koštanja rešavanja problema u različitim fazama razvoja softvera.

- Rešavanje problema u fazi zahteva košta 1.
- Rešavanje problema u fazi razvoja košta 10.
- Rešavanje problema u fazi testiranja košta 100.
- Rešavanje problema u fazi proizvodnje košta 1000.




Slika 6.2 Cena rešavanja problema koda u različitim fazama projekta [Md. Shahadat Hossain, Challenges of software quality assurance and testing, IJSECS, ISSN: 2289-8522, Volume 4 Issue 1, 2018]

# IZAZOVI OGRANIČENIH RESURSA I RADNOG OKRUŽENJA

## *Izazovi okruženja (hardver, radno i testno okruženje) predstavljaju veliki problem za obezbeđivanje kvaliteta softvera*

- *Izazov ograničenih resursa* je uobičajeni problem za skoro sve kompanije za razvoj softvera širom sveta.

U nekim firmama jedna osoba završava ukupan životni ciklus razvoja softvera za određeni projekat. Kao rezultat, jedan ili više koraka nedostaje svesno ili nesvesno. Zbog ograničenja ljudskih resursa, jedan profesionalac radi na više projekata sa više vrsta zadataka, bilo da je profesionalni stručnjak ili početnik u ovoj oblasti. Ova praksa uzrokuje ozbiljan nedostatak kvaliteta. Ograničenje drugih resursa takođe direktno ili indirektno utiče na kvalitet softvera, na primer hardverska oprema, uređaji, alat za testiranje, licencni proizvod itd.

- *Izazovi okruženja*

Ovo je veliki problem za osiguranje kvaliteta softvera i testiranje, kao i razvoj softvera. U nastavku su neki od izazova okruženja:
A) Hardversko okruženje - Vitalni faktor za razvoj i testiranje softvera. Primarni server, sekundarni server, aplikacija i server izveštaja se razlikuju od razvojnog okruženja do okruženja za testiranje i proizvodnog okruženja. Nedovoljan hardver i nedostatak drugih uređaja ometaju proces razvoja i testiranja. Integracija hardvera, mašina, uređaja i druge

opreme koju korisnici koriste i API-ja koji je potreban za integraciju sa aplikacijom treće strane, kao i za pružanje informacija za simulaciju, predstavlja kritičan izazov okruženja.
B) Radno okruženje - Još jedno vitalno pitanje za razvoj i testiranje softvera. Zagušena i nezdrava kancelarijska prostorija, neudobni sto i stolice i drugi nameštaj ne samo da predstavljaju izazov za održavanje kvaliteta, već stvaraju i ozbiljnu bolest kod profesionalaca. Softverske firme malih i srednjih preduzeća ne mogu da obezbede odgovarajuće radno okruženje za razvoj i za testiranje.
C) Testno okruženje: Testno okruženje zahteva realno podešavanje ciljanog okruženja, ali za različite vrste ograničenja mnogo puta je teško obezbediti stvarno okruženje proizvodnje za testiranje. Zbog ovog ograničenja tester ne može da kreira stvarni test slučaj, podatke o testiranju i izuzetne scenarije. Test inženjer takođe ima ograničenje da unapred napiše izuzetan test slučaj koji se može dogoditi nekoliko godina kasnije.

# OSTALI GLOBALNI IZAZOVI

## *Tu spadaju izazovi veličine projekta, broja linija, promene tehnologija, vremena i pravnih okvira*

- *Izazovi veličine projekta*

Ako je projekat preveliki onda je teško i uspostaviti njegovo upravljanje. Sa druge strane, ako je projekat premali on dobija manji značaj. Ponekad softverska kompanija ne pridaje važnost projektu male veličine ili niske cene. U oba slučaja kvalitet trpi.

- *Izazov broja linija koda*

Veličina softverskih proizvoda se više ne meri u hiljadama linija koda, već u milionima, zbog čega proces testiranja trpi. Veb aplikacije suočavaju se sa većom složenošću za testiranje linije koda zbog preuzimanja podataka iz velikih baza podataka.

- *Izazov promene tehnologije*

Tehnologija se menja i unapređuje svaki dan. Sa promenom tehnologije, ako se softver ne može promeniti istovremeno, u tom slučaju kvalitet pati. U mnogim slučajevima softver više verzije ne podržava nižu verziju. Prelazak sa starog sistema na novi sistem postavlja pitanje kvaliteta. Testiranje trpi ukoliko se tester ne može brzo prilagoditi novoj tehnologiji.

- *Izazovi vremenskog okvira*

Pravi scenario je onaj u kome se testiranje vrši u okviru trajanja razvoja. Nedovoljno vreme za testiranje izaziva ozbiljan nedostatak kvaliteta u sledećoj fazi.

