UNIVERZITET METROPOLITAN BEOGRAD

SE322 - INŽENJERSTVO ZAHTEVA

Uvod u inženjerstvo zahteva

Lekcija 01

PRIRUČNIK ZA STUDENTE

SE322 - INŽENJERSTVO ZAHTEVA

Lekcija 01

UVOD U INŽENJERSTVO ZAHTEVA

- Uvod u inženjerstvo zahteva
- Poglavlje 1: Studija slučaja: Sistem za praćenje hemikalija
- Poglavlje 2: Definisanje softverskih zahteva
- Poglavlje 3: Razvoj zahteva i upravljanje zahtevima
- Poglavlje 4: Problemi vezani za zahteve sistema
- Poglavlje 5: Značaj inženjerstva zahteva
- Poglavlje 6: Vežba
- Poglavlje 7: Domaći zadatak
- Poglavlje 8: Projektni zadatak
- Zaključak

UVOD

Šta ćemo naučiti u ovoj lekciji?

Cilj ovog predmeta je da ukaže na značaj softverskih zahteva u procesu razvoja softverskih proizvoda i predstaviti proces inženjeringa zahteva kojim su obuhvaćeni izbor, analiza, validacija i upravljanje zahteva za izgradnjom složenih softverskih sistema. Prvih nekoliko predavanja je fokusirano na pitanje **„šta”** je obuhvaćeno procesom inženjeringa zahteva dok će u kasniji predavanjima biti reči o tome **„kako”** se u okviru svakog od ovih procesa mogu primeniti specifične tehnike.

U ovom predavanju će biti dati odgovori na “pitanja koja se najčešće postavljaju ” (FAQ) o zahtevima. Zatim se opisuju problemi koji se mogu javiti kada zahtevi sistema nisu dobri i razlika između proizvodnih i procesnih zahteva. U okviru studije slučaja dat primer koji pokazuje kako se specificiranim zahtevima mogu rešiti odgovarajući problemi.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 1

Studija slučaja: Sistem za praćenje hemikalija

VIDEO PREDAVANJE ZA OBJEKAT "STUDIJA SLUČAJA: SISTEM ZA PRAĆENJE HEMIKALIJA"

Trajanje video snimka: 34min 40sek

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

PRAĆENJE HEMIKALIJA U CONTOSO PHARMACEUTICALS

Sistem za praćenje hemikalija komunicira sa korisnicima sistema i drugim sistemima

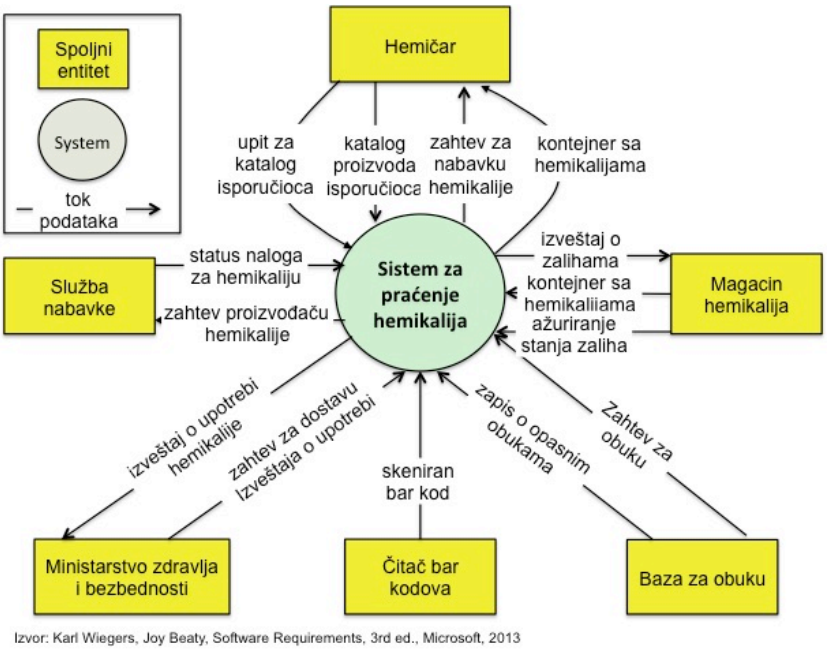
Da bismo ilustrovali metode za utvrđivanje, analizu i specifikaciju zahteva koje softverski sistem treba da ostvari, u predavanjima koristimo primer firme Contoso Pharmaceuticals koja koristi *Sistem za praćenje hemikalija.* To je deo informacionog sistema firme koji se odnosi na upravljanje hemikalijama koje firma koristi. U predavanjima se analiziraju zahtevi koje softverski podsistem, koji čini ovaj deo informacionog sistema firme, treba da zadovolji. Ovu studiju slućaja ćemo koristiti u svakoj lekciji, gde će se problem praćenja nabavke i korišćenja himikalija u ovoj firmi analizirati iz različitih ugla gledanja, te će se zbog toga koristiti različite vrste dijagrama.

Slika 1 pokazuje Sistem za praćenje hemikalija u vidu tzv. kontekstnog dijagrama. Ceo Sistem za praćenje hemikalija je predstavljen kao jedan krug; kontekstni dijagram ne omogućava vidljivost u unutrašnjim objektima, procesima ili podacima sistema. Sistem u krugu može obuhvatiti bilo koju kombinaciju softvera, hardvera i ljudskih komponenti.

Spoljni entiteti u pravougaonicima mogu predstavljati klase korisnika (Hemičar, Kupac), organizacije (Odeljenje zdravlja i bezbednosti), druge sisteme (baza podataka za obuku) ili hardverske uređaje (čitač bar kodova). Strelice na dijagramu predstavljaju protok podataka (kao što je zahtev za hemikalije) između sistema i njegovih eksternih entiteta.

Na ovom dijagramu se dobavljači hemijskih proizvoda prikazuju kao eksterni entitet. Na kraju, kompanija će uputiti naloge prodavcima na ispunjenje, dobavljači će poslati kontejnere za hemikalije i fakture kompaniji Contoso Pharmaceuticals, a odeljenje za nabavku kompanije

platiće prodavcima. Međutim, ti se procesi odvijaju izvan okvira sistema za praćenje hemikalija, kao deo operacija odeljenja za nabavku i prijem.



Izvor: Karl Wiegers, Joy Beaty, Software Requirements, 3rd ed., Microsoft, 2013

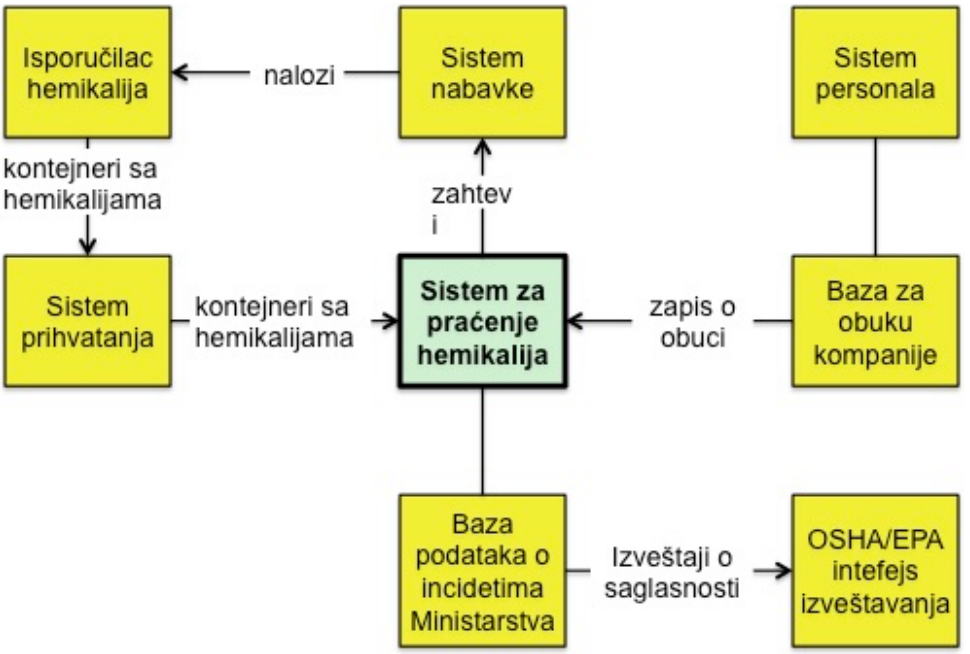
Slika 1.1 Kontekstni dijagram Sistema za praćenje hemikalija

INTERFEJSI SISTEMA ZA PRAĆENJE HEMIKALIJA

Preko interfejsa Sistem za praćenje hemikalija komunicira sa drugim sistemima

Slika 2 je mapa ekosistema za Sistem praćenja hemikalija. Eko sistem sadrži druge sisteme sa kojima je Sistem za praćenje hemikalija povezan. Svi sistemi prikazani pravougaonicima (kao što su Sistem nabavke ili Sistem prjema) su sistemi sa kojim Sistem za praćenje hemikalija komunicira, ali nisu deo tog sistema. Sistem za praćenje hemikalija je prikazan sa podebljanim okvirom. Linije koje povezuju prikazane softverske sisteme prikazuju interfejse između sistema (na primer, interfejs sistema nabavke za Sistem za praćenje hemikalija). Linije sa strelicama i nazivima interfejsa pokazuju da glavni delovi podataka prelaze iz jednog sistema u drugi (na primer, „zapis o obuci“ se prenosi iz korporativne baze podataka o obuci u Sistem za praćenje hemikalija).

U lekcijama koje slede, postepeno ćemo utvrditi procese koje treba da sadrži Sistem za praćenje hemikalija i zahteve koji on treba da zadovolji.



Izvor: Karl Wiegers, Joy Beaty, Software Requirements, 3rd ed., Microsoft, 2013

Slika 1.2 Sistemi u okruženju sa kojima Sistem za praćenje hemikalija komunicira

Poglavlje 2

Definisanje softverskih zahteva

ŠTA SU ZAHTEVI?

Zahtevi određuju svrhu i ponašanje sistema. Mogu biti prikazani na tri nivoa: poslovni zahtevi, zahtevi korisnika i funkcionalni zahtevi.

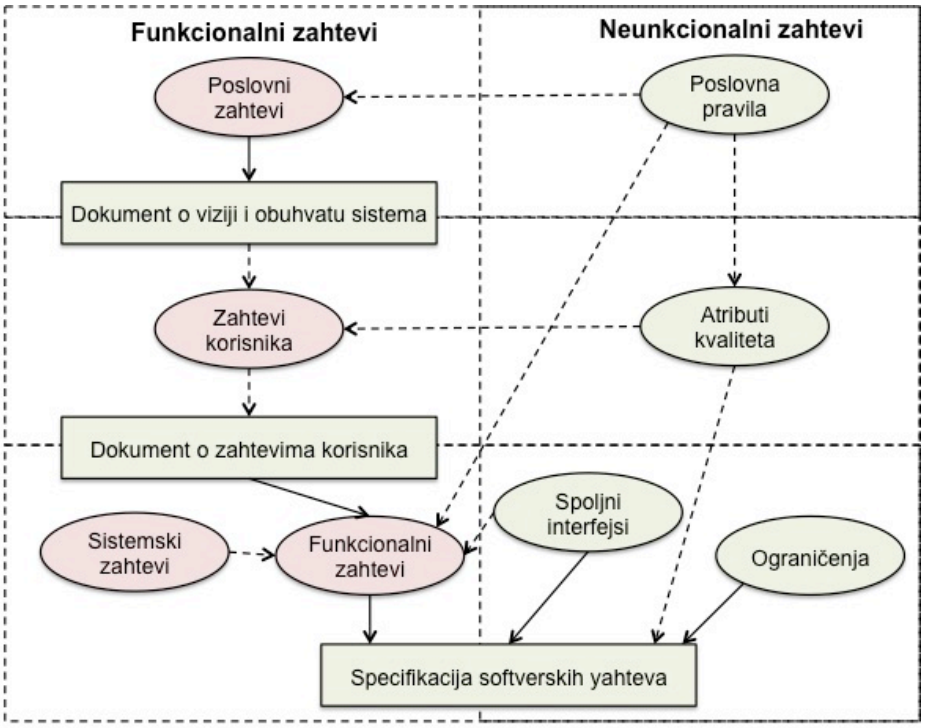
Zahtevi predstavljaju specifikaciju nečega što treba da bude primenjeno. Oni opisuju kako bi sistem trebalo da se ponaša, ili opis nekog svojstva sistema ili njegovog atributa. Oni ograničavaju, ali i usmeravaju proces razvoja sistema. Ovde pod sistemom podrazumevamo softverski sistem, tj. softver.

Postoji veliki broj vrsta informacija koje predstavljaju zahteve, one se klasifikuju prema određeni kriterijumima, i to na tri nivoa:

- poslovni zahtevi
- zahtevi korisnika, i
- funkcionalni zahtevi.

Pored ovoga, postoje i tzv. nefunkcionalni zahtevi.Na slici 1 je prikazan jedan od modela klasifikacije zahteva, koji nije sveobuhvatan, ali pomaže u razumevanju postojanja različitih vrsta zahteva. Linije sa punom linijom i strelicom označavaju dokument u koji se smeštaju kreirani zahtevi,a isprekidane linije sa strelicom označavaju izvor informacije o zahtevu. Zahtevi i njihovi izvori označeni su elipsama, a pravouganicima su označeni dokumenti koji sadrže različite vrste zahteva

Slika ne prikazuje zahteve o podacima, jer se oni pojavljuju na sva tri nivoa.



