



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

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

## ❖ Uvod

### 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 inženjeringu 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 inženjeringu zahteva dok će u kasnijim 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*

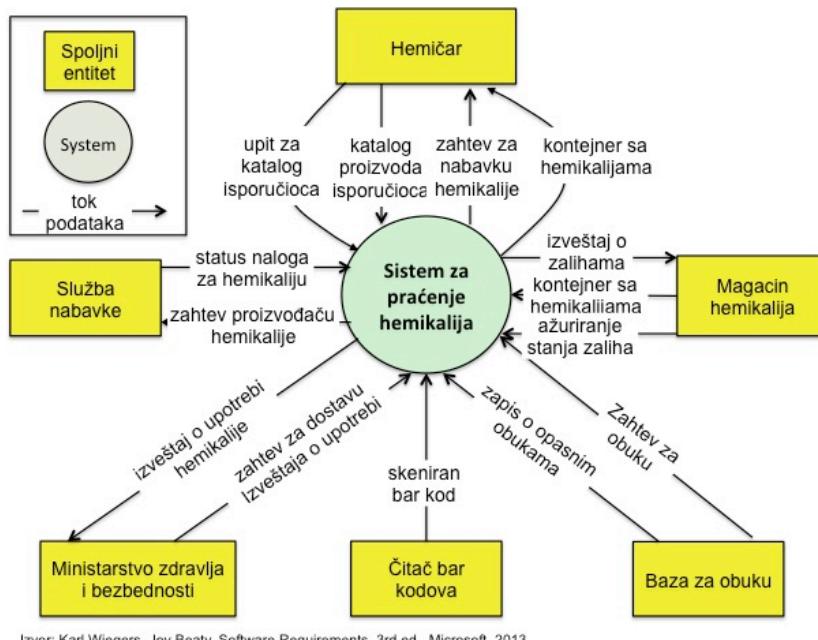
Da bismo ilustrovali metode za utvrđivanje, analizu i specifikaciju zahteva koje softverski sistem treba da ostvari, u predavanjima koristimo primer firme Contoso Pharmaceuticals koja koristi *Sistem za praćenje hemikalija*. To je deo informacionog sistema firme koji se odnosi na upravljanje hemikalijama koje firma koristi. U predavanjima se analiziraju zahtevi koje softverski podsistem, koji čini ovaj deo informacionog sistema firme, treba da zadovolji. Ovu studiju slučaja ćemo koristiti u svakoj lekciji, gde će se problem praćenja nabavke i korišćenja hemikalija 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 pravougaonima 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.



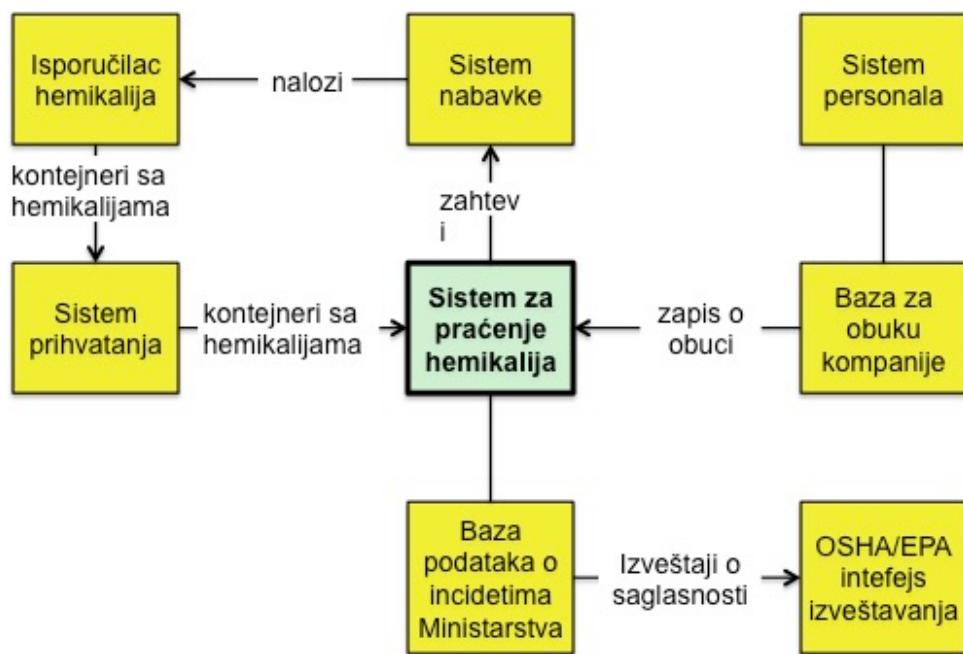
Slika 1.1 Kontekstni dijagram Sistema za praćenje hemikalija

## 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 pravougaonim (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 interfejs 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 Beatty, 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.*

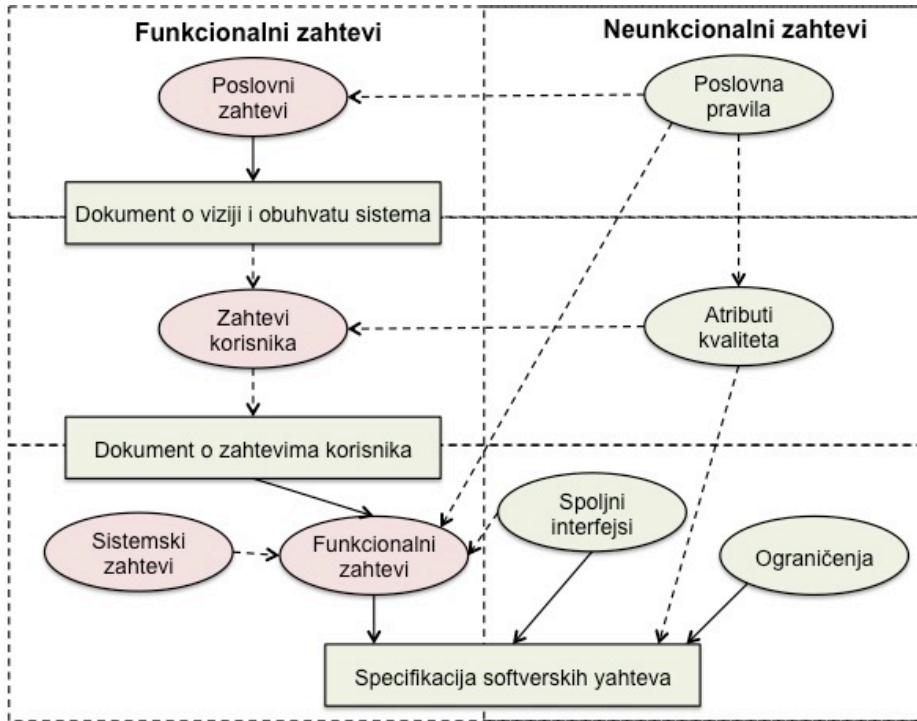
Zahtevi predstavljaju specifikaciju nečega što treba da bude primenjeno. Oni opisuju kako bi sistem trebalo da se ponaša, ili opis nekog svojstva sistema ili njegovog atributa. Oni ograničavaju, ali i usmeravaju proces razvoja sistema. Ovde pod sistemom podrazumevamo softverski sistem, tj. softver.

Postoji veliki broj vrsta informacija koje predstavljaju zahteve, one se klasificuju 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 pravougaonicima su označeni dokumenti koji sadrže različite vrste zahteva

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



Preuzeto iz: Karl Wiegers, Joy Beatty, 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** (business 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 definije 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 ljudе 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:

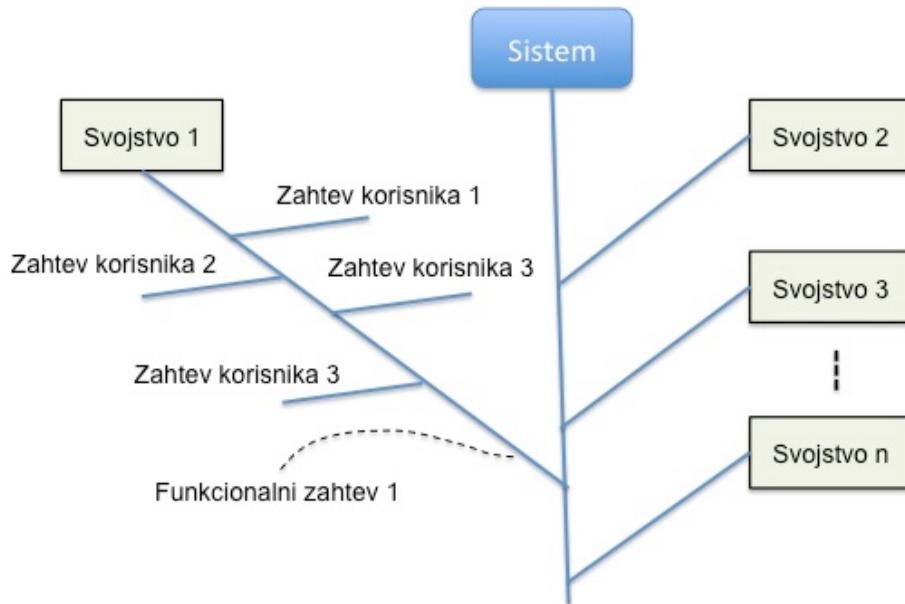
- Atributi kvaliteta (*quality attributes*) su faktori kvaliteta ili zahtevi za kvalitetom servisa, i opisuju karakteristike sistema u različitim dimenzijama koje su značajne ili za korisnike, ili za inženjere razvoja i održavanja, kao što su performanse, bezbednost, raspoloživost i prenosivost.
- Spoljni interfejsi (*external interfaces*) opisuju veze između sistema i spoljnog 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]