- *Političke i pravne promene*

Odluke vlade ponekad direktno ili indirektno utiču na zahteve softvera, razvoj softvera i osiguranje kvaliteta. Primer je digitalizacija zastarelog sistema.

# Poglavlje 7

# Vežba - Pokazni primeri

## OBJAŠNJENJE PRAKTIČNE VEŽBE SA PRIMEROM AIRBUS-OVOG PROJEKTA

*Kroz teme za diskusiju je neophodno proći za sve navedene primere - 90 minuta*

U okviru dva časa vežbi u trajanju od 90 minuta studentima treba prikazati primere problema u softverskim projektima koji su nastali zbog izazova navedenih u lekciji.

Teme za diskusiju za svaki od navedenih primera:

1. Šta, prema vašem mišljenju, predstavlja glavni uzrok problema u datom primeru? Koja je to aktivnost ili nedostatak aktivnosti?
2. Koji od izazova navedenih u lekciji prepoznajete u datom primeru?
3. Na koji način su problemi projekta mogli biti sprečeni?

- **Airbus A380**

Primer:

Kada je Airbus A380 lansiran 2007. godine, mnogo se očekivalo od aviona, ali samo 10 godina kasnije, prodavali su se samo za rezervne delove. Očekivana promena igre dovela je do toga da se Erbas mučio da obezbedi poslove sa avio-kompanijama. A380 je bio skup za proizvodnju, a Airbus-ovi **proizvodni timovi nisu komunicirali i koristili su različite CAD programe**. Samo ta greška koštala je 6 milijardi dolara. Štaviše, tržište polovne robe nije postojalo jer su avioni jednostavno bili preveliki da bi bilo koja aviokompanija vratila uloženi novac.

Uzrok problema:

Uticaj loše interne komunikacije

## SAINSBURIJEVA AUTOMATIZACIJA SKLADIŠTA I AŽURIRANJE SOFTVERA NEST-A

*U svim primerima moguće je prepoznati po jedan izazov naveden u teorijskom delu lekcije*

- **Sainsburijeva automatizacija skladišta**

Primer:
Britanski gigant supermarketa započeo je ogroman projekat instaliranja automatizovanog sistema skladištenja u njihovom distributivnom centru Valtham Point. Opslužujući veći deo Londona i jugoistoka, sistem zasnovan na bar kodovima trebalo je da pojednostavi operacije i poboljša efikasnost.
Od početne instalacije 2003. godine, postojale su ozbiljne greške u čitanju bar kodova. Bez obzira na to, Sainsburi je tvrdio da je sistem funkcionisao kako je planirano 2005. Dve godine kasnije, odlučili su da odustanu od celog projekta, otpisavši više od 150 miliona funti troškova.

Uzrok problema:

Zanemarivanje testiranja u početnim fazama projekta

- **Ažuriranje Nest-ovog softvera**

Primer:

2016, Nest (pametni termostat koji pokreće Google) objavio je katastrofalno ažuriranje softvera koje je bukvalno ostavilo kupce na hladnoći. Greška u ažuriranju je dovela do toga da se baterije uređaja isprazne, zbog čega termostat nije mogao da kontroliše temperaturu ostavljajući kupce bez grejanja ili tople vode usred zime. Nije bilo potrebno mnogo vremena za izbacivanje novog ažuriranja rešivši problem za većinu korisnika, ali ovaj neuspeh značajno je uticao na celokupan imidž kompanije.

Uzrok problema:

Nedovoljno testiranje u fazi razvoja

# PROJEKAT „VIRTUELNA OGRADA“ MINISTARSTVA ZA UNUTRAŠNJU BEZBEDNOST SAD

## *U okviru primera potrebno je pronaći dva osnovna izazova koji su doveli do problema*

Primer:

Ministarstvo za unutrašnju bezbednost SAD 2006. godine donelo je odluku da pojača graničnu patrolu SAD-a mrežom radara, satelita, senzora i komunikacionih veza, što se zajedno naziva „virtuelna ograda“. U septembru 2006. godine, ugovor za ovu mrežu inicijative za bezbednu granicu (SBInet) dodeljen je Boingu, koji je dobio 20 miliona dolara za izgradnju pilot deonice od 28 milja duž granice između Arizone i Meksika. Međutim, početkom 2008. godine Kongres je saznao da se pilot projekat odlaže jer su **korisnici bili isključeni iz procesa i da je složenost projekta potcenjena**. U februaru 2008. godine, Vladina kancelarija za odgovornost je izvestila da bi radar za otkrivanje migranata koji prelaze granicu mogao biti aktiviran kišom i drugim vremenskim prilikama, a slike sa kamere bi morale da budu zumirane jer su beskorisno niske rezolucije za objekte preko 3,1 milje. Takođe, pilotov komunikacioni sistem je ometao WiFi mreže lokalnih stanovnika. Sve navedeno nije bio dobar početak ovako velikog projekta. U aprilu 2008. DHS je objavio da nadzorni tornjevi pilotske ograde nisu ispunili ciljeve Granične patrole i da su zamenjeni. Sve navedeno, kao i mnogobrojni dodatni

problemi doveli su do značajno kašnjenja projekta i konačne odluke da on bude zaustavljen 2011. godine.