Preuzetoi iz: Karl Wiegers, Joy Beaty, Software Requirements, 3rd ed., Microsoft, 2013

Slika 2.1 Veze između vrste zahteva i njihovih izvora

VRSTE ZAHTEVA

Poslovni zahtevi, zahtevi korisnika, funkcionalni zahtevi, sistemski zahtevi, nefunkcionalni zahtevi

Poslovni zahtevi (budiness requirements) objašnjavaju **ZAŠTO organizacija primenjuje sistem, koje poslovne želi da ostvari**. Fokus je na poslovnim ciljevima. Poslovni zahtevi se unose u *Dokument o viziji i obuhvatu sistema.* Podrazumeva se da je organizacija, pre početka rada na projektu razvoja sistema, utvrdila poslovne potrebe ili mogućnosti tržišta koje sistem treba da zadovolji.

Zahtevi korisnika (user requirements) opisuju **ciljeve ili zadatke koje korisnici moraju da izvrše na sistemu da bi obezbedili nekome neku vrednost.** Zahtevi korisnika takođe uključuju opise atributa sistema ili njegove karakteristike koje su od važnosti za korisnika. Zahtevi korisnika se izražavaju u vidu *slučajeva korišćenja* (use cases), *priča korisnika* (user stories) i *tabela događaj-odgovor* (event-response table).

Funkcionalni zahtevi (functional requirements) **određuju ponašanje sistema pod određenim uslovima.** Oni opisuju ŠTA inženjer razvoja treba da uradi da bi omogućui korisniku da izvrši svoj zadatak. Oni se obično specificiraju iskazima u kojima se korsiti reč "mora".

Analitičar poslovanja (business analyst) na osnovu razgovora sa korisnicima i inženjerima razvoja prevodi zahteve korisnika u funkcionalne zahteve i ciljeve kvaliteta. On definiše dokument:Specifikacija softverskih zahteva (Software Requirement Specification - SRS), koji opisuje, koliko je to potrebno, očekivano ponašanje softverskog sistema.

Sistemski zahtevi (system requirements) opisuju zahteve za sistem koji važe za više komponenata ili podsistema. Sistem može da bude čisto softverski sistem, ali može da, pored softvera, uključi i hardver. Ljudi i procesi su takođe deo sistema, tako da neki sistemski zahtevi mogu da se odnose i na ljude koji rade u okviru sistema.

Poslovna pravila (business rules) sadrže poslovnu politiku i pravila poslovanja organizacije, industrijske standarde i algoritme računanja. Poslovna pravila nisu softverski zahtevi jer ona postoje nezavisno od softvera. Međutim, ona često bitno određuju ponašanje sistema, njegovu funkcionalnost, a u skladu sa određenim pravilima.

Pored funkcionalnih, postoje i nefunkcionalni zahtevi:

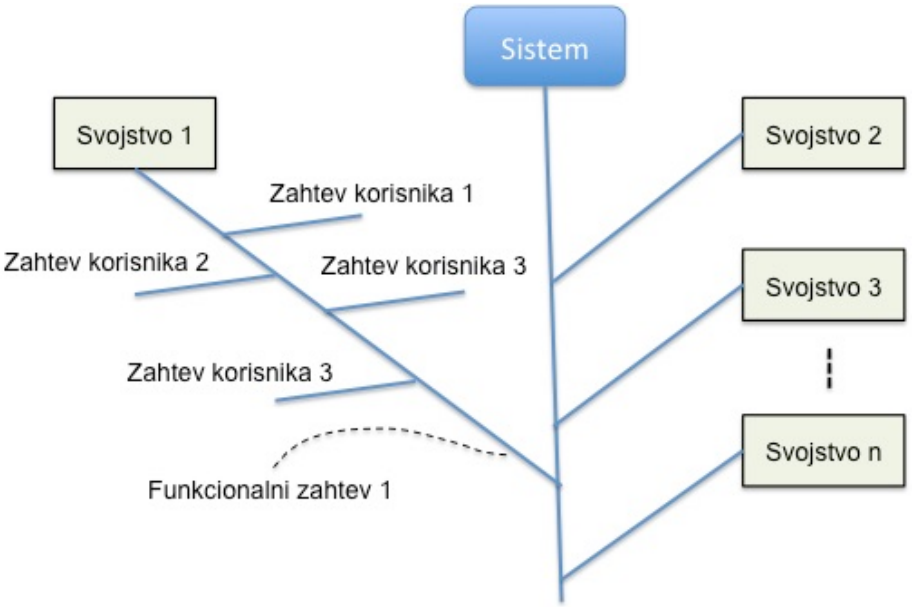
- Atributi kvaliteta (quality attributes) su faktori kvaliteta ili zahtevi za kvalitetom servisa, i opisuju karakteristike sistema u različitim dimenzijama koje su značajne ili za korisnike, ili za inženjere razvoja i održavanja, kao što su performanse, bezbednost, raspoloživost i prenosivost.
- Spoljni interfejsi (external interfaces) opisuju veze između sistema i spoljnjeg okruženja, kao što su veze sa drugim softverskim sistemima, hardverskim komponentama, korisnicima i komunikacionim interfejsima.
- Ograničenja (constraints) koja ograničavaju svojstva sistema.

SVOJSTVA SISTEMA

Svojstvo sadrži jednu ili više mogućnosti sistema koje obezbeđuje neku vrednost korisniku sistema, a opisuje se skupom funkcionalnih zahteva.

Svojstvo (feature) sadrži jednu ili više mogućnosti sistema koje obezbeđuju neku vrednost korisniku sistema, a opisuje se skupom funkcionalnih zahteva. Lista svojstava sistema koju želi korisnik nije identična potrebama rada i korišćenja korisnika. Jedno svojstvo može da obuhvati zahteve više korisnika, od kojih se svaki odnosi na određene funkcionalne zahteve koji mora da budu ispinjeni da bi korisnik mogao da uradi neki zadatak, definisan u zahtevu korisnika.

Slika 2 prikazuje stablo svojstava nekog sistema ili nekog njegovog dela, koje kao model analize, daje hijerarhijsku strukturu složenog svojstva koju čine manja, pod-svojstva koja se odnose na specifične zahteve korisnika i koji vode ka skupu funkcionalnih zahteva.



Slika 2.2 Veza svojstva sistema i zahteva korisnika i funkcionalnih zahteva [Autor]

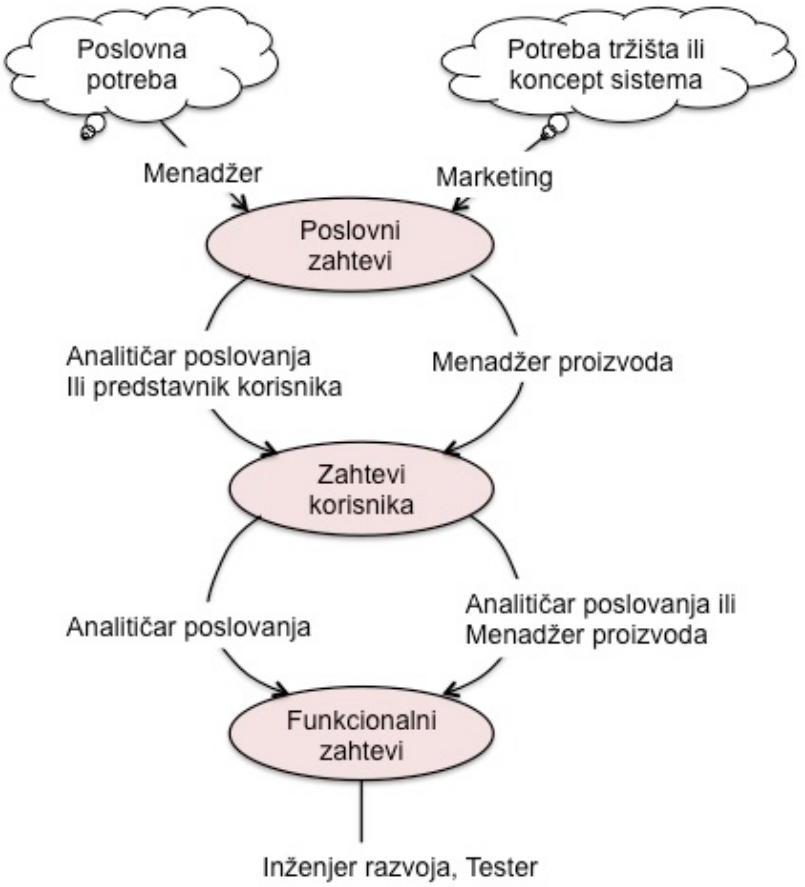
ULOGE AKTERA U RAZVOJU FUNKCIONALNIH ZAHTEVA

Akteri sa različitim ulogama učestvuju u definisanju funkcionalnih zahteva, na osnovu postavljenih poslovnih zahteva i zahteva korisnika.

Slika 3 prikazuje kako razni akteri u razvoju softvera mogu da učestvuju u utvrđivanju zahteva na navedena tri nivoa. Nazivi uloga koje dobijaju akteri zavise od organizacija, te se mogu razlikovati.

Zavisno od utvrđenih potreba poslovanja, kao i potreba tržišta, ili od postavljenog koncepta novog proizvoda, **menadžeri** ili **marketing** definišu poslovne zahteve za softver koji bi trebalo da učine kompanije efiksnijim (za slučaj informacionih sistema), ili uspešnim na tržištu (za slučaj softverskog proizvoda). **Analitičar poslovanja** radi sa predstavnikom korisnika s ciljem utvrđvanja zahteva korisnika. Svaki zahtev korisnika i svojstvo mora da bude u saglasnosti postavljenim poslovnim zahtevima. U skladu sa zahtevima korisnika, **poslovni analitičar** ili **menadžer proizvoda** definiše funkcionalnost koja će omogućiti korisnicima da ostvare svoj cilj. **Inženjeri razvoja**, koristeći postavljene funkcionalne i nefunkcionalne zahteve, projektuju rešenja u saglasnosti sa postavljenim ograničenjima. **Testeri** određuju kako da se izvrši provera da li su zahtevi korektno primenjeni.

Važno je da se zahtevi zapišu na odgovarajući način u dokumentu ili na nekom medijumu tako da ga mogu koristiti svi u razvojnom timu. Na slici 1 prikazana su tri dokumenta koja se mogu pripremiti na osnovu utvrđenih zahteva. U slučaju manjih projekata, ova tri dokumenta se mogu spajati u dva ili u samo jedan dokument.



Preuzetoi iz: Karl Wiegers, Joy Beaty, Software Requirements, 3rd ed., Microsoft, 2013

Slika 2.3 Uloga aktera u određivanju posebnih vrsta zahteva

KO DONOSI ODLUKE?

Poslovne odluke o prihvatanju predloga treba da donose oni koji mogu da sagledaju posledice na troškove u vremenu i u resursima, kao i na eventualne kompromise sa drugim zahtevima.

Na slici 1, prikazanoj ranije u ovom poglavlju, utvrđena su tri glavna rezultata: dokument o viziji i opsegu, dokument o zahtevima korisnika i specifikacija softverskog zahteva. Ne morate nužno da kreirate tri diskretna rezultata za svaki projekat. Često ima smisla kombinovati neke od ovih informacija, posebno o malim projektima. Međutim, priznajte da ova tri rezultata sadrže različite informacije, razvijene u različitim tačkama projekta, moguće od strane različitih ljudi, sa različitim svrhama i ciljanom publikom.

Model na slici 1 prikazao je jednostavan tok informacija o zahtevima odozgo na dole. U stvarnosti, trebalo bi da očekujete cikluse i iteracije među poslovnim, korisničkim i funkcionalnim zahtevima. Kad god neko predloži novu funkciju, zahtev korisnika ili malo funkcionalnosti, analitičar mora da pita: „Da li je ovo u okviru projekta?“ Ako je odgovor „da“, zahtev spada u specifikaciju. Ako je odgovor "ne", to neće, barem ne za naredno izdanje ili ponavljanje. Treći mogući odgovor je „ne, ali podržava poslovne ciljeve, pa bi to trebalo i biti.“ U tom slučaju, ko kontroliše obim projekta - sponzor projekta, menadžer projekta ili vlasnik

proizvoda - mora odlučiti da li će povećati trenutni okvir projekta ili iteracije u skladu sa novim zahtevom. .

Ovo je poslovna odluka koja ima posledice na plan i budžet projekta i može zahtevati kompromise sa drugim mogućnostima.

U procesu utvrđivanja zahteva, često se javljaju novi zahtevi i vrlo je važno postaviti jedan efikasni proces promena i dopuna prethodno utvrđenih zahteva. Zato je važno da u ovom procesu, odluke donose pravi ljudi, tj. oni koji donose poslovne odluke o tome koje promene treba da se prihvate, a koje se odbacuju, imajući u vidu posledice na troškove u vremenu, resursima i na kompromise koje eventualno izazivaju.

ŠTA SU SISTEMSKI A ŠTA SOFTVERSKI ZAHTEVI?