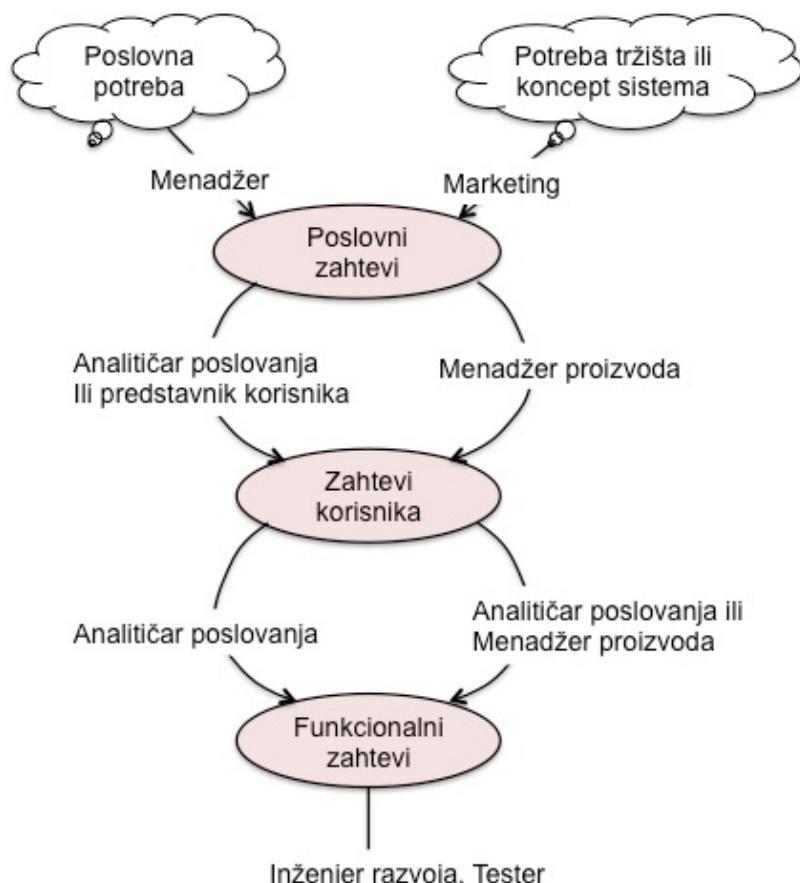
## ULOGE AKTERA U RAZVOJU FUNKCIONALNIH ZAHTEVA

*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 utvrđivanja 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 razvojnem 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.



Preuzeto iz: Karl Wiegers, Joy Beatty, 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 obrazovanim 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 attribute 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 zahteva 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 inženjerstvo podrazumeva sistematsko i višestruko korišćenje tehnika kojima se obezbeđuje da zahtevi sistema budu kompletni, konzistentni, relevantni itd. Inženjerstvo 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 inženjerstvo zahteva, ona često odnosi i na jedno i na drugo.

U ovom predmetu se naglasak stavlja na proces inženjerstva 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.

Inženjerstvo zahteva košta zavisno od tipa i veličine sistema koji se razvija i aktivnosti koje su uključene u inženjerstvo zahteva. U nekim slučajevima, sistemske zahteve 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 inženjerstva 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 što 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 sugerisu 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 podistem 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 skatoholdera, 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



Preuzeto iz: Karl Wiegers, Joy Beatty, Software Requirements, 3rd ed., Microsoft, 2013

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 veznih za njih. Kao što se vidi na slici 1, razvoj zahteva se realizuje izvršenjem sledećih aktivnosti:

- Prikupljanje 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 intervjuji, radionice, analiza dokumenata, izrada prototipa, i dr. Ključne akcije su:

- Utvrđivanje očekivane klase proizvoda korisnika i drugih aktera,
- Razumeti zadatka 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 ocjenjenog 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. Potvrđivanje 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

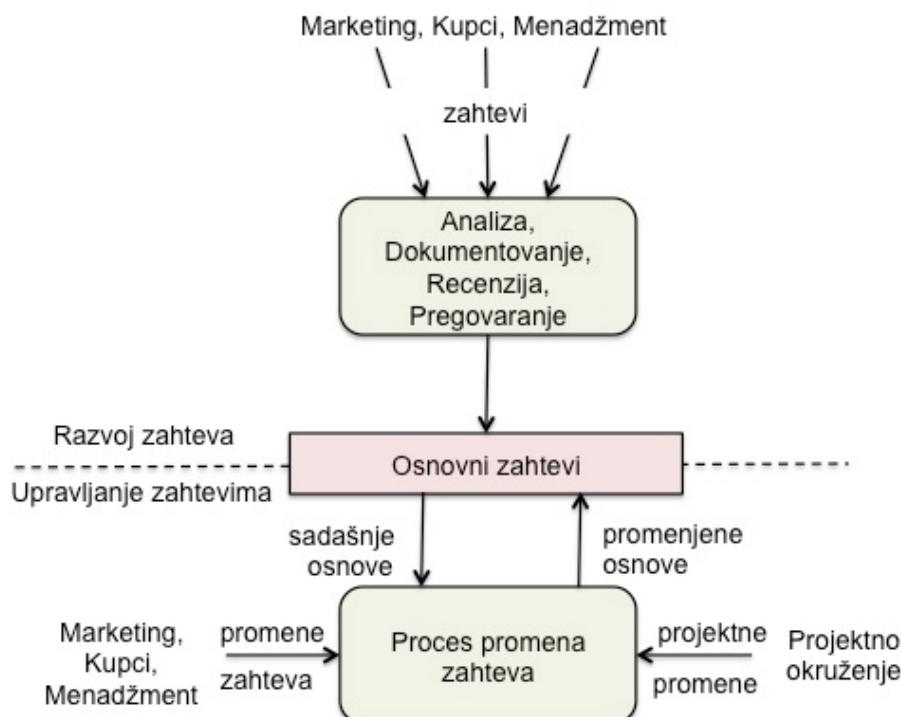
*Zahtevi se razvijaju, ali i stalno menjaju, tako da treba upravljati procesom menjanja zahteva*

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.



Preuzeto iz: Karl Wiegers, Joy Beatty, 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 servisom, 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, međutim, 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 apstrakcije 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 napor 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-orientisani razvoj, to znači da arhitektura sistema ne može biti objektno-orientisana.

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 preuzeti 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 zakašnjnjem i prevazilazi se planirani budžet*

Razvoj softverskih sistema je vezan sa različitim problemima: oni se često isporučuju sa zakašnjnjem 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 da bi 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 ljudi iz održavanja, kao i integratore SW.

## IDENTIFIKOVANJE KORISNIKA I AKTERA RAZVOJA

*Akter je bilo ko koje materijalno pogoden 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 stekholdera. 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 pogoden 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 klijentata, 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 grešaka
- 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, ljudi 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 tehniki 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.*

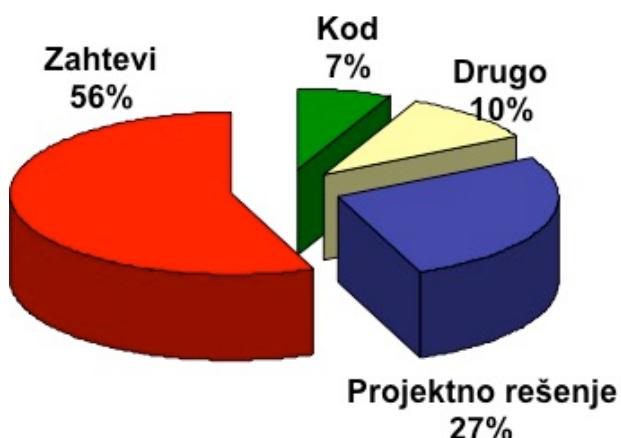
Američki nacionalni institut za standardizaciju i tehnologiju - NIST (National Institute of Standards and Technology) izdao je izveštaj o uspešnosti projekata razvoja softvera na bazi velikog broja projektata. Ustanovio je da:

- 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.

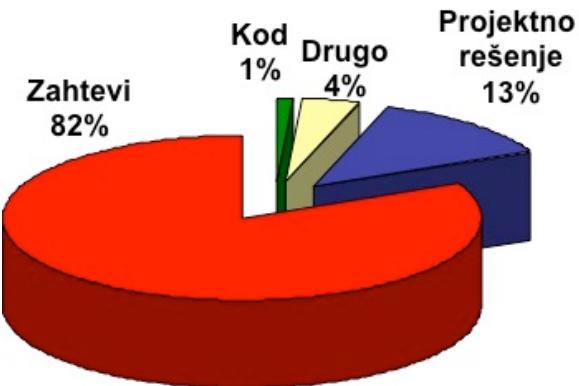
Izveštaj NIST-a zaključuje da je intezivno testiranje od ključne važnosti, ali testiranje utvrđuje greške, koje su dominantne u specifikaciji) u kasnim fazama procesa razvoja.

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)

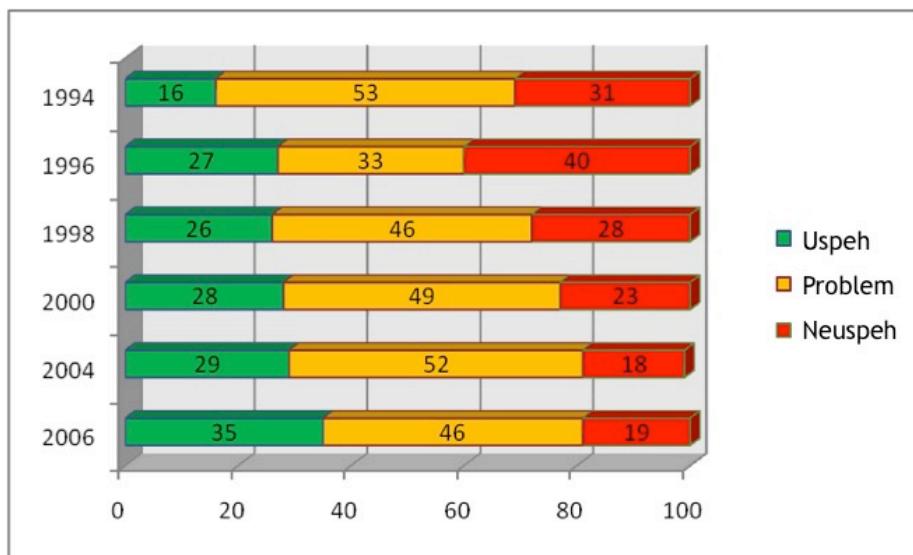
## USPEŠNOST PROJEKATA RAZVOJA SOFTVERA

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

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

- 53% projekata ne ispunjava osnovne projekte zahteve, te prekoračuju rokove, premašuju ugovoren 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.