Uzroci problema:

- Isključenost korisnika iz procesa
- Preveliki, odnosno veoma složen projekat

# Poglavlje 8

# Vežba - zadaci

## ZADACI ZA INDIVIDUALNI RAD STUDENATA

### *Zadaci za samostalno rešavanje na vežbi i kod kuće*

Posle predavanja a pre časova vežbanja (održavaju se dva dana kasnije), poželjno je da pokušate da rešite neke od ovih zadatka radom kod kuće i da rezultate pošaljete mejlom, preko Zimbre, saradniku koji drži vežbe, najkasnije jedan sat pre održavanja vežbi. Ako neki zadatak nije urađen pre ili za vreme vežbi, preporučuje se studentu da ih uradi posle vežbi, kod kuće.

Obim zadatka može da zahteva najviše 30 do 40 minuta rada studenta. Svaki student mora da ima poseban zadatak, a na temu koja se ovde definiše. Studenti koji pošalju identična ili vrlo bliska rešenja, neće im ta rešenjabiti prihvaćena, tj. dobiće 0 poena.

Tekst domaćeg zadatka:

Odaberite sopstveni primer (različit za svakog studenta) softverskog projekta koji nije uspeo ili je imao ozbiljne probleme zbog nekog od izazova navedenih u lekciji, definišite koji od izazova prepoznajete i zašto i objasnite na koji način je neuspeh mogao biti izbegnut.

# Poglavlje 9

# Zaključak

## ZAKLJUČAK

### *Sumiranje stečenih znanja*

- Mobilno računarstvo i računarstvo u oblaku kao dva nova trenda u softverskom inženjerstvu menjaju ponašanje i tokove rada potrošača i IT odeljenja preduzeća
- Testiranje mobilnih aplikacija je sve teže kada se ima u vidu da povezanost i ambijentalna lokacija korisnika značajno utiču na njihovo iskustvo korišćenja
- Izazovi koji utiču na razvoj softvera u oblaku su sigurnost i privatnost osetljivih podataka i uparvljanje pristupom
- Postoji više razloga zbog kojih softver odgovara promenama zahteva. Shodno tome, nedostatak odražavanja promena na softveru ometa zadovoljstvo korisnika i blokira kontinuirani napredak funkcionalnosti softvera.
- RCM je definisan kao „procedura upravljanja promenama u zahtevima tokom procesa inženjeringa zahteva i razvoja sistema“.
- Ključni izazovi za upravljanje promenama su: ponovna upotreba, merenje aktivnosti promene, povezanost sa softverskim artefaktima, predviđanje promena i automatizacija upravljanja promenama

## LITERATURA

### *Pogodna literatura za učenje*

**Obavezna literatura:**

1. Nastavni materijal za e-učenje na predmetu SE101 Razvoj softvera i inženjera softvera, Univeziitet Metropolitan, školska 2022/23. godina
Nastavni materijal je pripremljen korišćenjem referenci 2-8.

**Dodatna literatura:**

- E. Michael Maximilien, Pedro Campos, Facts, Trends and Challenges in Modern Software Development, Int. J. Agile and Extreme Software Development, Vol. 1, No. 1, 2012
- Md. Shahadat Hossain, Challenges of software quality assurance and testing, IJSECS, ISSN: 2289-8522, Volume 4 Issue 1, 2018
- Juyun Joey Cho, An exploratory study on issues and challenges of agile software development with Scrum, Utah State University, Logan, Utah, 2010

- Hussin Ahmed, Azham Hussain, Fauziah Baharom, Current Challenges of Requirement Change Management, School of Computing, Universiti Utara Malaysia, 06010 UUM, Kedah, Malaysia, 2016.
- Wen-MingHanSun-JenHuang, An empirical analysis of risk components and performance on software projects, ELSEVIER, Journal of Systems and Software, Volume 80, Issue 1, January 2007
- Juyun Cho, Globalization and global software development, Utah State University, Volume VIII, No. 2, 2007
- Aakriti Gupta, Shreta Sharma, Software Maintenance:Challenges and Issues, IJCSE, Vol. 4 No.01, Jan 2015
- Ajmal Iqbal, Syed Shahid Abbas, Communication Risks and Best practices in Global Software Development, School of Computing, Blekinge Institute of Technology, Sweden, 2011