Dok sistemski zahtevi opisuju ponašanje sistema posmatrano spolja sa strane korisnika, softverski zahtevi služe za komunikaciju sa programerima

Sistemski zahtevi opisuju ponašanje sistema posmatrano spolja , na primer, sa strane korisnika koji služe kao sredstvo za delimičnu komunikaciju sa tehnički obrazovanijim korisnicima , ili nalazenje organizacije, analitičara i projektanata koji se bave razvojem elemenata ili komponenata sistema.
Zahtevi za elemente ispod nivoa sistema (softverski zahtevi), bez obzira da li su to elementi koji se odnose na hardver ili softver, ili i na hardver i softver, su obično su od minimalnog interesa za korisnike. Softverski zahtevi služe za komunikaciju sa programerima, koji treba da znaju šta se očekuje od elemenata za koje su odgovorni, a kojima su takođe potrebne informacije o elementima koji predstavljaju njihov interfejs. U okviru ovog predmeta ćemo se uglavnom baviti softverskim zahtevima sistema.

ŠTA SADRŽE SOFTVERSKI ZAHTEVI?

Može se reći da sadrže mešavinu informacija o problemima, ponašanju sistema, njegovim svojstvima, dizajnu i ograničenjima u njegovoj izradi.

Softverski zahtevi se definišu u ranim fazama razvoja sistema kao specifikacija onoga što treba implementirati. Oni opisuju kako sistem treba da se ponaša, informacioni domen aplikacije, ograničenja rada sistema, specificiraju svojstva ili atribute sistema. Ponekad oni predstavljaju ograničenja u procesu razvoja sistema. Tako se zahtevi mogu opisati kao:

- Mogućnosti sistema na nivou korisnika (npr. “u word procesor mora da bude uključena i provera pisanja i komande za korekciju”)
- Veoma uopštena svojstva sistema (npr. “sistem treba da omogući da personalne informacije nikada ne mogu postati dostupne bez prethodne autorizacije“)
- Specifična ograničenja sistema (npr. “senzor se mora puniti 10 puta u sec.“)

- Način na koji treba izvršiti neka izračunavanja (npr. “ukupna ocena se izračunava dodavanjem ocena sa ispita, projekata i zalaganja studenta na osnovu sledeće formule ukupna_ocena = ocena_sa_ispita + 2*ocena_sa_projekta + 2/3*ocena_sa_zalaganja “)
- Ograničenja koja se odnose na razvoj sistema (npr. “sistem mora biti razvijen korišćenjem Ade“)

Neki ljudi sugerišu da zahteve treba tretirati kao naredbe onoga što sistem treba da radi, a ne i kako to treba da radi. To je atraktivna ideja, ali je u praksi nije lako sprovesti. Zbog toga se može reći da zahtevi sadrže mešavinu informacija o problemima, ponašanju sistema, njegovim svojstvima, dizajnu i ograničenjima u njegovoj izradi. To može proizvesti poteškoće, jer dizajn i ograničenja koja se odnose na njegovu izradu mogu biti u sukobu sa drugim zahtevima. To je realnost i u proces analize sistema i specifikacije zahteva treba uključiti aktivnosti za pronalaženje i rešavanje problema koji nastaju iz te činjenice.

ŠTA SE REŠAVA SOFTVERSKIM ZAHTEVIMA?

Softverski zahtev je svojstvo koje mora biti izloženo kako bi se rešio neki problem u stvarnom svetu

Najprostije rečeno, softverski zahtev je svojstvo koje mora biti izloženo kako bi se rešio neki problem u stvarnom svetu.

Stoga, softverski zahtev je svojstvo koje mora posedovati softver koji je razvijen ili prilagođen da reši određeni problem. Takav softver može imati za cilj da automatizuje deo zadatka nekoga ko će koristiti softver, da podrži poslovne procese organizacije koja je naručila softver, da ispravi nedostatke postojećeg softvera ili da kontroliše uređaj - da navede samo nekoliko od mnogih problema za koje su softverska rešenja moguća.

Način na koji korisnici, poslovni procesi i uređaji obično funkcionišu su složeni. Zbog toga, zahtevi za određeni softver su obično složena kombinacija zahteva različitih ljudi na različitim nivoima organizacije i iz okruženja u kojem će softver raditi
Suštinska osobina svih softverskih zahteva je da se oni mogu proveriti. Verifikacija određenih softverskih zahteve može biti teška i skupa. Na primer, verifikacija zahteva o frekvenciji poziva u call centru može zahtevati razvoj softvera za simulaciju. I tokom specifikacije softverskih zahteva i tokom testiranja softvera i provere kvaliteta mora se omogućiti da se zahtevi provere i potvrdite u okviru ograničenja koja se odnose na resurse.

Pored karakteristika ponašanja koje izražavaju, zahtevi sadrže i druge osobine. Uobičajeni primeri uključuju definisanje prioriteta zahteva koji omogućavaju kompromise u odnosu na ograničene resurse kako bi se omogućilo praćenje napretka projekta. Obično su softverski zahtevi jedinstveno identifikovani tako da se mogu podvrgnuti kontroli i njima upravljati tokom životnog ciklusa softvera.

ŠTA JE INŽENJERSTVO ZAHTEVA?

Relativno nov termin koji je smišljen sa namerom da pokriju sve aktivnosti obuhvaćene otkrivanjem, dokumentovanjem i održavanjem zahteva za sisteme

Inženjerstvo zahteva je relativno nov termin koji je smišljen sa namerom da pokrije sve aktivnosti obuhvaćene otkrivanjem, dokumentovanjem i održavanjem zahteva za sisteme bazirane na radu računara.Korišćenje termina inženjering podrazumeva sistematsko i višestruko korišćenje tehnika kojima se obezbeđuje da zahtevi sistema budu kompletni, konzistentni, relevantni itd. Inženjering zahteva ima mnogo zajedničkog sa sistem analizom – analizom i specifikacijom poslovnog sistema. Razlika je u tome što u praksi, sistem analizu prvenstveno treba fokusirati na poslovanje a ne na sistem, mada se kao i inženjering zahteva, ona često odnosi i na jedno i na drugo.

U ovom predmetu se naglasak stavlja na proces inženjeringa zahteva za softverske sisteme. Procesi i metodi koji se koriste za razvoj i analizu softverskih zahteva su generalno primenljivi i na sistemske zahteve tj. zahteve koji se primenjuju na sistem kao celinu a ne samo na softverske komponente sistema.

Inženjering zahteva košta zavisno od tipa i veličine sistema koji se razvija i aktivnosti koje su uključene u inženjering zahteva. U nekim slučajevima, sistemski zahtevi se ne razvijaju detaljno; u drugim se moraju proizvesti formalne specifikacije.
Na osnovu daljih istraživanja se može reći da za velike softversko/hardverske sisteme, oko 15% ukupnog budžeta se troši na aktivnosti inženjeringa zahteva. To uključuje troškove detaljne specifikacije sistema. Za manje sisteme koji su pretežno softverski, troškovi zahteva su manji od ovoga i iznose oko 10% ukupnog budžeta.

KO SU AKTERI U RAZVOJU SOFTVERSKOG SISTEMA?

Ljudi ili organizacije na koje sistem ima uticaja i koji s druge strane imaju posredan ili neposredan uticaj na sistemske zahteve.

Akteri u razvoju sistema (system stakeholders) sistema su ljudi ili organizacije na koje sistem ima uticaja i koji s druge strane imaju posredan ili neposredan uticaj na sistemske zahteve. Oni uključuju krajnje korisnike sistema, menadžere i druge koji su uključeni u procese organizacije na koje sistem ima uticaja, inženjere koji su odgovorni za razvoj i održavanje sistema, kupce koji će koristiti sistem kako bi od njega dobili odgovarajuće usluge, eksterna tela kao što su vladine organizacije koje su zadužene za donošenje zakona ili sertifikata itd.

Na primeru razvoja jednog automatizovanog sistema za signalizaciju na železnici, mogući stakeholderi su:

- operatori u železničkim kompanijama koji su odgovorni za obavljanje signalizacije
- društvo železničara

- menadžeri na železnici
- putnici
- inženjeri koji su zaduženi za instalaciju i održavanje uređaja
- vlasti koje su odgovorne za izdavanje sertifikata koji se odnose na sigurnost

Potrebno je identifikovati značajne stakeholdere sistema i otkriti njihove zahteve. Ako se to ne uradi, može se desiti da oni za vreme razvoja sistema ili pošto je sistem isporučen, insistiraju na promenama.

KAKVA JE VEZA IZMEĐU SOFTVERSKIH ZAHTEVA I PROJEKTOVANJA?

Inženjering zahteva se odnosi na ono šta treba uraditi; projektovanje - na ono kako to treba uraditi. Međutim postoji puno međuzavisnosti koje treba imati u vidu pri utvrđivanju zahteva.

Često između zahteva i projektovanja postoji veoma složena veza. Neki autori sugerišu da su to zasebne aktivnosti; zahtevi se uglavnom odnose na probleme koje treba rešiti; projektovanje (design) se odnosi na rešenje tih problema. Drugim rečima, inženjering zahteva se odnosi na ono što treba uraditi; projektovanje se odnosi na ono kako to treba uraditi.

Bilo bi lepo da je to zaista tako i posao ljudi koji rade na specifikaciji zahteva kao i projektovanju bi bio mnogo lakši. Međutim, u stvarnosti, inženjering zahteva i projektovanje su međusobno zavisne aktivnosti. Za to postoji mnogo razloga:

- Sistemi se uvek instaliraju u nekom okruženju, i danas u tom okruženju gotovo uvek postoje i drugi sistemi. Ti drugi sistemi obično predstavljaju ograničenja za projektovanje sistema. Na primer, ograničenje za projektovanje novog sistema može biti da sistem koji se razvija mora da svoje informacije dobija iz postojeće baze podataka. Ona je već projektovana i deo njene specifikacije mora biti uključen u dokument zahteva.
- U okviru velikih sistema, neophodno je identifikovati podsisteme i njihove međusobne veze što se karakteriše kao arhitekturalni dizajn. U tom slučaju, zahtevi se definišu na nivou identifikovanih podsistema a ne čitavog sistema. Identifikacija podsistema znači da se procesi inženjeringa zahteva za svaki podsistem mogu paralelno odvijati.
- Zbog uštede budžeta, vremena ili kvaliteta, organizacija može da pri implementaciji novog sistema ponovo projektuje deo ili čitav postojeći softverski sistem. Ovo ograničava kako sistemske zahteve tako i projektovanje.
- Ako sistem treba da bude prihvaćen od strane nekog eksternog regulatora, može biti neophodno koristiti standardni, „sertifikovani“ način projektovanja koji je već testiran u drugim sistemima.

ŠTA ZNAČI UPRAVLJANJE SOFTVERSKIM ZAHTEVIMA?

Proces upravljanja promenama u zahtevima sistema

Upravljanje zahtevima je proces upravljanja promenama u zahtevima sistema. Zahtevi sistema se uvek menjaju kako bi odslikali promene potreba skateholdera, promene u okruženju u kojem je sistem instaliran, promene u preduzeću koje planira da instalira sistem, promene u zakonskoj regulativi itd. Tim promenama mora da se upravlja, kako bi one imale ekonomski smisao i doprinele poslovnim potrebama organizacije koja kupuje sistem. Mora se oceniti tehnička izvodljivost predloženih promena i omogućiti da se one odvijaju u okviru predviđenog budžeta i vremena.

Osnovne aktivnosti upravljanja zahtevima su kontrola promena i ocena uticaja promena. Kontrola promena se odnosi na utvrđivanje i izvršavanje formalnih procedura za sakupljanje, verifikovanje i ocenjivanje promena; ocena uticaja promena se odnosi na procenu kako će predložene promene uticati na sistem kao celinu. Kada se promene odnose na specifične zahteve, važno je proveriti na koje će druge zahteve verovatno te promene uticati. Upravljanje zahtevima zahteva beleženje informacija o npr. vezama između zahteva, izvoru zahteva i dizajnu sistema. O upravljanju zahtevima će se govoriti detaljnije u narednim predavanjima.

ZAHTEVI PROIZVODA I ZAHTEVI PROJEKTA

Zahtevi proizvoda utiču na svojstva softvera u razvoju. Zahtevi projekta obuhvataju elemente koji učestvuju u projektovanju i u izradi softvera.

Zahtevi proizvoda (product requirements) su svi zahtevi koji utiču na svojstva softverskog sistema koji treba razviti. Kako projekti imaju i druge rezultate, sem dobijanje proizvoda, to se zahtevi projekta razlikuju od zahteva proizvoda. Dokument Specifikacija softverskih zahteva (SRS) sadrži zahteve proizvoda, ali ne i zahteve projekta, te ne sadrži detalje vezane za projektovanje i izradu softvera, plan projekta, planove testiranja, i slične informacije. Zahtevi projekta (project requirements) uključuju:

- Fizičke resurse koji su potrebni razvojnom timu,
- Potrebe obuke članova tima,
- Korisničku dokumentaciju, tj. materijal za obuku korisnika, vežbe, uputstva, i obaveštenja o izdanjima softvera.
- Dokumentaciju za podršku, kao što su help-desk resursi, i informacije o održavanju i servisiranju instalirane opreme.
- Potrebne promene u infrastrukturi okruženja u kome sistem treba da radi.
- Zahteve i procedure za izdavanje proizvoda, za njegovu instalaciju u okruženju rada, konfigurisanje i testiranje postavljene instalacije.
- Zahteve i procedure za prelazak sa starog na novi sistem, kao što su informacije o migraciji podataka i konverziji zahteva, postavljanja sigurnosti, dodatna obuka,
- Sertifikacija proizvoda i njegova usaglašenost sa standardima.

- promene u postupcima rada, u procesima, u organizacionoj strukturi, i sl.
  Izvora softvera i hardvera trećih lica i njihovo licenciranje.
- Zahtevi beta testiranje, proizvodnje, pakovanja, marketinga, i distribucije sistema.
- Sporazumi o servisu na nivou kupaca.