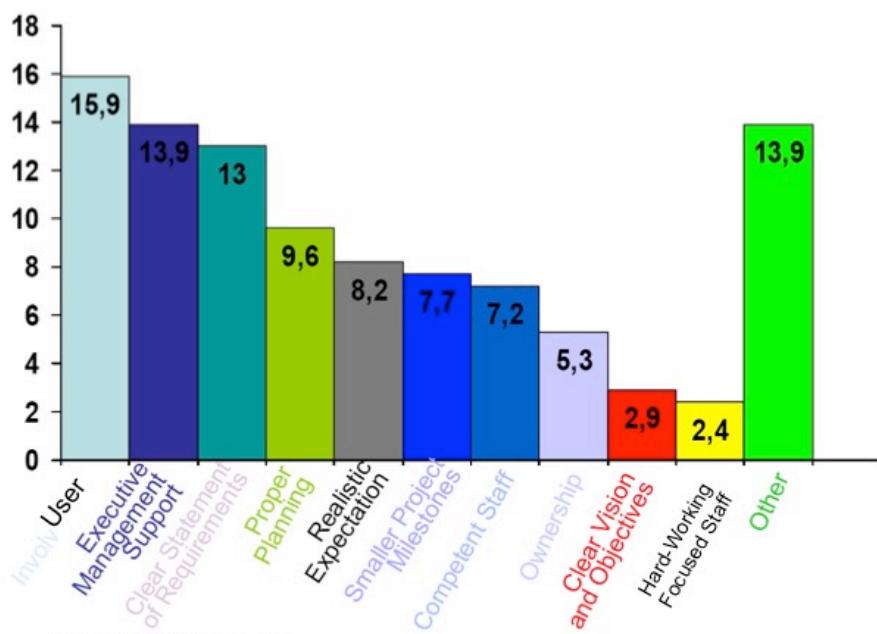
Source: Standish Group Inc., 1994-2006

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.*

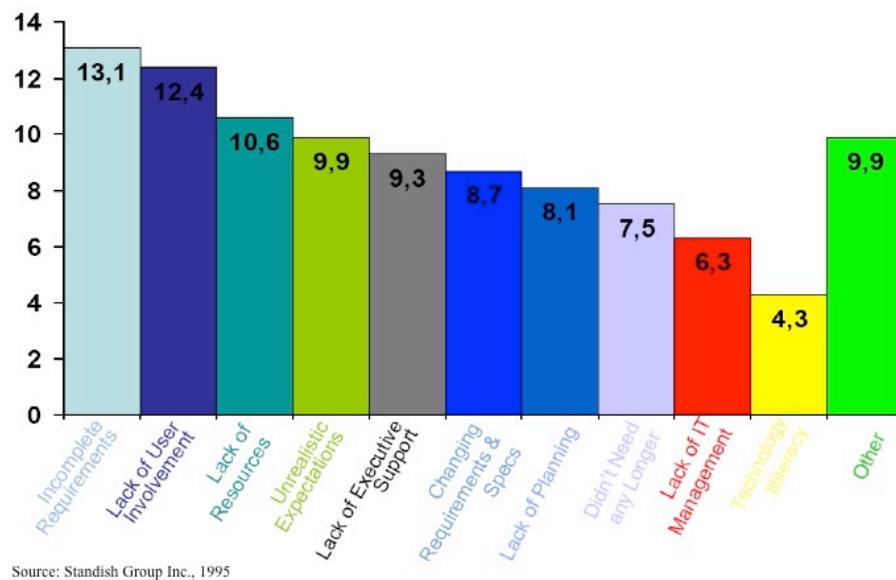
Na slici 4 prikazani su rezultati iz izvešaja koji pokazuju da je na uspešnost projekta razvoja softvera najznačajnije učešće korisnika softvera, a onda izvršni menadžment organizacije.



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.



Source: Standish Group Inc., 1995

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 pregledne 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 obavljuju 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 zalihamama 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:

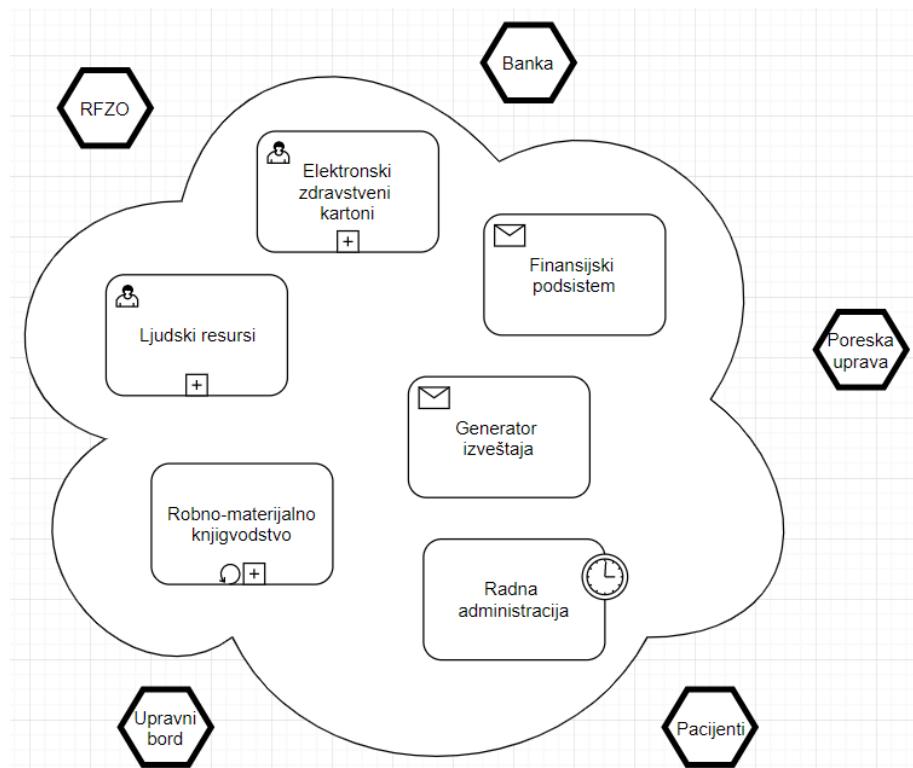
- Upravni bord
- Predstavnici marketinga

## 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 kreće da gubi pacijente zbog loše organizacije rada ili bolje tehničke opremljenosti konkurenčije.
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 šifarnicima, 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 lokacije 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 ispunji zahtev centrali 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
- Evidenciranje 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)

*Upustvo za izradu vežbe:*

Prilikom izrade vežbe podite 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 objašnjnjem i videom
- Kreiranje dnevnih planova treninga i ishrane za svakog korisnika
- Praćenje istorije saradnje
- Logovanje kreiranih korisnika
- Pregled planova treninga i planova ishrane na nedeljnem 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. Odrhana projekata i upis poena je nakon 15. nedelje nastave.

## ZADATAK ZA RAD NA PROJEKTU

*Tekst zadatka za rad na projektu*

Kreirajte **predlog teme za projekt** koji će 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

### ZAKLJUČAK

#### *Šta smo naučili u ovoj lekciji?*

U predavanju je naglašeno da se problemi u razvoju sv 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 inženjeringu 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 poglaviju 1 knjige: Karl Wiegers, Joy Beatty, 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.



## SE322 - INŽENJERSTVO ZAHTEVA

Modeli procesa

Lekcija 02

PRIRUČNIK ZA STUDENTE

# SE322 - INŽENJERSTVO ZAHTEVA

## Lekcija 02

### ***MODELI PROCESA***

- ✓ Modeli procesa
- ✓ Poglavlje 1: Dobra praksa u modeliranju procesa zahteva
- ✓ Poglavlje 2: Akteri procesa
- ✓ Poglavlje 3: Podrška projektu i upravljanje
- ✓ Poglavlje 4: Vežba
- ✓ Poglavlje 5: Domaći zadatak
- ✓ Poglavlje 6: Projektni zadatak
- ✓ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

## ❖ Uvod

### UVOD

*Šta ćemo naučiti u ovoj lekciji?*

U ovoj lekciji proučićemo modele procesa koji se koriste u inženjerstvu zahteva.