- Zahteve za dobijanje zakonske zaštite (patenti, robne marke, ili autorska prava), za intelektualnu svojinu koja je povezana sa softverom.

Projektni zahtevi nisu obuhvaćeni programom ovog predmeta, jer je fokus na zahtevima proizvoda

REQUIREMENTS ENGINEERING - GEORGIA TECH (VIDEO)

Trajanje: 2:15

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

VIDEO 1 - WHAT IS A REQUIREMENT

Trajanje: 5:53 minuta

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

VIDEO 2 - THREE LEVELS OF SOFTWARE REQUIREMENTS, WIEGERS (VIDEO)

Trajanje: 7:29

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 3

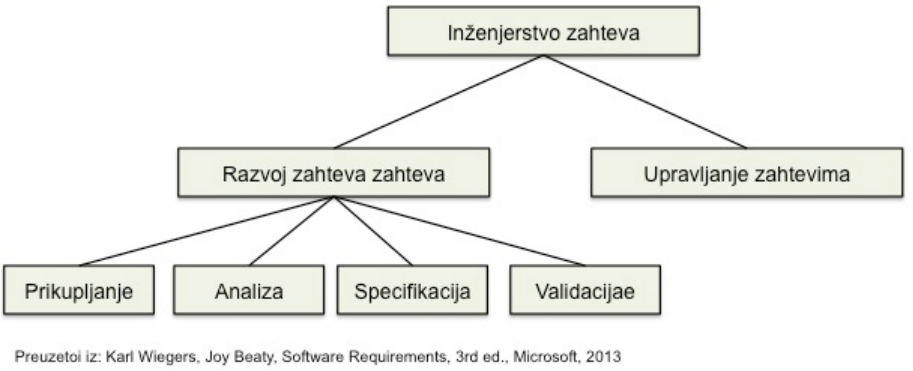
Razvoj zahteva i upravljanje zahtevima

RAZVOJ ZAHTEVA: PRIKUPLJANJE I ANALIZA

Inženjerstvo zahteva se deli na razvoj zahteva i upravljanje zahtevima.

Inženjerstvo zahteva se može podeliti na (slika 1):

- razvoj zahteva, i na
- upravljanje(menadžment) zahtevima



Slika 3.1 Poddiscipline inženjerstva softverskih zahteva

Bez obzirana na tip projekta razvoja softvera, to su aktivnosti koje morate da sprovedete. Zavisno od životnog ciklusa projekta, ove aktivnosti se izvršavaju u različitim vremenima i sa različitom dubinom u analizi zahteva, kao i detalja veznaih za njih. Kao što se vidi na slici 1, razvoj zahteva se realizuje izvršenjem sledećih aktivnosti:

- Pikupljanje zahteva (elicitation)
- Analiza zahteva (analysis)
- Specifikacija zahteva (specification), i
- Potvrđivanje (validation)

Prikupljanje zahteva (requirements elicitation) obuhvata sve aktivnosti povezane sa otkrivanjem zahteva, kao što su intervjui, radionice, analiza dokumenata, izrada prototipa, i dr. Ključne akcije su:

- Utvrđivanje očekivane klasa proizvoda korisnika i drugih aktera,
- Razumeti zadataka korisnika i ciljeva, kao i poslovnih ciljeva sa kojima su ovi zadaci u vezi

- Proučavanje okruženja u kome će novi proizvod raditi.
- Raditi sa pojedincima koji predstavljaju klasu korisnika radi razumevanja njihovih potreba funkcionalnosti i njihova očekivanja kvaliteta.

Analiza zahteva se radi da bi se dobilo bogatije i što preciznije razumevanje svakog zahteva i radi predstavljanje skupa zahteva na različite načine.

- Analiza informacija dobijenih od korisnika radi utvrđivanja razlika ciljeva njihovih zadataka na osnovu funkcionalnih zahteva, očekivanog kvaliteta, poslovnih pravila, predloženih rešenja, i dr.
- Usaglašavanje plana projekta sa razvojem novih zahteva
- Pregovaranje o novim opredeljenjima na osnovu ocenjenog efekta promene zahteva,
- Definisanje veza i zavisnosti između zahteva
- Povezivanje zahteva do odgovarajućih projektnih rešenja, koda i testova
- Praćenje statusa zahteva i aktivnosti menjanja tokom projekta.

RAZVOJ ZAHTEVA: SPECIFIKACIJA I POTVRĐIVANJE

Specifikacija zahteva - predstavljanje i čuvanje znanja o prikupljenim zahtevima. Potvrđivnje zahteva potvrđivanje korektnosti prikupljenih zahteva.

Specifikacija zahteva (requirement specification) obuhvata predstavljanje znanja o prikupljenim zahtevima i njihovo trajno smeštaj na dobro organizovan način. Glavna aktivnost je:

- Prevođenje prikupljenih potreba korisnika u zapisane zahteve i dijagrama pogodne za razumevanje, analizu i upotrebu od strane aktera kojim su namenjeni.

Potvrđivanje zahteva (requirements validation) potvrđuje da imate korektan skup informacija o zahtevima koji će omogućiti inženjerima razvoja softvera da razviju rešenje koje zadovoljava poslovne ciljeve. Glavne aktivnosti su:

- Analiza dokumenta sa zahtevima radi uklanjanja problema i pre nego što razvojni tim ih usvoji.
- Razvoj testova prihvatanja i kriterijuma koji će potvrditi da će proizvod koji se bazira na zahteve zadovoljava potrebe korisnika, i da ostvaruje poslovne ciljeve.

> Iteracija je od ključnog značaja za uspeh procesa razvoja zahteva. Zato, planirajte više iterativnih ciklusa za ispitivanje zahteva, za njihovo poboljšanje i obuhvatanje detaljnijih i preciznijih zahteva, kao i potvrđivanje utvrđenih zahteva od strane korisnika. Ovo obično zahteva vreme i deluje frustrirajuće.
> Nikada nećete dobiti perfektne zahteve. Cilj razvoja zahteva je da se dobiju dovoljno dobri zahtevi koji će omogućiti da se dobije sledeća konstrukcija softvera sa prihvatljivim nivoom rizika. Najveći rizik je da morate da izvršite neplanirane ponovne aktivnosti na izmeni, jer tim nije u potpunosti razumeo zahteve za sledeću fazu razvoja softvera pre početka daljeg projektovanja i konstruisanja softvera.

UPRAVLJANJE ZAHTEVIMA

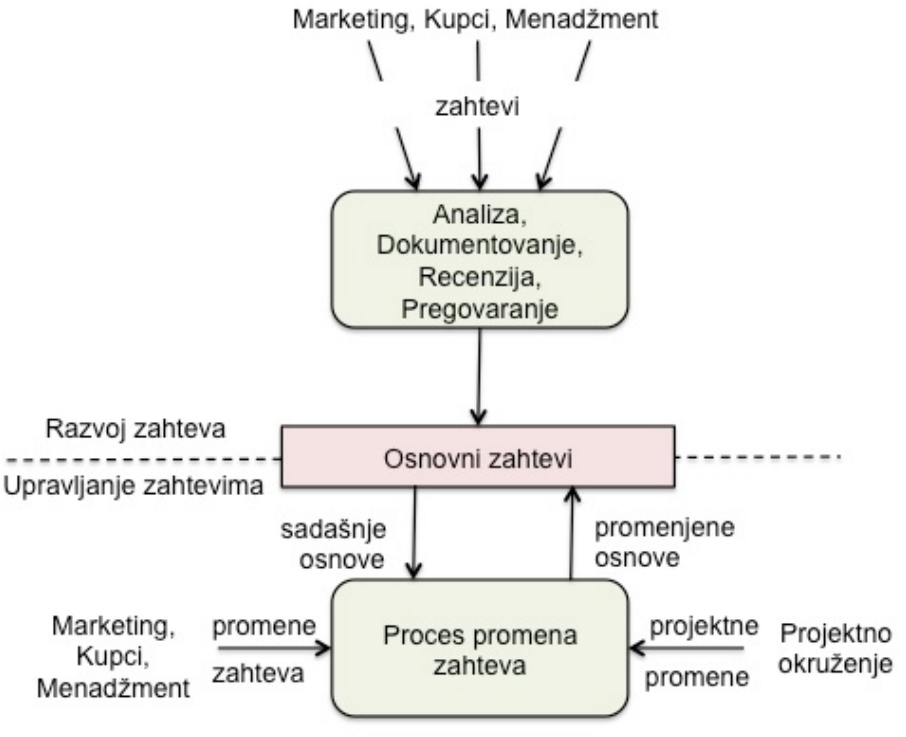
Zahtevi se razvijaju, ali i stalno menjaju, tako da treba upravljati procesom menjanja zahteva

Upravljanje zahtevima uključuje sledeće aktivnosti:

- Definisanje osnovnih vrednosti zahteva, presek u vremenu funkcionalnih i nefunkcionalnih zahteva koji su usaglašeni, recenzirani, i odobreni.
- Ocenjivanje uticaja predloženih promena u zahtevima i ugradnja usvojenih promena u projekat na jedan kontrolisan način.
- Održavanje plana projekta u uslovima promena zahteva.
- Pregovaranje novih opredeljenja na osnovu procenjenog uticaja promena zahteva.
- Definisanje veza i zavisnosti koje postoje između zahteva.
- Praćenje pojedinačih zahteva do njihovih odgovarajućih projektnih rešenja, izvornog programa (koda) i testova.
- Praćenje statusa zahteva i aktivnosti promena tokom realizacije celog projekta.

Svrha upravljanja zahtevima nije da oteža promene zahteva, ili da ih načini teškim za primenu promena. Cilj je da se predvide i primene, vrlo realne promene koje možete da očekujete, kako bi minimizirali i eventualne negativne posledice na projekat.

Slika 2 prikazuje drugi pogled na granicu između razvoja zahteva i upravljanje zahtevima. U ovom predmetu ćemo izučavati više tehnika za realizaciju prikupljanja zahteva, njihovu analizu, specifikaaciju, potvrđivanje i upravljanje zahtevima.



Preuzetoi iz: Karl Wiegers, Joy Beaty, Software Requirements, 3rd ed., Microsoft, 2013

Slika 3.2 Granica između razvoja zahteva i upravljanja zahtevima

ZAHTEVI PROIZVODA

Zahtevi proizvoda se odnose na specifikaciju ograničenja ponašanja projektovanog sistema

Može se napraviti razlika između parametara koji se odnose na softverski proizvod koji se razvija i parametara koji se odnose na proces razvoja softvera.

Zahtevi proizvoda su zahtevi koji se vezuju za sam softverski proizvod koji se razvija (na primer, "Softver mora proveriti da li student ispunjava sve preduslove pre nego što se registruje za kurs"). Oni specificiraju željene karakteristike softverskog sistema ili podsistema. Proizvodni zahtevi koji se odnose na performanse i kapacitet se često mogu formulisati veoma precizno, što nije slučaj sa proizvodnim zahtevima koji se odnose na korišćenje. Zbog toga se oni često iskazuju na neformalan način.

1. Mnogi zahtevi koji se odnose na s/w proizvode se odnose na specifikaciju ograničenja ponašanja projektovanog sistema. Oni ograničavaju slobodu dizajnera sistema. Zahtevi s/w proizvoda se mogu definisati na sledeći način:
2. Servis X sistema treba da ima raspoloživost 999/1000 ili 99%. To je zahtev koji se odnosi na pouzdanost koji znači da na svakih 1000 zahteva za datim servisim, 999 mora biti zadovoljeno.
3. Sistem Z mora da obradi minimum 8 transakcija u sekundi. To je zahtev koji se odnosi na performanse.
Izvršni kod sistema Z mora da bude ograničen na 512 Kbyte. To je zahtev koji se odnosi na prostor i kojim se specificira maksimalna veličina memorije sistema.

4. IN korisnika se mora uneti 5 sec nakon što se kartica unese u čitač kartica. To je zahtev za performansom.

Pored ovih zahteva kojima se ograničava ponašanje sistema, mogu postajati i **proizvodni zahtevi** koji se odnose na izvršni kod sistema. Na primer:

1. Sistem se mora razvijati za PC i Macintosh platforme; to je zahtev koji se odnosi na prenosivost (portabilnost) i koji ima uticaja na način na koji sistem treba da bude dizajniran.
2. Sistem mora da celokupnu eksternu komunikaciju enkriptuje korišćenjem RSA algoritma; to je zahtev koji se odnosi na sigurnost kojim se specificira da se u proizvodu mora koristiti specifičan algoritam.
3. Sistem X mora da se implementira korišćenjem izdanja 5 biblioteka Z; ovaj zahtev se može smatrati proizvodnim zahtevom ako on označava neko moguće projektantsko rešenje, medjutim, on takodje može predstavljati i neki eksterni zahtev ako je nametnut kao eksterna odluka organizacije

Često se dešava da su proizvodni zahtevi medjusobno konfliktni. Pored specifičnih zahteva, postoje zahtevi organizacije na veoma visokom nivou apstrkcije i drugi globalni zahtevi prema kojima se moraju analizirati svi drugi zahtevi. Rešavanje ovih konflikata nije u programu ovog predmeta..

PROCESNI ZAHTEVI

Procesni zahtevi se odnose na proces razvoja softverskog sistema i uključuju zahteve za korišćenjem standarda, CASE alata i dobijanje različitih izveštaja za menadžment.