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

## ✓ Poglavlje 1

# Dobra praksa u modeliranju procesa zahteva

## VIDEO PREDAVANJE ZA OBJEKAT "DOBRA PRAKSA U MODELIRANJU PROCESA ZAHTEVA"

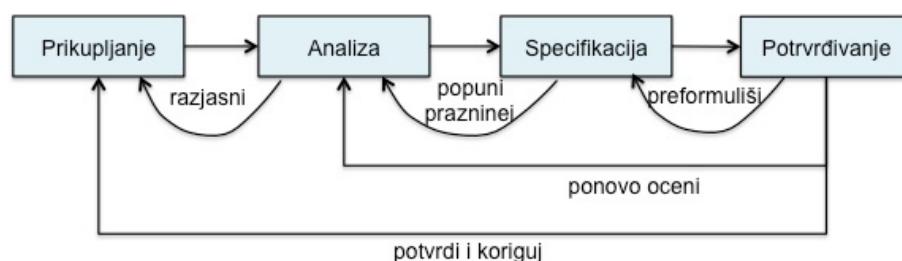
Trajanje video snimka: 34min 16sek

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

## AKTIVNOSTI PROCESA RAZVOJA ZAHTEVA

*Proces nije linearan, već ima i povratne sprege, te se iterativno realizuje.*

Na slici 1 prikazane su aktivnosti procesa razvoja zahteva. Taj proces nije linearan, već ima i povratne sprege, kao što se vidi na slici. To znači da su ove aktivnosti isprepletene, inkrementalne i iterativne. Ako obavljate ulogu analitičara poslovanja, onda ćete razgovarati sa korisnikom da bi utvrdili njegove zahteve. Onda ćete ih malo kasnije analizirati i eventualno zaključiti da treba da neke stvari da razjasnite sa korisnikom, te opet razgovarate sa njim. Sledeće što treba da uradite je da napišete u dokument o zahtevima, novi zahtev koji ste utvrdili. Ako tada utvrdite da je potrebna dodatna analiza, vi ćete je izvršiti. Kada ste spremili dokument sa zahtevima, dostavljate ga odgovarajućem akteru na potvrđivanje. Potvrđivanje može zahtevati da se neki zahtev dodatno razjasni sa korisnikom, ili da se neka analiza doradi ili da se izvrše neke preformulacije u dokumentu sa zahtevima. U slučaju agilnih projekata razvoja softvera, ovaj proces se ponavlja u svakom ciklusu.



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

Slika 1.1 Osnovne aktivnosti procesa razvoja zahteva

Zbog različitosti projekata i načina poslovanja organizacije, tj. organizacione kulture, ne postoji samo jedan model procesa razvoja zahteva. Ipak, na slici 2 prikazan je okvir procesa razvoja zahteva koji se može primeniti u većini slučajeva. On podrazumeva da je prethodno izvršena analiza poslovnih potreba ili mogućnosti tržišta. Prikazani zadaci se uglavnom izvršavaju po navedenom numeričkom redosledu, mada se neki od njih obavljaju i iterativno. Na primer, zadatke 8,9,10 možete raditi u delovima i da idete na recenziju (zadatak 12), posle svake iteracije.

Slika 1.2 Uobičajeni zadaci u procesu razvoja zahteva

## PRIKUPLJANJE ZAHTEVA

*Primenuju se različiti načini za prikupljanje zahteva od korisnika i drugih aktera.*

Sve vrste zahteva treba prikupiti primenom različitih tehnika. Ovde se daju neke preporuke, na osnovu prakse.

**Definiši viziju projekta i njegov okvir:** Dokument o viziji i okviru projekta sadrži poslovne zahteve za proizvod. Iskaz o viziji daje svim akterima zajedničko razumevanje u proizvodu koji treba razviti. Okvir definije granice sistema koji treba razviti, tj. šta treba, a šta ne treba. Na osnovu ovog dokumenta se vrši ocena predloženih zahteva. Dok je vizija, po pravilu, stabilna tokom trajanja projekta razvoja, iskaz o okviru je promenljiv u svakoj iteraciji.

**Utvrdi klase korisnika i njihove karakteristike:** Utvrdi grupe korisnike sistema, koje se razlikuju po frekvenciji korišćenja sistema, svojstvima koja koriste, nivou privilegija, ili iskustvu. Opiši njih posao, tj. radne zadatke, stavove, lokaciju, i dr.

**Odaberi "šampiona proizvoda" za svaku klasu korisnika:** Odaberi pojedinaca koji je tipičan za svoju klasu korisnika. On predstavlja potrebe svoje klase i donosi odluke u njeno ime. U slučaju razvoja proizvoda za tržište, odaberite pojedince iz firmi koje su glavni kupci ili rade testiranje beta verzija proizvoda.

**Vodite fokus grupu s tipičnim korisnicima:** Važne su pri razvoju komercijalnih proizvoda. Najčešće one ne donose nikakve odluke.

**Radi sa predstavnicima korisnika da bi utvrdio zahteve korisnika:** Sa predstavnicima razgovarajte šta očekuju od softvera da bi uspešno obavili svoj posao i koju vrednost očekuju da postignu. Zahtevi korisnika se najčešće daju u formi slučajeva korišćenja, priča korisnika i scenarija. Utvrdite potrebnu interakciju korisnik-sistem da bi obavio svoj posao.

**Utvrdi događaje u sistemu i moguće odgovore sistema:** Sačini listu spoljnih događaja i odgovora koje sistem treba da da na njih. Postoje tri klase događaja: signalni, i poslovni događaji.

**Organizuj intervjuje sa akterima:** Mogu biti pojedinačni i grupni. Oni su efikasni u prikupljanju zahteva, jer su fokusirani samo na zahteve određenog aktera.

**Organizuj radionice radi prikupljanja zahteva:** Omogućavaju kolaborativni rad analitičara i kupaca, radi utvrđivanja njihovih potreba i zahteva.

**Osmatraj rad ljudi na poslu:** Napravi dijagram procesa radnog mesta.

**Raspodeli upitnike:** Korisni u slučaju velikih i distribuiranih grupa korisnika.

**Prouči raspoloživu dokumentaciju:** Prikupi potrebne informacije.

Prouči prihvatljive postojeće zahteve, kao i izveštaje o problemima.

## SPECIFIKACIJA ZAHTEVA

*Svrha specifikacije zahteva je dokumentovanje zahteva različitog tipa na jedan konzistentan, pristupačan i proverljiv način koji je razumljiv akterima.*

Svrha **specifikacije zahteva** je dokumentovanje zahteva različitog tipa na jedan konzistentan, pristupačan i proverljiv način koji je razumljiv akterima sistema. **Poslovni zahtevi** se mogu zapisati u dokumentu o viziji i okviru sistema. Zahtevi korisnika su obično izražene u formi slučajeva korišćenja ili priča korisnika. Detaljniji funkcionalni i nefunkcionalno softverski zahtevi se zapisuju u specifikaciji softverskih zahteva (**SRS-Software Requirements Specification**) ili u alternativnom rezervorijumu pod kontrolom alata za upravljanje zahtevima. Evo preporuka dobre prakse:

- **Usvoji formular za dokumentovanje zahteva** - dokument o viziji i okviru projekta, obrazac za zapis slučajeva korišćenja i formular za specifikaciju softverskih zahteva (SRS).
- **Utvrди izvore zahteva** - daju indikaciju o razlozima za postavljanje zahteva. Na primer, to može biti neki slučaj korišćenja ili neko poslovno pravilo. Zapis o akterima na koje se odnose ova ograničenja se koristi za komunikaciju sa njim kada dođe do promene ovih zahteva. Praćenje izvora zahteva može da se realizuje preko linka datog u vidu atributa zapisa.
- **Jedinstveno označavanje zahteva** - u skladu sa konvencijom o oznakama zahteva koju treba da donešete. Označavanje zahteva omogućava njihovo praćenje i zapisivanje njihovih promena.
- **Zapisivanje poslovnih pravila** - ona uključuju pravila poslovanja firme, zakonsku regulativu, standarde, i algoritme računanja. Poslovna pravila važe i posle i pre projekta, i tome se razlikuju od projektnih pravila. Potrebno je zapisati linkove zahteva sa poslovnim pravilima koji su uzroci postavljanja tih zahteva.
- **Specificiraj nefunkcionalne zahteva** - uključuju kvalitativna svojstva, kao što su performanse, pouzdanost, upotrebljivost, promenljivost, i dr. Oni utiču na projektna rešenja softvera, koja na osnovu njih definiše projektant softvera.

## ANALIZA ZAHTEVA

*Analiza zahteva poboljšava zahteve, otklanja greške i nedostatke u njima, a omogućava bolje razumevanje zahteva od strane svih aktera.*

**Analiza zahteva** (*requirements analysis*) poboljšava utvrđene zahteve i obezbeđuje da ih svi akteri razumeju i ukažu na eventualne greške, ispuštene zahteve i druge nedostatke. Analiza uključuje dekompoziciju uopštenih zahteva na odgovarajući nivo detalja, izradu prototipova, ocenjuje ostvarljivost, i usklađuje prioritete zahteva. Cilj je dobijanje zahteva dovoljnog kvaliteta i sa dovoljnom preciznošću koje omogućavaju menadžerima da urade realističnu procenu projekta, a tehničkim članovima razvojnog tima mogućnost projektovanja, konstrukcije i testiranja softvera.

Vrlo je preporučljivo da se pojedini zahtevi prikažu na više načina, u tekstualnoj i u vizuelnoj formi, i u vidu zahteva i u vidu testova. Ovi različiti pogledi otkrivaju detalje koje ponekad samo jedan pogled ne može da otkrije. Višestruki pogledi takođe olakšavaju akterima da dođu da zajedničke vizije, tj. razumevanja šta treba da se razvije.

Analiza obuhvata sledeće aktivnosti:

- Određivanje prioriteta zahtevima - daje se prednost onim zahtevima koji omogućavaju najveću vrednost ili hitnu funkcionalnost.
- Kreiranje rečnika podataka - definicija podataka, olakšava komunikaciju
- Modeliranje zahteva - daje grafičko predstavljanje zahteva (model toka podataka, ER dijagrami, dijagrami stanja, mape dijaloga i dr.)
- Analiza interfejsa između sistema i spoljnog okruženja - to omogućava dobru usaglašenost sistema sa okruženjem.
- Dodeljivanje zahteva podsistemima - Složeni sistemi imaju podsisteme i preporučljivo je da im se dodele odgovarajući zahtevi.