Procesni zahtevi se odnose na proces razvoja sistema i postoje kada korisnici sistema žele da utiču na njega. Procesni zahtevi uključuju zahteve za korišćenjem standarda i metoda koje se moraju primeniti, CASE alati koje treba koristiti i dobijanje različitih izveštaja za menadžere. Primeri procesnih zahteva mogu biti:

1. Razvojni proces mora biti eksplicitno definisan i mora se uklapati u ISO 9000 standard.
2. Sistem se mora razvijati korišćenjem XYZ paketa CASE alata
3. Moraju se proizvoditi izveštaji za menadžment koji pokazuju koliko je napora uloženo za razvoj svake identifikovane komponente sistema
4. Mora se specificirati plan za otklanjanje katastrofalnih situacija pri razvoju sistema

Procesni zahtevi se obično uključuju kada sistemi proizvode velike organizacije, sa postojećim standardima i paksom i kvalifikovanim osobljem. Oni mogu biti različiti, od veoma specifičnih instrukcija koje se moraju poštovati u procesu razvoja sistema do veoma generalnih zahteva kao što je onaj gornji kojim se specificira da se proces mora voditi po ISO 9000 standardu.
Procesnim zahtevima se mogu postaviti realna ograničenja na projektovanje sistema. Na primer, jedna organizacija može specificirati da se pri razvoju sistema mora koristiti specificirani skup CASE alata jer ima iskustva u korišćenju tih alata. Ako ti alati recimo ne podržavaju objektno-orijentisani razvoj, to znači da arhitektura sistema ne može biti objektno-orijentisana.

Neki softverski zahtevi generišu implicitne procesne zahteve. Jedan primer je Izbor tehnike verifikacije. Drugi primer bi mogao biti korišćenje posebno rigoroznih tehnika analize (poput formalnih metoda specifikacije) kako bi se smanjile greške koje mogu dovesti do neadekvatne pouzdanosti. Procesni takođe mogu biti nametnuti direktno od strane organizacije za razvoj, kupca proizvoda ili treće strane, kao što je sigurnosni regulator.

EKSTERNI ZAHTEVI

Mogu se svrstati i u zahteve proizvoda i u procesne zahteve; proizilaze iz okruženja u kojem se sistem razvija.

Eksterni zahtevi su zahtevi koji se mogu svrstati i u zahteve proizvoda i u procesne zahteve. Oni proizilaze iz okruženja u kojem se sistem razvija pa se moraju bazirati na informacijama o aplikativnom domenu, mišljenju organizacije, potrebi da sistem radi sa drugim sistemima, regulativi koja postoji u oblasti zdravstva, zaštite ili prirodnim zakonima kao što su zakoni fizike.

Neki primeri eksternih zakona su:

1. Sistem za edukaciju studenata: Format podataka o studentskom zapisu se mora podudarati sa onim koji se koristi u nacionalnom sistemu za evidenciju studenata.
2. Sistem medicinskih podataka: Službenik za zaštitu podataka u organizaciji mora da pre nego što se sistem stavi u rad, izda sertifikat da se svi podaci održavaju prema legislativi za zaštitu podataka.

Prvi od gornjih zahteva je zahtev koji proizilazi iz eksternog okruženja. Drugi od zahteva proizilazi iz potrebe da se sistem usaglasi sa legislativom za zaštitu podataka.
Eksterni zahtevi retko imaju formu “sistem treba” ili “sistem ne treba”. Umesto toga, oni opisuju okruženje sistema koje se mora uzeti u obzir. Iz tih razloga je često veoma teško eksterne zahteve uključiti u struktuirani model sistema.

REQUIREMENTS ENGINEERING PROCESS - GEORGIA TECH (VIDEO)

Trajanje: 1:18 minuta

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 4

Problemi vezani za zahteve sistema

VIDEO PREDAVANJE ZA OBJEKAT "PROBLEMI VEZANI ZA ZAHTEVE SISTEMA"

Trajanje video snimka: 18min 57sek

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

ŠTA JE PROBLEM?

Može se definisati kao razlika između percepcije i realnosti (želja i stanovišta). Problem se rešava u pet koraka.

Problem se može definisati kao razlika između stanovišta kako stvari treba izvesti i načina na koji su one izvedene.

Prema definiciji, ako korisnik vidi nešto kao problem, onda je to pravi problem, i njega treba rešavati. Ipak, dužnost onih koji rešavaju problem je da se istraže sva alternativna rešenja pre nego što se pređe na novo rešenje sistema. Prilikom rešavanja kompleksnog problema, na početku moramo imati cilj. *Cilj analize problema je da se, pre nego što počne razvoj, stekne bolji uvid u problem koji treba da bude rešen.* Specifični koraci koji se moraju preduzeti u cilju postizanja cilja su:

1. *Postizanje dogovora o definiciji problema,*
2. *Razumevanje potreba korisnika,*
3. *Utvrđivanje aktera i korisnika,*
4. *Definisanje granica sistemskog rešenja,*
5. *Utvrđivanje ograničenja kojima je izloženo rešenje.*

Problem treba opisati u standardnom formatu. Popunjavanje same tabele kojom se opisuje problem za aplikaciju je jednostavno, mada se treba pridržavati određenih tehnika kako bi se osiguralo da svi učesnici na projektu rade ka istom cilju.

Potrebno vreme da se dobije ugovor o problemu koji se rešava može izgledati kao mali i beznačajan korak. Međutim, ponekad, to nije slučaj, što pokazuje sledeći slučaj.

Jedan od klijenata je bio angažovan na nadogradnji IS / IT sistema, koji je trebalo da obezbedi fakturisanje i finansijsko izveštavanje između kompanije i njenih dobavljača. Cilj ovog projekta je bio da se "poboljša komunikacija sa dobavljačima". Tim je krenuo u razvoj

novog sistem kojim se predviđalo bolje finansijsko izveštavanje, poboljšanje formata faktura , onlajn naručivanje delova i korišćenje elektronske pošte. Vizija rukovodstva je bila bitno drugačija: Primarni cilj novog sistema je bio da se obezbedi elektronski transfer sredstava koji će poboljšati protok gotovine. Nakon žestoke rasprave, postalo je jasno da je problem prvog reda kojim se treba baviti u okviru novog sistema elektronski transfer sredstava, dok je korišćenje e-mail i drugih načina komunikacije sa dobavljačima osobine za koje se smatra da je samo "lepo imati".

LOŠI ZAHTEVI STVARAJU PROBLEMA U RAZVOJU SOFTVERA

Najčešći problemi su: softver se isporučuju sa zakašnjenjem i prevazilazi se planirani budžet

Razvoj softverskih sistema je vezan sa različitim problemima: oni se često isporučuju sa zakašnjenjem i prevazilaze planirani budžet. Ne rade ono što korisnici zaista žele i često se nikada ne iskoriste neke mogućnosti sistema za koje su korisnici platili.

Najčešći razlozi za takve probleme su poteškoće vezane za zahteve sistema.

- Kao što samo ime kaže, zahtevi sistema definišu šta se od sistema traži da radi i okolnosti pod kojima se traži da to radi. Drugim rečima, zahtevima se definišu servisi koje sistem treba da pruži i ograničenja koja se odnose na rad sistema.
- Obzirom da postoje različiti tipovi zahteva, nemoguće je definisati standardni način za opisivanje zahteva ili definisanje najboljeg načina da se oni specificiraju. To zavisi od toga ko zahteve specificira, ko će zahteve najverovatnije pročitati, prakse organizacije koja postavlja zahteve i aplikativnog domena sistema.
- Opis zahteva sistema može da uključi i druge informacije koje se moraju uzeti u obzir pri implementaciji sistema ali koje nisu iskazane u obliku servisa ili ograničenja sistema. Te informacije se mogu odnositi na opšte informacije o tipu sistema koji se specificira (informacioni domen), informacije o standardima koji se moraju pratiti prilikom razvoja sistema, informacije o drugim sistemima koji su u interakciji sa datim sistemom itd.

POSLEDICE PRIMENE LOŠIH ZAHTEVA

Greške u zahtevu treba otklanjati što pre, jer se troškovi otklanjanja posledica uvećavaju s vremenom.

Glavna posledica pogrešno definisanih zahteva je ponovni rad na utvrđivanju boljih i realnijih zahteva, a to povećava troškove projekta. Praksa je pokazala da to povećava ukupne troškove razvoja i za 30-50%. Greške u postavljanju zahteva učestvuju u troškovima ponovnog rada sa 70 do 85%. Neke promene povećavaju vrednost proizvoda. Međutim, najčešće taj dodatni rad samo otklanja probleme nastale zbog loših zahteva, a da pri tome ne uvećava vrednost proizvoda. To je vrlo zamorno i frustrirajuće za sve aktere u razvoju zahteva. Prema tome,

kreiranje boljih zahteva nije samo trošak, to je investicija koja se na kraju isplati.
Troškovi otklanjanja loših zahteva i definisanja boljih su najmanji ako se te promene izvrše što pre, tj. odmah po njihovom definisanju. Ako se te izmene vrše kasnije, troškovi znatno rastu, jer se ponovno moraju obaviti sve aktivnosti koje zavise od loše definisanih zahteva. Na primer, ako se greške otkriju pri upotrebi softvera, troškovi otklanjanja grešaka u zahtevima mogu biti i 100 i više puta veće nego ako se otklone na početku razvoja softvera, tj. odmah posle definisanja zahteva. Zato, vrlo je važno preventivno delovati d abi se izbegle greške u zahtevima, ili ih pronaći vrlo rano u procesu razvoja softvera.
Nedostaci u zahtevima uvode mnogo rizika u uspeh projekta razvoja softvera. Pod uspehom se ovde smatra razvoj softvera koji u potpunosti zadovoljava stvarne funkcionalne i kvalitativne zahteve korisnika, u skladu sa dogovornom cenom i ugovorenim rokom isporuke softvera

Greške u zahtevima nastaju najčešće iz sledećih razloga:

1. **Nedovoljno uključenje korisnika:** Korisnici često nisu svesni značaja svog uključenja u razvoj zahteva. Inženjeri razvoja često misle da znaju i razumeju zahteve korisnika. Analitičar poslovanja nije razumeo stvarne potrebe korisnika softvera.
2. **Pogrešno planiranje:** Procena potrebnog rada u projektu i njegovog trajanja zahteva da budete svesni veličine posla vezana za primenu nekog zahteva i na mogućnosti razvojnog tima.
3. **Puzeći razvoj korisničkih zahteva:** Često su projekti optimistički planirani. To dovodi do njihovog produžetka i većih troškova razvoja. I zahtevi se stalno menjaju i uvećavaju. Zato, pri planiranju, planirajte vreme i novac za nove zahteve i njihove promene, kako to ne bi dovelo do pomeranja rokova i budžeta projekta.
4. **Dvosmisleni zahtevi:** Dvosmisleni zahtevi dovode do neočekivanog razvoja softvera. Zbog različitog razumevanja istog zahteva. Opasnost od ovoga se umanjuje ako razvojni tim ima članove koji na zahteve gledaju iz različitih perspektiva i sprovede se kolaborativni rad, kao i testiranje zahteva.
5. **Dodavanje neplaniranih zahteva:** Inženjer razvoja samoinicijativno dodaje zahtev jer misli da je on neophodan, a nije.
6. **Zanemareni akteri:** Neke kategorije korisnika nisu uključene u razvoj zahteva. Uključite i ljude iz održavanja, kao i integratore SW.

IDENTIFIKOVANJE KORISNIKA I AKTERA RAZVOJA

Akter je bilo ko koje materijalno pogođen implementacijom novog sistema; neki akteri su direktni a neki indirektni korisnici sistema.

Efikasnim rešavanjem bilo kog kompleksnog problema je obično obuhvaćeno zadovoljavanje potreba različitih grupa stejkholdera. Akteri (stakholders) obično imaju različite poglede na problem i različite potrebe koje moraju biti rešene. Akter razvoja se može definisati kao bilo ko ko bi mogao biti materijalno pogođen implementacijom novog sistema ili aplikacije. Mnogi akteri razvoja su i korisnici sistema i njihove potrebe se mogu lako fokusirati, jer su oni direktno obuhvaćeni definicijom i korišćenjem sistema . Međutim, neki akteri su samo indirektni korisnici sistema ili na njih utiču samo poslovni rezultati koje sistem generiše.

Ovi akteri razvoja (stejkholderi) se mogu naći bilo gde u biznisu ili u "okruženju". U drugim slučajevima, oni su još dalje uklonjeni iz okruženja aplikacije

Na primer, oni uključuju ljude i organizacije koje se bave razvojem sistema, kupčeve kupce, spoljne agencije koje su u interakciji sa sistemom ili razvojnim procesom. Svaka od ovih klasa aktera može uticati na zahteve sistema ili će na neki način biti uključena u ishode sistema. Pronalaženje aktera sistema i njihovih posebnih potreba je važan faktor u razvoju efikasnog rešenja. U zavisnosti od poznavanja domena problema razvojnog tima, identifikovanje stejkholdera može biti trivijalan ili ne trivijalan korak u analizi problema .

Često, to podrazumeva razgovor sa donosiocima odluka, potencijalnim korisnicima i drugim zainteresovanim stranama. Pitanja koja mogu biti od pomoći u tom procesu su:

- Ko su korisnici sistema ?
- Ko je kupac sistema ?
- Koga pogađaju izlazi koji sistem proizvodi? Ko procenjuje sistem
- kada se on isporuči i razmesti ?
- Da li postoje neki drugi interni ili eksterni korisnici sistema čije se potrebe moraju rešavati ?
- Ko će održati novi sistem ?
- Da li postoji neko drugi ?