## POTVRĐIVANJE ZAHTEVA

*Potvrđivanje (validation) obezbeđuje kontrolu da li su zahtevi korektni, i pokazuju karakteristike kvaliteta i zadovoljenje potreba korisnika*

**Potvrđivanje** (*validation*) obezbeđuje kontrolu da li su zahtevi korektni, i pokazuju karakteristike kvaliteta i zadovoljenje potreba korisnika. Ponekad zahteve treba korigovati da bi programerima bili jasniji i jednosmisleni. Ova aktivnost podrazumeva sledeće akcije:

- **Recenzija zahteva:** Recenzenti koji predstavljaju različite poglede na softver (analitičar, kupac, programer, tester) pažljivo analiziraju zapisane zahteve, modele analize, i povezane informacije, radi nalaženja grešaka. Od koristi je i preliminarna recenzija, dok su zahtevi još u fazi razvoja. Važno je obučiti članove tima za realizaciju efektivnih recenzija.
- **Testiranje zahteva:** Definisanje testova koji proveravaju da li su određeni zahtevi korisnika ostvareni
- **Definisanje kriterijuma prihvatljivosti:** Pitate korisnika da opiše kako bi on odredio da li je neko rešenje zadovoljava njegove potrebe i da je spremno za upotrebu. U to svrhu se definišu **testovi prihvatljivosti** softvera zasnovani na zahtevima korisnika, koji treba da pokažu zadovoljenje i određenih nefunkcionalnih zahteva, vođenja do grešaka, kao i posedovanje odgovarajuće infrastrukture i obučenog personala.

- **Simulacija zahteva:** Postoje komercijalni alati koji omogućavaju razvojnom timu da simuliraju rad razvijenog prototipa sistema. Primenom prototipova, analitičar može, u interakciji sa korisnikom, da utvrdi prihvatljivost pojedinih rešenja. Korisnik može da potvrdi valjanost rešenja ili da izvrši izbor varijantnih rešenje, ako mu se ponude.

## 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 2

### Akteri procesa

#### KORISNICI SOFTVERA

*Ne postoji jedna kategorija korisika. Korisnike treba razvrstati u više klasa korisnika.*

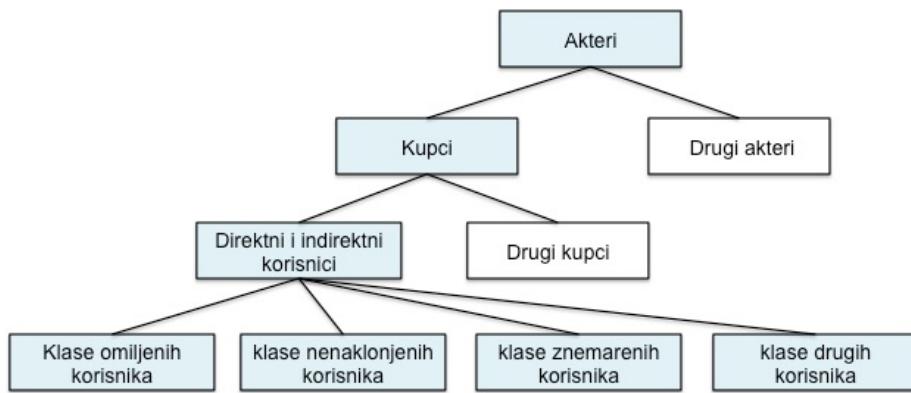
Uspeh u utvrđivanju zahteva softvera, a to znači i uspeh u razvoju softvera, prvenstveno zavisi od uključenja korisnika softvera u proces razvoja zahteva. Zbog toga, potrebno je da uradite sledeće:

- Utvrditi različite klase (kategorije) korisnika softvera koji treba da se razvije
- izaberite i radite sa pojedincima koji predstavljaju pojedine klase korisnika.
- Dogovorite ko će biti donosioci odluka u vezi zahteva u vašem projektu

Međutim, ovaj posao nije tako jednostavan kao što izgleda. Ako bukvalno usvojite sve što su vam predstavnici korisnika rekli i razvijete takav softver, vrlo je verovatno da to neće biti uspešan proizvod. I korisnici ne znaju tačno šta žele, a postoji i mogućnost da ih niste dobro razumeli. To što oni "žele" ne znači automatski da je to funkcionalnost koju softver treba da ima. Analitičar mora da prikupi znatno širi broj predloga korisnika, da ih analizira i razjasni i da ustaniši šta je stvarno potrebno da softver ima da bi korisnici mogli da uspešno rade svoj posao. U tome je vodeća odgovornost analitičara poslovanja i da to uskladi sa ostalim akterima procesa razvoja. To usklađivanje zahteva i definisanje svojstava softvera je jedan iterativan proces koji zahteva određeno vreme. U suprotnom, lako se može desiti da dobijete softver koji mora da se dorađuje, koji kasni, koji je probio budžet i izazva nezadovoljstvo korisnika.

Ne postoji jedna kategorija korisnika. Nema monolitnog korisnika. Uvek ima više kategorija korisnika koje mi opisujemo posebnima klasama korisnika. Ko što se vidi na slici 1, **klasa korisnika** je podskup šireg skupa korisnika, a skup korisnika je podskup aktera procesa. Jedan pojedinac može biti deo više ovih podskupova, te i da pripada većem broju klasa korisnika. Kako se korisnici razlikuju, oni se predstavljaju različitim klasama korisnika. Te razlike se ogledaju u sledećem:

- imaju različita prava pristupa ili nivoje bezbednosti
- imaju različite zadatke na poslu u vezi korišćenja softvera
- koriste različita svojstva softvera, različite frekvencije korišćenja softvera, različite domene primene ili IT kompetencije i dr.



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

Slika 2.1 Hjерархија актера, купача, корисника и класа корисника softvera

## OMILJENI I NENAKLONJENI KORISNICI

*Svaka klasa korisnika ima svoj skup zahteva u skladu sa zadacima koje članovi klase treba da izvrše.*

Pri klasifikaciji korisnika prvenstveno treba gledati posao koji obavljaju korišćenjem softvera koji treba razviti, a ne zvanja, lokacija rada, nivo u kadrovskoj hijerarhiji i dr. Kada počnete da definišete slučajeve korišćenja, priče korisnika, da radite dijagrame procesa - ukazaće vam se jasnije koje sve kategorije (klase) korisnika imate u projektu.

Neke klase korisnika mogu da budu važnije od drugih. **Klase omiljenih korisnika** (**favored user class**) je ona čije zadovoljenje je blisko povezano sa ostvarenjem poslovnih ciljeva firme koja kupuje softver. Ovo su korisnici zbog koji se prvenstveno i razvija novi softver.

**Klase nenaklonjenih korisnika** (**disfavored user class**) su korisnici koji ne koriste softver iz legalnih, bezbednosnih ili sigurnosnih razloga. To su korisnici koji mogu da budu i bespravni korisnici softvera, ili legalni korisnici, ali koji pokušavaju da koriste funkcije koje nisu njima namenjene i onda se suočavaju sa raznim problemima (pristup, nerazumevanje i dr.). Sa stanovišta softvera, sa korisnicima iz ove kategorije treba biti oprezan u davanju privilegija i pristupa različitim funkcijama sistema.

Koliko različitih klasa korisnika ćete koristiti i razviti, to zavisi od vas, tj. od specifičnosti vašeg softvera.

Svaka klasa korisnika ima svoj skup zahteva u skladu sa zadacima koje članovi klase treba da izvrše. Njihovi zahtevi se mogu preklapati.

Različite klase korisnika mogu imati različita očekivanja kvaliteta, kao što je način korišćenja softvera. To može da utiče na razvoj korisničkih interfejsa. Novi i povremeni korisnici pre svega žele jednostavan i razumljiv interfejs, te znači i dosta grafike. Iskusni i stalni korisnici žele pre svega efikasan interfejs, manje grafike, a više direktnog poziva funkcija koje su im potrebne, kao i manje klikova.

Korisničke klase mogu da predstavljaju i druge sistema, tj. da budu softverski agenti. Softverski agent može da za vaše potrebe obavi neki potreban posao i da vam isporuči rezultat takve obrade podataka.

Pored direktnih i indirektnih klasa korisnika, postoje i drugi mogući korisnici koje ne treba da zaboravite. Svakom od njih treba omogućiti servise koje očekuju od vašeg softvera.

## KAKO IZABRATI KLASE KORISNIKA?

*Primena organizacionih šema i poslovnih procesa može da olakša utvrđivanje klasa korisnika.*

Utvrđivanje klasa korisnika rano u projektu pomoći će vam da od njihovih predstavnika dobijete njihove specifične zahteve. Da bi utvrdili potrebne klase korisnika, pitajte sponzora projekta ko će sve da koristi softver. Razgovarajte sa svi navedenim mogućim korisnicima softvera. Grupišite ih prema sličnosti zahteva koje imaju. Tako možete doći i do 15-tak klasa korisnika.

Pri kreiranju klasa korisnika može vam pomoći i organizaciona struktura organizacije koje treba da koristi softver. Pri analizi aktera i korisnika, obratite pažnju na sledeće:

- Organizacione jedinice (OJ) koje koriste softver
- Organizacione jedinice koje su zahvaćene poslovnim procesom
- Organizacione jedinice ili nosioci zaduženja (**roles**) koji su direktni ili indirektni korisnici sistema
- Klase korisnika koje obuhvataju više organizacionih jedinica
- Organizacione jedinice koje imaju interfejs sa akterima van organizacije

Moguće je da utvrdite više klas korisnika i u okviru samo jedne organizacione jedinice. Analizom organizacione strukture, pokušajte da utvrdite i vrstu informacija korisnika iz svake OJ.

Dokumentujte utvrđene klasе korisnika i njihove karakteristike, odgovornosti i fizičku lokaciju.

Procenite obim transkacija sa sistemo koji svaka klasа korisnika može da ima. Na slici 2 prikazana je tabela klase korisnika u slučaju Chemical Tracking System, koji ćemo koristiti u ovom predmetu da bi na njemu pokazali kako se primenjuje znanje koje vam izlažemo.

Ime	Broj	OPIs
Hemičar (omiljeni)	Oko 1000 u 6 zgrada	Zahtevaju hemikalije od isoručioca i od skladišta himikalija. Svaki hemičar koristi sistem nekoliko puta dnevno, najčešće zahtevajući info o hemikalijama i o kontejnerima koje sadrže te hemikalije, preko kojih dolaze i odlaze hemikalije iz njihove laboratorije. Heičari pretražuju kataloge prodavaca za odredene hijiske strukture preuzete iz alata koji koriste dabi nacrtali te strukture.
Kupci	5	Kupci rade u odeljenju nabavke i obraduju zahteve za kupovinom. Oni šalju naloge za kupovinu isporučiocima hemikalija. Oni malo znaju o hemiji, i treba im jednostavni način za pretraživanje kataloga. Kupci ne koriste kontejnere sistema za praćenje lekova. Svako od njih koristi sistem oko 25 puta dnevno.
Osoblje skladišta hemikalija	6 tehničara i 1 supervizor	Skladište sadrži oko 500.000 hemijskih kontejnera. Isporučuju kontejnere iz 3 skladišta, zahtevaju nove hemikalije od isporučilaca, i prat edolazak i odlazak kontejnera iz skladišta. Oni su jedini korisnici iyeštaju o stanju skladišta. Zbog velikog obima, funkcije koje oni interno kroiste moraju da budu automatične i efikasne.
Osoblje OJ za zdravlje i bezbednost	1 menadžer	Koristi sistem samo za dobijanje kvartalnih izveštaja koji zadovoljavaju propise korišćenja hemikalija. Menadžer povremeno yahteva promene u iyeštajima u skladu sa promenama propisa. Ove promene imaju najviši prioritet i primena mora da bude brza.

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

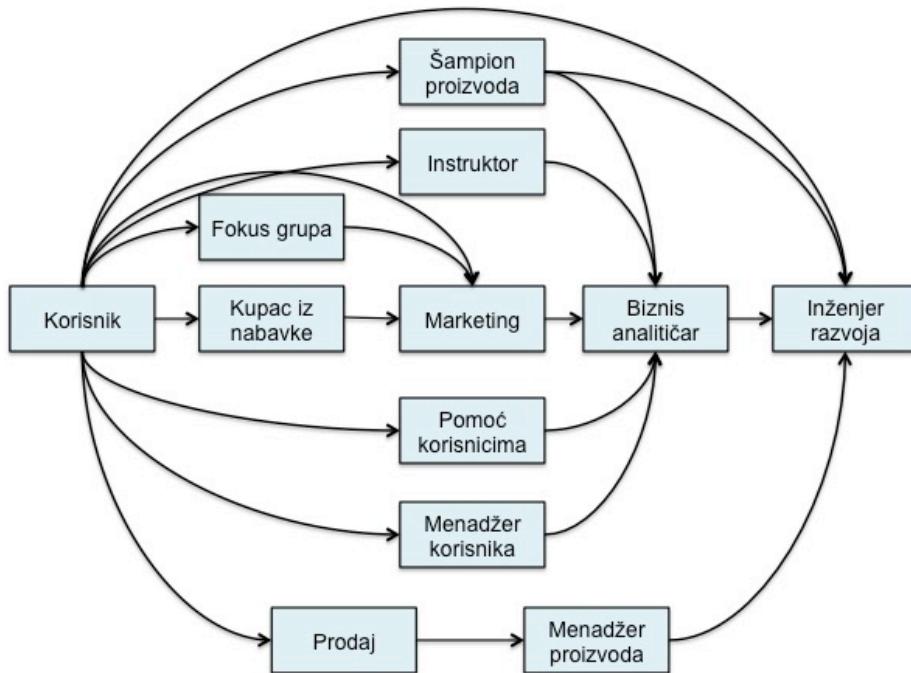
Slika 2.2 Primer klasa korisnika u slučaju ChemicalTracking System-a

## KOMUNIKACIJA SA PREDSTAVNICIMA KLASA KORISNIKA

*Neophodno je koristiti sve moguće vidove komunikacija između korisnika i inženjera razvoja. Treba biti svestan da posredništvo uvodi rizik, ali je nužno*

Neophodno je uključiti predstavnike sadašnjih korisnika sistema, a ako je sistem nov, predstavnike očekivanih prvih korisnika sistema (beta testeri). U tom cilju se pozivaju predstavnici različitih klasa korisnika na fokus grupe, radi utvrđivanja njihovih potreba. One treba da uključe i iskusnije, ali i manje iskusne korisnike softvera. Ne treba da uključite samo avangardne korisnike koji traže samo nove sisteme i svojstva, jer time bi zanemarili većinu drugih, običnih korisnika. Najbolje je da inženjer razvoja, ako ima dovoljno veštine analitičara, da razgovara sa krajnjim korisnicima sistema. Ali, to je moguće samo u slučaju manjih projekata. Kod većih to nije moguće, jer ima puno korisnika. Na slici 3 prikazane su neke od mogućih komunikacija sa korisnicima sistema.

Uvođenje posrednika između kupca i inženjera razvoja uvodi rizik nerazumevanja stvarnih potreba kupaca i njegovih zahteva. Veštih biznis analitičara eliminiše taj rizik i dodaje vrednost eliminisanjem manje važne ili nevažne zahteve korisnika. Rizik najviše unosi aktiviranje marketinga, menadžera proizvoda, stručnjaka za domenska znanja, i drugih koji treba da predstavljaju krajnjeg korisnika. Bez obzira na prepreke, pa i na troškove, treba optimizirati predstavljanje korisnika, jer će u suprotnom, vaš softver trpeti u kvalitetu (ne odgovara zahtevima korisnika) te treba da omogućite razgovor sa ljudima koji mogu da vam obezbede najbolju informaciju.



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

Slika 2.3 Komunikacije između korisnika i inženjera razvoja softvera

## ŠAMPION PROIZVODA

*Šampion proizvoda služi kao primarni interfejs između određene klase korisnika i biznis analitičara.*

**Šampion proizvoda** (product champion) služi kao primarni interfejs između određene klase korisnika i biznis analitičara. Idealno, oni su i krajnji korisnici, a ne neki posrednici. Šampioni proizvoda prikupljaju zahteve od drugih članova njegove klase korisnika i otklanjaju nekonzistentnosti. Na ovaj način, razvoj zahteva postaje zajednička odgovornost šampiona proizvoda i biznis analitičara, mada ovaj drugi i piše specifikaciju softverskih zahteva.

Najbolji šampioni projekta imaju jasnu viziju novog sistema. Šampion mora da bude dobar komunikator i da uživa poštovanje svojih kolega. Mora dobro da razume domen primene sistema, i radom okruženje sistema.

Šampion proizvoda postaje još efektivniji ako ima ovlašćenja da može da odlučuje u ime svoje klase korisnika. Međutim, on mora da ima podršku svojih kolega i da ne zastupa samo svoje, već i njihove stavove.

U slučaju razvoja novog komercijalnog sistema, šampiona proizvoda možete dobiti ili angažovanjem konsultanta ili angažovanjem predstavnika budućeg kupca. Vaša organizacija može i da zaposli ljudе koji su te poslove obavljali u nekoj drugoj organizaciji, gde su radili.

Kategorija	Aktivnosti
Planiranje	Poboljšava okvir i granice sistema Utvrđuje druge sisteme sa kojim sistem treba da bude u iterakciji Ocjenjuje efekat novog sistema na poslovne operacije kupca Definiše tranzicioni put od sadašnjih aplikacija ili ručnih operacija Utvrđuje relevantne standarde i zahteve sertifikacije
Zahtevi	Prikuplja zahteve ostalih korisnika u klasi koju predstavlja Razvija scenarije korišćenja, slučajeva korišćenja, i priče korisnika Rešava konflikte između zahteva iz svoje klase korisnika Određuje prioritete primene zahteva Obezbeđuje unos potrebnih performansi i drugih zahteva kvaliteta Ocjenjuje prototipve U saradnji sa drugima, rešava konflikte između zahteva različitih klasa korisnika Obezbeđuje specifične algoritme
Potvrđivanje (validation) i proveravanje (verification)	Recenzija specifikacije zahteva Definiše kriterijume prihvatanja Razvija testove prihvatanja korisnika iz scenarija korišćenja Obezbeđuje skup podataka za testiranja od poslovnog dela firme Realizuje beta testiranje ili testiranja prihvatanja od strane korisnika
Pomoć korisnicima	Piše delove teksta namenjen korisnicima sistema Učestvuje u pripremi materijala za obuku ili kurseve

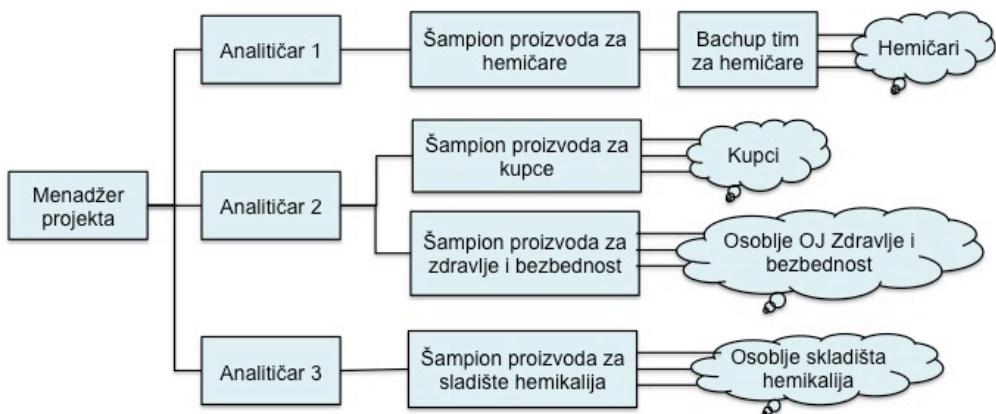
Izvor: Karl Wiegers, Joy Beatty, Software Requirements, 3rd ed., Microsoft, 2013

Slika 2.4 Aktivnosti šampiona proizvoda

## PRIMER ŠAMPIONA PROIZVODA - THE CHEMICAL TRACKING SYSTEM

*Broj biznis analitičara i broj šampiona proizvoda se određuju zavisno od vrste i veličine projekta.*

Obično jedna osoba ne može da bude šampion projekta za ceo sistem. U slučaju Chemical Tracking System, koriste se 4 šampiona proizvoda (slika 5). Oni su izabrani kao stručnjaci u kompaniji Contosa Pharmaceuticals, koja je i naručilac softvera. Kao što se vidi na slici, Menadžer projekta radi sa tri biznis analitičara i sa 4 šampiona proizvoda, da bi se prikupili zahtevi na pravi način. Šampioni proizvoda ne rade puno radno vreme za ovaj projekat, ali nedeljno mu posvećuju nekoliko sati. Jedan od biznis analitičara radi sa dva šampiona proizvoda, a ostali sa po jednim. Jedan od biznis menadžera je zadužen da sve prikupljene zahteve spoji i stavi u jedinsven SRS dokument (specifikacija softverskih zahteva).