DEFINISANJE GRANICA SISTEMA I OGRANIČENJA

Granica sistema definiše ivicu između rešenja i stvarnog sveta koji okružuje naše rešenja. Ograničenje se može definisati kao zabrana izvesnog stepena sl.

Sledeći važan korak je utvrđivanje granica sistemskog rešenja. *Granice sistema definišu ivicu između rešenja i stvarnog sveta koji okružuje softverska rešenja.* Drugim rečima, granica sistema opisuje omotač u kojoj je sadržano rešenje sistema. Informacije, u obliku ulaza i izlaza, idu napred i nazad od sistema do korisnika koji se nalaze izvan njega. Sva interakcija sa sistemom se odvija preko interfejsa između sistema i spoljašnjeg sveta .
Pre bilo kakvih početnih ulaganja u softverski sistem moramo uzeti u obzir ograničenja kojima će rešenje biti izloženo. Ograničenje se može definisati kao zabrana izvesnog stepena slobode prilikom isporuke sistema. Svako ograničenje ima potencijal da ozbiljno ograniči našu mogućnost da isporučimo rešenje kako smo zamislili. Dakle, svako ograničenje se mora pažljivo razmotriti kao deo procesa planiranja, a mnoga mogu prouzrokovati potrebu da se preispita tehnološki pristup koji je na početku predviđen.
Izvori ograničenja mogu biti raznovrsni: raspored, povratak investicija, budžet za rad i opremu, zaštita životne sredine, operativni sistemi, baze podataka, host i sistema klijenata, tehnička pitanja, politička pitanja unutar organizacije, kupovina softvera, politika i procedure kompanije, izbor alata i jezika, osoblje ili drugi izvori.

KORISTI OD KVALITETNO DEFINISANIH ZAHTEVA

Vrlo je važno da se prikupljanje zahteva odvija na način koji će razvojnom timu dati pravu sliku potreba korisnika i tržišta.

Neki smatraju da se gubi vreme na prikupljanju zahteva. Međutim, vreme potrošeno na razvoju pravih zahteva se višestruko isplati, jer se izbegavaju ispravke softvera zbog loše definisanih zahteva.

Proces razvoja zahteva na kolaborativan način treba da uključi sve aktere razvoja softvera. Vrlo je važno da se prikupljanje zahteva odvija na način koji će razvojnom timu dati pravu sliku potreba korisnika i tržišta. To je ključni faktor uspeha. Uključivanje korisnika u razvoj eliminiše slučajeve da se definišu zahtevi koji nisu potrebni korisniku. Korisnike treba uključiti u početnim fazama razvoja softvera.

Razvoj softvera zahteva jedan sistemski pristup. Neophodni je primeniti jedan efektivan proces upravljanja promenama softvera. Dokumentovanje jasno definisanih zahteva olakšava i testiranje sistema. Sve ovo doprinesi razvoju kvalitetnih zahteva, pa i kvalitetnog softvera, koji zadovoljava sve aktere razvoja. Da bi se ovo postiglo, potrebno je da upotrebite nove procedure razvoja i obrasce dokumenata, da obezbedite buku članova tima, i da nabavite potrebne alate za razvoj zahteva. Vaša najveća investicija je vreme koje vaš razvojni tim posveti razvoju zahteva. Time dobijate sledeće koristi:

- manje grešaka u zahtevima i u isporučenom softveru
- smanjenje ponovnog rada na otklanjanju grešeka

- brži razvoj i brža isporuka softvera
- manje nepotrebnih svojstava softvera i svojstva koja se ne koriste
- manji troškovi poboljšanja softvera
- manje grešaka u komunikacijama među akterima razvoja
- smanjivanje puzećeg proširenja svojstava sistema
- smanjenje haosa u projektu
- veće zadovoljstvo korisnika i članova razvojnog tima
- dobijeni proizvod radi ono što se od njega i očekuje

PREPORUKE ZA DOBIJANJE BOLJIH ZAHTEVA

Na osnovu analize dosadašnje prakse, organizovanjem timskog pristupa, utvrdite poboljšanja koje možete uvesti u razvoj zahteva u vašoj organizaciji.

Ovde se daju preporuka za one koji već rade u razvoju softvera i imaju određena iskustva u razvoju. One mogu da pomognu studentima da kada budu na stručnoj praksi, naprave analizu problema koje firma ima sa zahtevima (primenom razgovora sa iskusnim inženjerima razvoja) i da predlože poboljšanja u radu na razvoju zahteva.

1. Zapišite koje probleme sa zahtevima imate u sadašnjem projektu, ili u ranijim projektima. Utvrdite da svaki od njih predstavlja poseban problem za razvoj zahteva ili za upravljanje razvojem zahteva. Opišite koren javljanja problema i njegov uticaj na projekat.
2. Podstakite diskusiju članova razvojnog tima i drugih aktera razvoja u vezi sa problemima sa zahtevima iz sadašnjeg i ranijih projektima, o posledicama koje su nastale zbog ovih problema, i na izvore njihovih nastanaka. Predložite izmene koje mogu da dovedu do smanjivanja grešaka u razvoju zahteva.
3. Preslikajte terminologiju o zahtevima i o rezultatima koja se koristi u vašoj organizaciji u terminologiju i postupke primenjene u ovoj lekciji, radi utvrđivanja da i organizacija sprovodi sve što je ovde preporučeno.
4. Izvršite jedno jednostavno ocenjivanje (na par stranica) jednog od dokumenata sa zahtevima koji koristite da bi utvrdili da li vaš tim može da predloži neka poboljšanja u ovom dokumentu i u njegovoj pripremi. Najbolje je da koristite nekog spolja radi dobijanja objektivne ocene.
5. Organizujete čas obuke iz oblasti softverskih zahteva za ceo vaš projektni tim. Pozovite ključne kupce softvera, ljude iz marketinga, menadžere, inženjere razvoja, testere i druge učesnike u razvoju zahteva. Obuka obezbeđuje učesnicima korišćenje istog rečnika i termina. Obuka obezbeđuje i zajedničko ocenjivanje i prihvatanje efektivnih tehnika i ponašanja tako da ceo tim sarađuje na efektivniji način u rešavanju zajedničkih izazova.

TEHNIKE ZA ANALIZU PROBLEMA

Poslovno modeliranje, specifične tehnike za analizu problema itd.

Treba takođe napomenuti da se prilikom analize problema mogu koristiti različite tehnike: modeliranje poslovanja, specifične tehnike za analizu problema koje se mogu uspešno primeniti u složenim informacionim sistemima koji podržavaju ključnu poslovnu infrastrukturu. Poslovno modeliranje se može koristiti i za razumevanje načina na koji se poslovanje odvija i identifikovanje mesta na kojima je najproduktivnije smestiti neku aplikaciju. Poslovni slučajevi korišćenja se takođe mogu koristiti za definisanje zahteva za samu aplikaciju

Za softverske aplikacije u ugrađenim sistemima, inženjerstvo softvera se koristi kao tehnika za analizu problema koja pomaže u dekompoziciji složenih sistema na podsisteme. Taj proces pomaže da se razume gde treba softver smestiti i šta je njegova svrha.

ELICITATION PROBLEMS - GEORGIA TECH (VIDEO)

Trajanje: 2:16 minuta

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 5

Značaj inženjerstva zahteva

IZVEŠTAJ NIST-A O GREŠKAMA U SOFTVERU

70% grešaka se nalaze u specifikaciji softvera, a samo 5% je otklonjeno u fazi izrade specifikacije, te su troškovi zbog toga 22 puta veći.

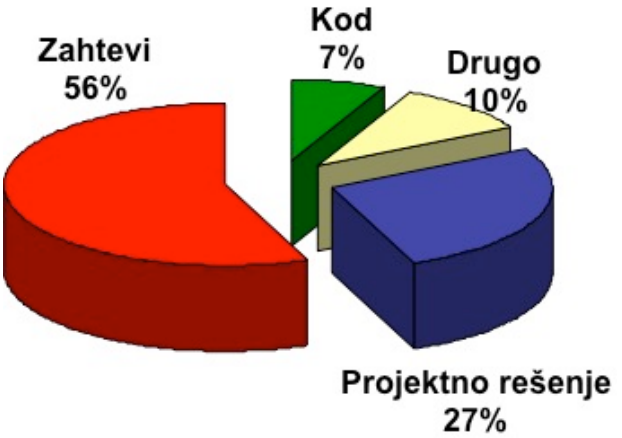
Američki nacionalni institut za standardizaciju i tehnologiju - NIST (National Institute of Standards and Technology) izdao je izveštaj o uspešnosti projekata razvoja softvera na bazi velikog broja projektata. Ustanovio je da:

- 70% grešaka je uneto u sistem u fazi specifikacije softvera
- 30% grešaka je uneto u ostalim fazama razvoja softvera.
- samo 5% grešaka u specifikaciji je korigovano u fazi pripreme specifikacije.
- 95% grešaka je otkriveno kasnije u projektu ili posle isporuke softvera kada su troškovi ispravke softvera zbog grešaka veće u proseku za oko 22 puta u odnosu na troškove ispravke u fazi specifikacije.

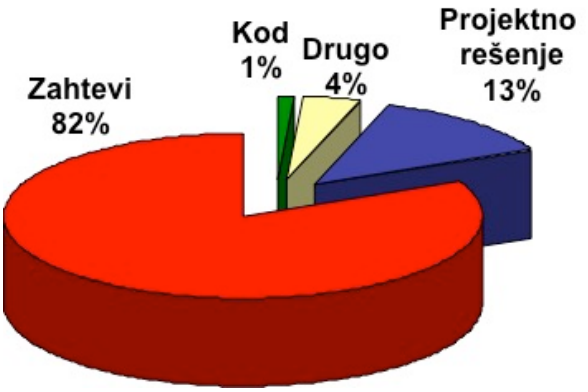
Izveštaj NIST-a zaključuje da je intezivno testiranje od ključne važnosti, ali testiranje utvrđuje greške, koje su dominantne u specifikaciji) u kasnim fazama procesa razvoja.

Na slici 1 prikazani su izvori grešaka u softverskim sistemima koji su analizirani. Očigledno, najdominantniji izvor grešaka su zahtevi (57%), koji nisu tačno definisani.

Na slici 2 prikazan je rad utrošen na otklanjanju grešaka. Vidi se da je najviše rada potrošeno na otklanjanju grešaka u zahtevima, jer su greške načinjene na početku, a utvrđene pri kraju procesa razvoja ili pri korišćenju softvera, a to, kao što smo naveli, dovodi do velikih troškova zbog ponavljanja rada u svim fazama razvoja procesa, praktično, od početka.



Slika 5.1 Izvori grešaka u softveru



Slika 5.2 Troškovi uklanjanja grešaka u odnosu na izvore grešaka (izraženi u procentima)

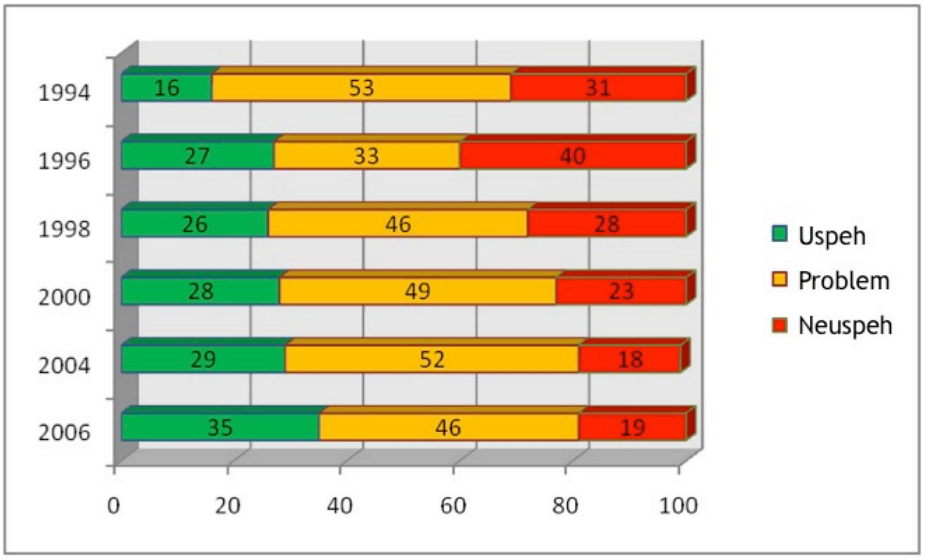
USPEŠNOST PROJEKATA RAZVOJA SOFTVERA

Samo 29% projekata je uspešno završen, a u 53% je bilo problema (prekoračenje rokova i troškova, problemi sa kvalitetom).

U jednom drugom izveštao (CHAOS, 2004) utvrđeno je da:

- 53% projekata ne ispunjava osnovne projekte zahteve, te prekoračuju rokove, premašuju ugovoreni budžet za razvoj ili ne ostvaruju postavljene zahteve.
- 29% je uspešno završeno
- 18% je obustavljeno ili su završeni bez prihvatljivih rezultata.

Ovo ukazuje na velike izazove inženjerstvu softvera, kao i njegovo zaostajanje za drugim inženjerskim disciplinama.. Zamislite šta bi se desilo kada bi 29% projekata razvoja aviona bilo uspešno, 28% obustavljeno, a 53% proizvelo avione sa različitim defektima? Očigledno, softversko inženjerstvo kao mlada inženjerska disciplina, mora da se dalje razvija, a pre svega u doslednoj primeni pravila i metoda koje su postavljeni, a koje se često ne poštuju i zanemaruju. Ipak, situacija se polako poboljšava, kao što pokazuje slika 3 koja prikazuje napredak u uspešnosti projekata razvoja softvera u periodu od 1994. do 2006. godine.