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

Slika 2.5 Model šampiona proizvoda u slučaju CHT

Kako ima mnogo hemičara, šampion proizvoda za hemičare je formirao svoju back-up grupu koja mu pomaže da pokupe sve zahteve. Članovi ove grupe predstavljaju podklase hemičare. Na taj način su izbegnute velike radionice, koje nisu tako efikasne kao individualni razgovori sa predstavnicima podklasa krajnjih korisnika, u ovom slučaju, sa hemičarima.

## STAKEHOLDERS, VIEWPOINTS AND CONCERNS (VIDEO)

*Trajanje: 8:06 minuta*

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

## ▼ Poglavlje 3

### Podrška projektu i upravljanje

#### VIDEO PREDAVANJE ZA OBJEKAT "PODRŠKA PROJEKTU I UPRAVLJANJE"

*Trajanje video snimka: 11min 49sek*

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

#### PRIPREMA TIMA - OBUKE I PRIPREMNE AKTIVNOSTI

*Svi akteri u projektu, a posebno analitičari poslovanja moraju da prođu odgovarajuću obuku za razvoj softvera u određenom domenu primene.*

Obično neki članovi projektnog tima dobiju ulogu analitičara poslovanja na projektu, iako često i nemaju formalno obrazovanje za vršenje te uloge. Analitika poslovanja je specijalizovana i izazovna uloga. Pored neophodne obuke za izvršenje ovakve uloge, potrebno je da analitičar ima dobre komunikacione sposobnosti, da je zainteresovan za ovu ulogu i da u što većoj meri poznaje i oblast primene softvera, te da razume jezik korisnika, da bi razumeo i njihove potrebe. Potrebno je realizovati sledeće akcije:

- **Obuka analitičara poslovanja (business analysts):** U većoj ili manjoj meri, svi članovi tima vrše i ulogu analitičara poslovanja. Analitičar poslovanja trebalo bi da ima bar nekoliko dana obuke u različitim aktivnostima poslovne analize. To je osnova za izgradnju sopstvenog iskustva i dalje usavršavanje. On mora da bude strpljiv, da je dobro organizovan, da ima efektivnu veština komunikacije sa drugima, kao i da razume domen primene softvera.
- **Obuka aktera o zahtevima:** Najefektivnija obuka obuhvata više funkcionalnih disciplina, ne samo rad sa zahtevima. Akteri treba da imaju dan-dva obuke za rad sa zahtevima, da bi razumeli terminologiju, ključne koncepte i prakse, da bi shvatili zašto je inženjerstvo zahteva klučno za uspeh projekta razvoja softvera. Ova obuka je dobra i za razvoj sposobnosti tima i povezivanje članova tima.
- **Obuka inženjera razvoja o domenu primene softvera:** Organizujte seminar za inženjere razvoja na kome će se upoznati sa domenom primene softvera, sa aktivnostima kupaca, sa njihovom terminologijom, i sa ciljevima zbog kojih se realizuje projekat. To je i prilika da se upoznaju sa predstavnicima korisnika sa kojima će saradivati tokom trajanja projekta.

- **Definisanje procesa inženjerstva zahteva:** Dokumentujete sve aktivnosti procesa inženjerstva zahteva. Dajte uputstva kako da se one sprovedu da bi analitičari uspešno odradili svoj posao.
- **Kreiranje rečnika termina:** Rečnik termina sadrži potrebne termine iz oblasti domena primene sistema. Time se minimiziraju nerazumevanja. Uključite sinonime, skraćenice i dr. Takav rečnik postaje stalna vrednost organizacije i služi i u sledećim projektima. Može da posluži novim članovima tima da se upoznaju sa problematikom i terminologijom, što je neophodno da bi se uključili uspešno u projekat.

## UPRAVLJANJE ZAHTEVIMA

*Postoje razlike u načinu definisanja zahteva pri primeni različitih metoda razvoja softvera.*

Upravljanje zahtevima (requirements management) je peta osnovna aktivnost u razvoju zahteva. Upravljanje zahtevima obuhvata aktivnosti koje treba da uradite posle utvrđivanja zahteva, kao što je kontrola verzija dokumenata sa zahtevima, održavanje osnovnih zahteva, utvrđivanje statusa zahteva, i prenošenje zahteva drugim elementima sistema. Upravljanje zahtevima traje za celo vreme trajanja projekta, ali sa niskim intenzitetom.

Iako su osnovni zadaci razvoj zahteva uglavnom isti, bez obzira na primenjenu metodologiju razvoja softvera, ipak, postoje neke specifičnosti

Pri primeni čisto *metoda vodopada*, planirate samo jedno glavno izdanje softvera, te se skoro svi zahtevi utvrđuju na početku projekta. Kasnije se ipak vrši dopuna zahteva, ako je potrebno.,

Kod *modela iterativnog razvoja softvera* kao što je RUP (Rational United Process) utvrđivanje zahteva se vrši u svakoj iteraciji procesa razvoja softvera, s tima što se najviše zahteva ipak radi u prvoj iteraciji. To se radi i ako softver ima više izdanja.

*Agilne i druge iterativne metode razvoja softvera* proizvodi nova izdanja svakih nekoliko nedelja, te imaju česte aktivnosti razvoja zahteva, ali malog intenziteta. U početku se prikupi malo veći broj zahteva, a onda se oni raspodeljuju po sprintovima, zavisno od određenog prioriteta. Kada dođu na red, ovi zahtevi se mogu više detaljisati.

Bez obzira na primenjeni metod razvoja softvera, morate kod svakog izdanja ili iteracije da se pitate koji će od zadataka prikazanih na slici 2 u poglavlju 1, najviše dodati vrednost softveru, i koji rizik unosite u projekat sa ovim zahtevom.

Posle završetka zadatka br. 17, vi ste spremni za početnu konstrukciju odgovarajućeg dela softvera. Za svaki zahtev korisnike, obično treba da izvršite zadatke od 8 do 17, i tako se stvara osnova sledećeg izdanja softvra.

## PLANIRANJE PROJEKTA U VEZI SA ZAHTEVIMA

*Zahtevi se menjaju tokom trajanja projekta. Neophodno je uspostaviti sistem za upravljanje zahtevima i koristiti odgovarajuće alate za podršku tom sistemu*

Posle kreiranja prve verzije zahteva, dolazi do neophodnih izmena zahteva posle kontakta s korisnicima, menadžerima, marketingom, razvojnim timom i drugima koji su postavili neke zahteve. Efektivno upravljanje promenama zahteva obradu predloženih izmena, ocenjujući troškove njihove primene i uticaj na projekat, i obezbeđuje da odgovarajući akteri donešu odgovarajuće odluke o prihvatanju predloženih promena.

Preduslov za primenu efektivnog upravljanja promenama je korišćenje upravljanja konfiguracijom. Alat za kontrolu mora da ima istu verziju kao i vaš kod, da bi mogao da upravlja dokumentima sa zahtevima. Druga i bolja alternativa je da svi zahtevi budu pod kontrolom alata za upravljanje zahtevima. Primenuju se sledeće akcije:

- **Postavljanje procesa za kontrolu promena zahteva:** Neophodno je izbeći haos pri promeni zahteva. Zato je neophodno postaviti odgovarajući mehanizam kontrole promene zahteva. Upravljanje svim predloženim promenama u ovom procesu. Primena alata za otkrivanje grešaka može da podržava ovaj proces kontrole. Imenuje se mala grupa aktera projekta koji čine **Odbor za kontrolu promena** ( Change Control Board - CCB). On ocenjuje predloge za promenu zahteva, CCB odlučuje koji zahtev prihvatiti i postavlja prioritete primene usvojenih zahteva i određuje izdanje u kome će biti primjenjeni.
- **Izvršenje analize uticaja promene zahteva:** Pomaže da CCB doneše odluku o predlogu izmene zahteva. Vrši sa analiza uticaja na projekat za svaku predloženu izmenu zahteva. Koristi se **matrica povezanih zahteva** da bi se odredili ostali zahtevi, elementi projektovanja, izvorni kod, i testovi koji možda moraju da se promene. Identificuje zadatke koji treba da primene promene i procenjuje rad koji je neophodan za realizaciju predloženih promena.
- **Postavljanje početne vrednosti zahteva i kontrole verzija skupa zahteva:** **Početan skup zahteva** (requirements baseline) predstavlja dogovoren skup zahteva za određeno izdanje softvera. Njihova promena je moguća samo preko posebnog sistema za kontrolu promena zahteva, da bi se izbegle zabune pri radu sa prethodnim i novim verzijama zahteva, potrebno je primeniti jedinstveni identifikator za specifikaciju zahteva.
- **Održavanje istorije promene svakog pojedinačnog zahteva:** To je potrebno naročito ako želimo da se vratimo na prethodnu verziju zahteva, ili kada želimo da vidimo kako smo došli do sadašnje verzije zahteva. Zapisuje se datum izmene, opisuje izmenu koja je izvršena, osoba koja je izvršila promenu i razlog za promenu.