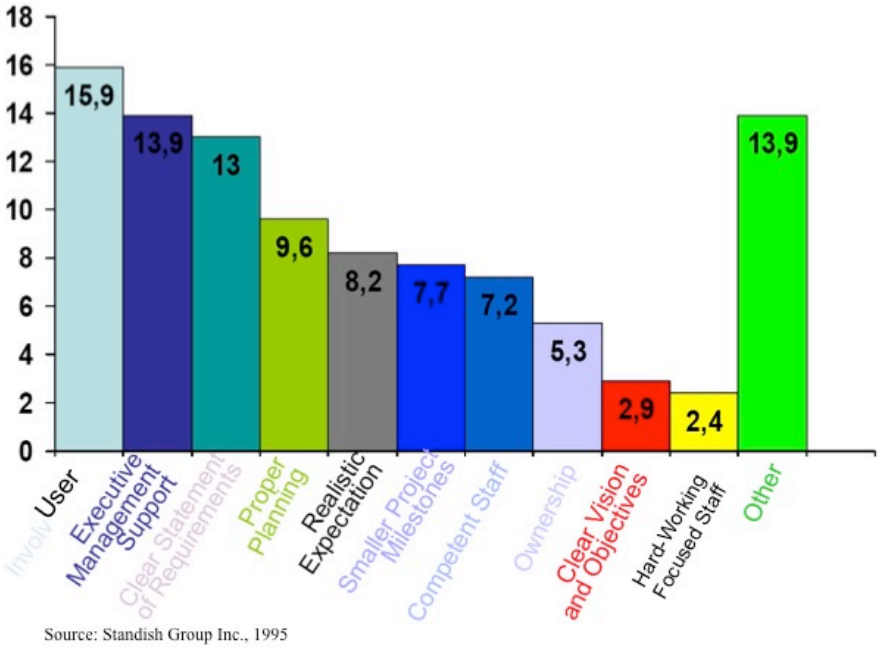
Source: Standish Group Inc., 1994-2006

Slika 5.3 Napredak u razvoju kvaliteta softvera i projekat razvoja softvera

FAKTORI USPEHA I IZVORI PROBLEMA PRI RAZVOJU SOFTVERA

Najznačajniji faktor uspeha je učešće korisnika u razvoju softvera, a najveći izvor grešaka se nalazi u nekompletnim zahtevima.

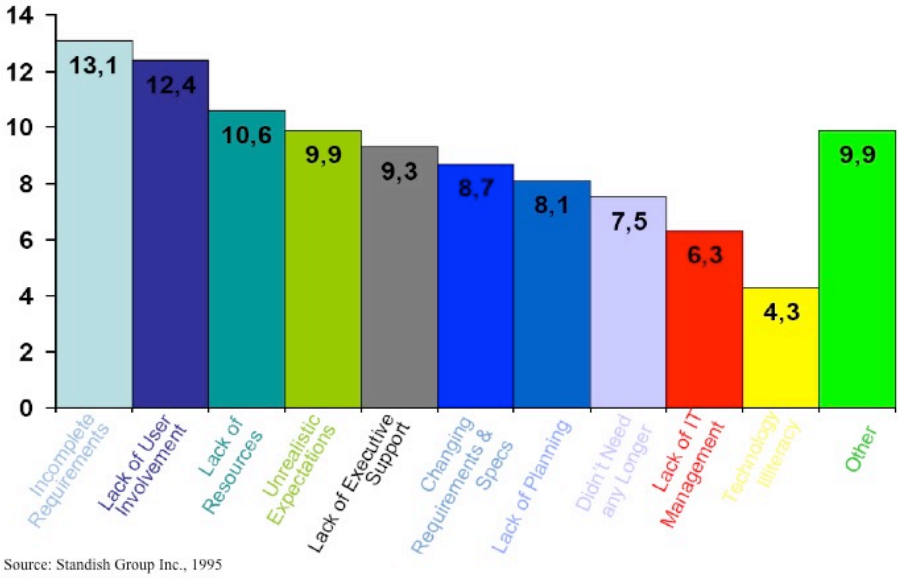
Na slici 4 prikazani su rezultati iz izvešaja koji pokazuju da je na uspešnost projekta razvoja softvera najznačajnije učešće korisnika softvera, a onda izvršni menadžment organizacije.



Source: Standish Group Inc., 1995

Slika 5.4 Faktori uspeha u razvoju softvera

Na slici 5 su prikazani najčešći izvori problema u kvalitetu softvera, tj. problema koji se javljaju u toku razvoja softvera. Kao što se video, prvi uzrok su nekompletni softverski zahtevi, a drugi po učešću u izvori grešaka je korisnik i njegovo (ne)učešće u razvoju softvera.



Slika 5.5 Izvori problema u projektima razvoja softvera

IMPORTANCE OF SOFTWARE ENGINEERING - GEORGIA TECH (VIDEO)

Trajanje: 9:94 minuta

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 6

Vežba

POSLOVNI SLUČAJ PRIVATNE KLINIKE HIPOKRAT

Opis problema sa kojima se susreće privatna klinika koja ne poseduje adekvatan sistem koji će podržati njen rad.

Privatna klinika Hipokrat uskoro obeležava desetogodišnjicu svog postojanja. Poslovanje je započela osnivanjem samo jednog odeljenja – interne medicine, zahvaljujući čemu je i danas prepoznata kao lider među privatnim klinikama u proučavanju, lečenju i sprečavanju bolesti unutrašnjih organa odraslih. U godinama nakon osnivanja, otvorena su odeljenja gastroenterologije, endokrinologije, imunologije, ultrazvučne dijagnostike i magnetne rezonance.

Od svega 12 zaposlenih na dan otvaranja, klinika Hipokrat je prerasla u ozbiljno preduzeće, koje sada broji 70 radnika na različitim funkcionalnim nivoima. Porast broja zaposlenih i otvaranje novih odeljenja, doveli su do povećanja prihoda. Međutim, osim pozitivnih uticaja, razvoj klinike je doveo do toga da je postalo teže pratiti rashode i organizovati rad ljudi. Lekari su zahtevni po pitanju termina za rad, medicinsko i administrativno osoblje ne sme da bude u deficitu, pa je potrebno pažljivo planirati rasporede rada, godišnjeg odmora, iznenadnih bolovanja i slično. Administrativni radnici trenutno vode ovaj deo poslovanja u vidu elektronske dokumentacije. Sistem odlučivanja se najčešće sprovodi kroz prepiske u mejlovima, a donosilac odluka je upravni bord menadžera i direktora. Praćenje stanja i naručivanje materijala za rad se sprovodi na sličan način ili telefonskim razgovorima sa dobavljačima.
Drugi problem sa kojim se susreće klinika je neadekvatno vođenje evidencije o pacijentima, istoriji njihovih poseta, primljenim terapijama i dijagnozama. Klinika već koristi jednu poslovnu aplikaciju namenjenu ustanovama tog tipa, ali je ta aplikacija zastarela i nema mogućnosti za prikupljanje svih podataka koji su klinici neophodni da imaju o svojim pacijentima. Kartoni pacijenata sa medicinskom dokumentacijom se i dalje vode u vidu kartonskih fascikli. Medicinsko osoblje zakazuje preglede putem aplikacije, u kojoj se vodi i veći deo podataka o pacijentima, ali dodatni podaci se zapisuju za strane. S druge strane, aplikacija ne ograničava lekare da obavezno unose izveštaj o obavljenom pregledu, na osnovu koga se kasnije vrši naplata, što ostavlja dosta prostora da se finansijski ošteti preduzeće i obavljaju pregledi mimo regularnih tokova.

POSLOVNI SLUČAJ PRIVATNE KLINIKE HIPOKRAT - NASTAVAK

Identifikovanje problema sa kojima se susreće privatna klinika

Upravni odbor je zabrinut da uspeh kompanije neće moći dugo da se održi, bez obzira na stečen dobar ugled, ukoliko firma ne reguliše način rada zaposlenih i sa pacijentima. Menadžment klinike je takođe frustriran zbog toga što postojeća aplikacija nema mogućnosti generisanja izveštaja koji će im dati podrobnije informacije o stanju kompanije. Oni žele da u svakom trenutku mogu da prate i stanje finansija, jer sumnjaju da evidencija kakva sada postoji nije realan pokazatelj finansijskih prilika. Iako je za sada sve u optimalnim granicama, direktor klinike planira još veći napredak i potreban je stabilan i pouzdan poslovni informacioni sistem koji će to da podrži.

Problemi sa aspekta kompanije:
1. Nedovoljna kontrola i organizacija rada
2. Nepostojanje kontrole rashoda
3. Neadekvatno praćenje finansijskog stanja

Problemi sa aspekta zaposlenih:
1. Promenljiv tok poslovnih procesa
2. Nedovoljna kontrola podataka
3. Ograničene funkcionalnosti postojeće aplikacije
4. Nepostojanje adekvatne automatizacije posla

Problemi sa aspekta pacijenata:
1. Povremeno čekanje zbog pogrešnih rasporeda rada
2. Povremeni gubitak medicinske dokumentacije
3. Ponavljanje istih informacija kod različitih lekara

ANALIZA POTREBA ZA NOVIM POSLOVNIM SISTEMOM

Identifikovanje učesnika u korišćenju budućeg poslovnog sistema

Korisnici novog poslovnog informacionog sistema klinike Hipokrat bi bili

- Menadžeri – kojima je važno da dobiju sistem izveštavanja i kontrole
- Medicinski radnici – koji imaju zadatak da evidentiraju pacijente, zakazuju im preglede i vrše naplaćivanje usluga
- Lekari – koji imaju dužnost da unose izveštaje o obavljenim pregledima, evidentiraju terapije, dijagnoze i propratnu medicinsku dokumentaciju
- Administrativni radnici – čiji je zadatak da upravljaju dnevnim, nedeljnim i godišnjim rasporedima rada, registruju nove zaposlene i vode računa o zalihama i nabavci materijalnih sredstava

Direktni korisnici (akteri) nisu jedini koje zanima razvoj ovog sistema. Jedino ime za sve zainteresovane strane za razvoj softverskog proizvoda je stejkholder. Oni ne koriste sistem direktno, ali mogu da utiču na njegov razvoj, da imaju izvesne zahteve ili očekivanja.

Eksterni stejkholderi:

- Pacijenti
- Banka
- Dobavljači

Interni stejkholderi:

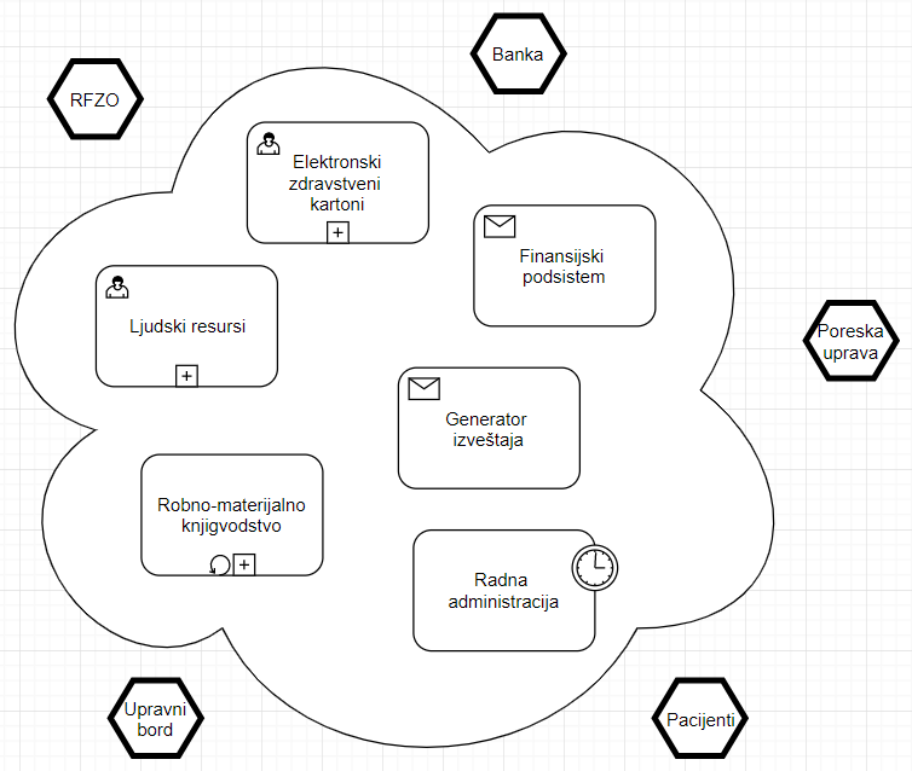
- Upravni bord
- Predstavnici marketinga

ANALIZA PODSISTEMA

Novi poslovni sistem privatne klinike bi imao nekoliko podsistema, koji će pružati različite servise različitim kategorijama korisnika.

Novi poslovni sistem privatne klinike bi imao nekoliko **podsistema**, na izgled odvojenih jer će pružati različite servise različitim kategorijama korisnika, ali će svi raditi sa istim podacima, što će sistem upravljanja konačno učiniti centralizovanim i održivim.

Na slici 1 su prikazane granice poslovnog sistema klinike. Sa unutrašnje strane su podsistemi, a sa spoljašnje faktori uticaja: stejkholderi i ograničenja.



Slika 6.1 Granice planiranog sistema klinike

OGRANIČENJA KOJIMA JE REŠENJE IZLOŽENO

Opis ograničenja

Rešenje je izloženo izvesnim ograničenjima. U tabeli na slici 2 su prikazana ograničenja, koja su razvojni tim novog sistema i menadžment klinike identifikovali:

ID	OPIS	OBRAZLOŽENJE
CON-1	Prva verzija treba da bude puštena u produkciju u roku od godinu dana od početka prikupljanja poslovnih zahteva.	Privatna klinika može pretrpeti poslovne gubitke, ako krene da gubi pacijente zbog loše organizacije rada ili bolje tehničke opremljenosti konkurencije.
CON-2	RFZO – Republički fond za zdravstveno osiguranje	Jako je jako važno da sistem ispoštuje standarde propisane od strane RFZO i da koristi njihove servise zbog automatskog pristupa šifrarnicima, fakturama i provere osiguranja.
CON-3	Banka	Cene usluga koje klinika nudi će se možda promeniti zbog naknade koju uzima banka za korišćenje POS terminala, koji je u planu da se uvede u upotrebu.
CON-4	Poreska uprava	Cena usluga koje klinika nudi su podložne promenama i shodno izmenama poreskog zakona.
CON-5	Obuka zaposlenih	Prilikom određivanja dužine trajanja i cene projekta, treba uzeti u obzir i obuku zaposlenih, koji do sada nisu radili sa naprednim sistemima.
CON-6	Kupovina gotovih komponenti	Dozvoljena je kupovina gotovih komponenti, sve dok bezbednost, fleksibilnost i nadgradnja sistema nisu u opasnosti zbog ograničenja gotovih proizvoda.

Slika 6.2 Tabela sa ograničenjima sistema

OPIS ZADATKA ZA VEŽBU

Rešavanje problema na osnovu opisa sistema

Sistem za upravljanje radom taksi udruženja je sistem koji treba da obuhvati sve potrebe rada jednog taksi udruženja i da obezbedi automatizaciju svih procesa koji su do sad obračunavani ručno.

Opis sistema i njegovi problemi:

Next je novoosnovano taksi udruženje koje raspolaže sa 20 taksi vozila. Centrala ovog taksi udruženja evidentira sve pozive i putem SMS poruke obaveštava svoje vozače na koje lokaciji treba da se pojave i za koji vremenski period je obavešten korisnik taksi usluge da će se vozilo pojaviti na navedenoj adresi. Taksi vozač odgovara na dobijenu poruku sa “Prihvaćeno” ili “Odbijeno” u odnosu na to da li može da ispuni zahtev centrale u odnosu na trenutnu

lokaciju ili vožnju koju trenutno obavlja. Centrala putem softvera koju trenutno kompanija koristi locira sve vozače i, ukoliko neki od vozača odbije vožnju, radnik u centrali samostalno procenjuje ko bi od ostalih vozača mogao da preuzme tu vožnju i dalje njemu prosleđuje zahtev. Taksi vozač je dužan da svaku vožnju evidentira u odnosu na kilometražu i račun koji je ispostavio taksimetar. Evidenciju vrši ručno i sve evidentirane vožnje predaje službeniku u centrali na kraju smene svakoga dana. Službenik u centrali u postojećem softveru evidentira broj vožnji po taksi vozilu i prosleđuje dnevni obračun finansijskoj službi. Finansijska služba na osnovu dnevnog obračuna dalje vrši sve potrebne obračune. Vozač taksi vozila ima zadatak da svakog drugog dana odvozi vozilo na pranje, što se takođe evidentira u dnevnom obračunu koji se prosleđuje finansijskoj službi. Vozač je takođe dužan da vodi evidenciju o svim kvarovima i potrošenom gorivu. Kako bi udruženje bilo konkurentno na tržištu, potrebno je evidentirati probleme ovog sistema i na što efikasniji način izvršiti automatizaciju istih.

ZADATAK ZA VEŽBU

Na osnovu opisa problema, probajte da uvidite potrebe korisnika.

Sistem koji treba razviti očekuje se da ispuni sledeće opšte ciljeve:

- Obezbeđivanje pravovremene komunikacije između centrale i taksi vozača
- Samoprocenu i delegiranje vožnji taksi vozačima
- Obračunavanje cena i izdavanje računa
- Evidentiranje vožnju, popravki, održavanja vozila od strane vozača
- Kreiranje osnovnog dnevnog obračuna
- Obezbeđivanje potrebnih statistika i finansijskih obračuna

Na osnovu opisa sistema, pokušajte da razumevanjem potreba korisnika identifikujete:

1. Glavne probleme i na koga utiču (15 min)
2. Korisnike i stejkholdere (10 min)
3. Podsisteme (10 min)
4. Ograničenja kojima je rešenje izloženo (10 min)

Uputstvo za izradu vežbe:
Prilikom izrade vežbe pođite od tog šta su problemi, koja su očekivanja korisnika i koje ciljeve sistem mora da ispuni da bi se problemi rešili. Prilikom identifikovanja stejkholdera i korisnika treba zapravo identifikovati osobe/sisteme koji/kojim mogu da se reše problemi tj. ispune zadati ciljevi.
Podsistemi se mogu definisati kao glavne oblasti na koje se zahtevi odnose (svaki podsistem se sastoji od više zahteva koji imaju za cilj da ispune slične funkcionalnosti sistema)
Svaki student samostalno rešava zadatu vežbu tako što zapisuje odgovor na papiru ili računaru nakon čega se o zadatim temama vodi diskusija.

Poglavlje 7

Domaći zadatak

UPUTSTVO ZA IZRADU DOMAĆIH ZADATAKA

Odnosi se na svih 15 domaćih zadataka

Asistent dodeljuje svakom studentu jedan sistem sa spiska od 5 predloženih sistema. Isti sistem student treba da analizira i za naredne domaće zadatke, kad god je tako naglašeno u tekstu domaćeg zadatka.

1. Mobilna aplikacija za prodaju autobuskih karata
2. Poslovni sistem lanca restorana za prodaju i pripremu brze hrane
3. Sistem za podršku specijalizovane radnje za prodaju domaćih pita
4. Onlajn platforma za podršku rada auto škole
5. Mobilna aplikacija za onlajn saradnju sa personalnim trenerom

Mobilna aplikacija za prodaju autobuskih karata očekuje se da podrži:

- Pregled reda vožnje za određenu autobusku stanicu
- Registraciju korisnika
- Rezervaciju autobuskih karata
- Prodaju autobuskih karata
- Ostvarenje popusta i pogodnosti
- Praćenje istorije korišćenja međugradskog autobuskog prevoza

UPUTSTVO ZA IZRADU DOMAĆIH ZADATAKA - NASTAVAK

Opis zadataka

Poslovni sistem lanca restorana za prodaju i pripremu brze hrane očekuje se da podrži:

- Evidenciju menija
- Evidenciju procedure za pripremu hrane iz menija
- Evidenciju cenovnika i specijalnih cenovnika
- Evidenciju prodaje
- Upravljanje radom i ljudskim resursima
- Analizu rada i ljudskih resursa
- Analizu prodaje

Sistema za podršku specijalizovane radnje za prodaju domaćih pita očekuje se da podrži:

- Onlajn poručivanje proizvoda unapred
- Onlajn poručivanje proizvoda za dostavu
- Evidenciju ponude i cenovnika
- Ostvarenje pogodnosti i plasiranje specijalnih ponuda
- Analizu kupaca
- Upravljanje porudžbinama

Onlajn platforma za podršku rada auto škole očekuje se da podrži:

- Praćenje teorijskih časova na daljinu
- Evidenciju grupa kandidata i prisustva kandidata
- Pristup dodatnim materijalima i vežbanjima od strane kandidata
- Evidenciju i praćenje napretka, finansija i statusa kandidata
- Evidenciju i raspored rada predavača
- Obaveštavanje kandidata i predavača o održavanju onlajn časova

Mobilna aplikacija za onlajn saradnju sa personalnim trenerom očekuje se da podrži:

- Kreiranje korisnika
- Postavljanje tipova vežbi sa objašnjenjem i videom
- Kreiranje dnevnih planova treninga i ishrane za svakog korisnika
- Praćenje istorije saradnje
- Logovanje kreiranih korisnika
- Pregled planova treninga i planova ishrane na nedeljnom nivou
- Beleženje ličnog napretka
- Komunikacija sa trenerom

DOMAĆI ZADATAK 1

Tekst domaćeg zadatka

Za dodeljeni sistem, osmislite i odredite:

1. Poslovnu priliku budućeg sistema
2. Korisnike i stejkholdere
3. Podsisteme
4. Ograničenja kojima je rešenje izloženo

Napomene:

Zadatak treba rešiti po ugledu na studiju koja je data u vežbi. Zadatak se rešava opisno i šalje kao .docx fajl.

Rešenje zadatka pošaljite na mejl adresu predmetnog asistenta. Rok za izradu je definisan Plan i programom predmeta.

Poglavlje 8

Projektni zadatak

UPUTSTVO ZA IZRADU PROJEKTNOG ZADATKA

Obavezan je kontinualan rad na projektnom zadatku, počevši od prve nedelje nastave.

Pravila u vezi sa izradom projektnog zadatka na predmetu SE322:

1. Za projektni zadatak, student mora odabrati temu u roku od prve dve nedelje nastave, a već do kraja 2. nedelje nastave mora poslati predlog teme za projekat.
2. **Tema** se može odabrati iz liste projekata date u dokumentu *SE322-PZ-Uputstvo za izradu i spisak tema*. Alternativno, student može sam dati predlog sistema za koji želi da radi analizu i specifikaciju zahteva na ovom predmetu.
3. Predlog teme projektnog zadatka treba da bude dužine do 500 reči. Predlog treba da sadrži naslov teme i kratak opis sistema, uz navođenje organizacije za koju se izrađuje proizvod, koja njegova buduća namena, koji su predviđeni korisnici i koje bi bilo njegovo operativno okruženje.
4. Cilj je da student ima verziju 1.0 svog projekta do kraja 12. nedelje nastave. U 13. i 14. nedelji nastave, kroz dve iteracije, student radi sa asistentom na ispravljanju grešaka i dopuni projektne dokumentacije.
5. Za studente tradicionalne nastave rokovi za slanje elemenata projektnog zadatka su definisani u tabeli **Raspored aktivnosti na projektu po nedeljama**. Ukoliko student preda zadatak na vreme, ima mogućnost da osvoji maksimalan broj poena predviđen za taj zadatak (pogledati tabelu sa **listom dokumenta koje obuhvata projektni zadatak**). Kod prekoračenja predviđenih rokova, primenjuje se umanjuje broja ostvarenih poena za 30%. Ove i dodatne informacije u vezi sa izradom projektnog zadatka, potražiti u dokumentu *SE322-PZ-Uputstvo za izradu i spisak tema*.
6. Odbrana projekata i upis poena je nakon 15. nedelje nastave.

ZADATAK ZA RAD NA PROJEKTU

Tekst zadatka za rad na projektu

Kreirajte **predlog teme za projekat** koji ćete raditi na predmetu. Temu student može izabrati iz liste projekata date u dokumentu SE322-PZ-Uputstvo za izradu i spisak tema, a može i sami dati predlog sistema za koji želi da radi analizu i specifikaciju zahteva na ovom predmetu. Predlog teme projektnog zadatka treba da bude dužine do 500 reči. Treba da sadrži

kratak opis sistema, uz navođenje organizacije za koju se izrađuje proizvod, koja njegova buduća namena i koji su predviđeni korisnici.

Projekat koji asistent usvoji u drugoj nedelji nastave se, paralelno sa domaćim zadacima, izrađuje kroz naredne nedelje nastave.

ZAKLJUČAK

Šta smo naučili u ovoj lekciji?

U predavanju je naglašeno da se problemi u razvoju sw proizvoda najčešće odnose na zahteve sistema, jer zahtevi ne oslikavaju realne potrebe korisnika, često su nekonzistentni ili nekompletni, postoji nerazumevanje između onih koji specificiraju zahteve i inženjera zaduženih za razvoj i održavanje sistema.

Kroz najčešće postavljena pitanja vezana za zahteve sistema je učinjen pokušaj da se daju osnovne informacije i definicije vezane za razumevanje zahteva. Tako je objašnjeno šta su zahtevi, šta podrazumeva proces inženjeringa zahteva, šta se dešava kada su zahtevi loši, šta je dokument zahteva, šta su stejkholderi sistema itd. U predavanju je takođe napravljena razlika između procesnih i proizvodnih zahteva.

REFERENCE

Nastavni materijal pripremljen za studente se pravi s namerom da im omogući brži i skraćeni uvid u program lekcije, a na bazi jedne ili više referentnih udžbenika i drugih izvora. Nastavni materijal nije zamena za ove udžbenike, koje treba koristiti ako student želi da se detaljnije upozna sa nastavnom materijom. Očekuje se od studenta da poseduje bar jedan od navedenih udžbenika u Planu i programu predmeta.

Ova lekcija je urađena na bazi teksta datom **u poglavlju 1 knjige**: **Karl Wiegers, Joy Beaty, Software Requirements, 3rd ed., Microsoft, 2013**. Za detaljnije proučavanje i primere, studentima se preporučuje da pročitaju ovo poglavlje. Manji uticaj na sadržaj lekcije imaju ostale reference navedene u Planu i programu predmeta.