## PRAĆENJE REALIZACIJE PROJEKTA U VEZI SA ZAHTEVIMA

*Praćenje statusa zahteva, praćenje pitanja vezanih za zahteve, održavanje matrice povezivanja zahteva, upotreba alata za upravljanje zahtevima.*

- **Praćenje statusa svakog zahteva:** Postavite bazu čiji svaki slog nosi informaciju o jednom zahtevu. Odredi atribute za ključeve nalaženja svakog atribut, uključujući atribut o statusu zahteva (predložen, odobren, primjenjen, verifikovan) tako da možete da kontrolišete sve zahteve u istom statusu. Praćenje statusa svakog zahteva tokom trajanja projekta odražava i status projekta tokom svog trajanja.
- **Praćenje svih pitanja u vezi zahteva:** Alat za praćenje otvorenih pitanja vezanih za zahteve pomaže da se ta pitanja ne zaborave i da se na njih reaguje. Potrebno je da se neko imenuje za ovu aktivnost. Posmatra se status zahteva da bi se pružila ocena o stanju zahteva u projektu.
- **Održavanje matrice povezivanja zahteva:** Ponekad je neophodno pogledati vezu nekog zahteva sa elementima projektnog rešenja, koda koji primenjuje zahtev i testa koji ga verifikuje. **Matrica povezivanja zahteva** (requirement traceability matrix) pomaže da se utvrdi da li su svi zahtevi primjenjeni i verifikovani. Koriste se i u održavanju softvera kada treba da dođe do promene zahteva. Matrica povezivanja zahteva može da pomogne da se poveže neki funkcionalni zahtev, sa zahtevima višeg nivoa apstrakcije (npr. poslovna pravila), da bi se videlo odakle zahtev potiče. Matricu treba popunjavati za vreme razvoja, a ne tek na njegovom kraju. Podrška odgovarajućeg alata je vrlo preporučljiva, sem u najmanjim projektima.
- **Upotreba alata za upravljanje zahtevima:** Komercijalni alati za upravljanje zahtevima omogućuju vam da sve tipove zahteva stavite u jednu bazu podataka. Ovakav alat vam omogućava da automatizujete najveći deo operacija nad zahtevima koji je ovde opisan.

## UPRAVLJANJE PROJEKTOM

*Na osnovu prikupljenih i prihvaćenih zahteva, potrebno je planirati njihovu primenu. Potrebne akcije: izbor vrste projekta, plan razvoja zahteva, procena potrebnog rada, i plana projekta.*

Upravljanje softverskim projektom je blisko povezano sa procesima projektnih zahteva. Menadžer projekta treba da podesi plan projekta, resurse, i zaduženja članovima tima na osnovu zahteva koje treba da se primene.

Alternativna strategija je primena vremenske kutije i ciklusa. Projektni tim treba da proceni rad koji se treba uložiti u jednu iteraciju fiksnog trajanja, što je slučaj kod alternativnih metoda razvoja softvera. Opseg funkcija se pregovara, a u skladu sa fiksnim trajanjem ciklusa ("Šta se može uraditi u te periodu?"). Vlasnik projekta može da zahteva šta hoće, ali mora da

odredi prioritete po kome će tim razvijate softver u iterativnim ciklusima. Evo akcija koje treba realizovati:

- **Izbor odgovarajućeg životnog ciklus razvoja proizvoda:** Svaki menadžer projekta treba da izabere životni ciklus projekta koji najviše odovara specifičnostima njegovog projekta. Pri tome treba da uključi definiciju zahteva u definicije životnog ciklusa projekta. Ako je moguće, treba da specificira i primeni skup funkcionalnosti inkrementalno kako bi isporučio prve verzije softvera što pre.
- **Planiranje zahteva:** Svaki projektni tim treba da planira kako će razvijati zahteve i aktivnosti upravljanja. Plan prikupljanja zahteva treba da omogući da dobijete predloge različitih aktera u pravim fazama projekta upotrebom tehnika koje najviše odgovaraju projektu. Poslovni analitičar (BA) i menadžer projekta treba da rade zajedno radi obezbeđenja da se zadaci i rezultati koji odgovaraju inženjerstvu zahteva nađu zajedno u planu upravljanja projektom.
- **Procena potrebnog rada:** Akteri obično žele da znaju u koje vreme se mogu dobiti zahtevi i koji procenat rada na projektu se treba dodeliti inženeringu zahteva. To zavisi od mnogih faktora. Pretpostavite vreme malo veće od prosečnog, kako bi imali dovoljno vremena da obezbedite zahteve koji treba da budu solidan temelj za razvoj.
- **Izrada plana projekta u skladu sa zahtevima:** Razvijajte plan i termine iterativno, kako opseg i detaljni zahtevi postaju jasniji. Počnite se procenom troškova i vremenskih rokova u skladu sa početnom vizijom i opsegom projekta. U slučaju agilnih projekata iteracije u formi vremenske kutije omogućuje planiranje u okviru fiksног trajanja jedne iteracije i ograničenih resursa.

## UPRAVLJANJE PROJEKTOM (NASTAVAK)

*Potrebne akcije: određivanje donosioca odluka o zahtevima, promena zahteva, upravljanje rizicima u vezi sa zahtevima, praćenje uloženog rada, i analiza stečenog iskustva.*

- **Utvrđite donosioca odluka u vezi zahteva:** Treba razrešiti konfliktne zahteve aktera, izvršiti izbor komercijalnih komponenti, Zato je važno na početku odrediti ko su oni koji mogu da donose odluke u vezi zahteva.
- **Ponovo pregovarajte raspodelu zadataka kada dođe do promene zahteva:** Zadaci projektnog tima, i njihov kapacitet, određen je na osnovu početnog plana i budžeta. Sa unošenjem novih zahteva u projekat, izvršite ponovnu procenu da li sa raspoloživim resursima možete da ispunite početni plan i budžet. Ako ne možete, upoznajte se ovim menadžment i dogovorite novi, realističan raspored zadataka i broj ljudi. To se isto radi kada počnete procene koje su bile zasnovane na nedovoljno jasnim zahtevima, postanu problem.
- **Analizirajte, dokumentujte i upravljajte rizicima vezanim za zahteve:** Iznenadni događaji i uslovi, mogu da dovedu do opustošenosti u nepripremljenom projektu. Zato, utvrđite i dokumentujte rizike povezane sa zahtevima kao deo sistem za upravljanje rizicima u projektu. Razmatrajte pristupe za sprečavanje ili izbegavanje rizika, primenite akcije ublažavanja, i pratite nihovog dejstvo i napredak, kao i efektivnost.

- **Pratite uložen rad na zahtevima:** Da bi poboljšali vašu sposobnost procenjivanja potrebnog rada na zahtevima u budućim projektima, pratite troškove rada na zahtevima i za upravljanje aktivnostima u ovom projektu. Osmatrajte efekte koje ostvaruju vaše aktivnosti sa zahtevima na projekat i na ocenu povraćaja investicija u inženjerstvo zahteva.
- **Analizirajte stečeno iskustvo vezano za zahteve i u drugim projektima:** Organizacija koja uči vrši periodične retrospektive radi prikupljanja stečenog iskustva i "naučenih lekcija" ("Lesson learnt") iz završenih projekata ili iz ranijih iteracija sadašnjeg projekta.

## SOFTWARE PROJECT MANAGEMENT - WHY IT'S DIFFERENT! (VIDEO)

*Trajanje: 4:38 minuta*

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

## ▼ Poglavlje 4

### Vežba

#### VIDEO PREDAVANJE ZA OBJEKAT "VEŽBA"

*Trajanje video snimka: 34min 16sek*

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

#### PITANJA ZA DISKUSIJU

*Tekst pitanja za diskusiju*

##### **PITANJE 1.**

Zapišite probleme u vezi sa zahtevima sa kojima ste se susreli na svojim prethodnim projektima. Identifikujte svaki deo problema razvoja projekta ili problema upravljanja zahtevima. Opišite osnovni uzrok svakog problema i njegov uticaj na projekat. Imate li ideju koje promene u trenutnom načinu upravljanja zahtevima bi mogle da odgovore na ove probleme. (15 min)

##### **PITANJE 2.**

Uzmite 10 do 15 minuta i osmislite proces inženjerstva zahteva kakav smatrate da bi bilo idealno primenjivati u jednoj kompaniji koja se bavi razvojem softvera. Proces predstavite sistematično, u koracima i uz čvrste stavove. Pre nego što krenete u razrađivanje ideje, razmislite o nekoliko ključnih pitanja (30 min):

- Koje je veličine firma?
- Koju metodologiju razvoja softvera primenjuje?
- Na kakvom tipu projekata najčešće radi: interni projekti, outsourcing...
- Kakav je sistem upravljanja?
- Kakva je radna struktura?
- Da li postoji sistem za praćenje procesa rada i dokumentacije?

#### ZADACI ZA VEŽBU

*Tekst zadataka za vežbu*

##### **ZADATAK 1.**

Uzmite projektnu dokumentaciju koju ste radili za potrebe nekog od predmeta sa prethodne godine studija, poput SE201, SE211 ili CS324. Razmenite međusobno sa kolegama dokumentaciju. Svako treba da pročita tudi rad i obavi jednostavnu procenu dokumenta, a posebno onog dela koji se odnosi na zahteve. Svaki revizor treba da iznese svoja zapažanja i, ako ih ima, jasne predloge za poboljšanje. (15 min)

## ZADATAK 2.

Prepostavite da ste sistem analitičar koji treba da razvije novi modul u okviru Informacionog sistema univerziteta Metropolitan (ISUM) koji će koristiti samo studenti i koji će imati naziv E-Student. Razmislite i zatim odgovorite na sledeća pitanja:

- Šta biste izdvojili kao sistemske zahteve iz ugla korisnika budućeg sistema?
- Na osnovu sistemskih, koji bi bili softverski zahtevi?
- Šta biste naveli kao zahteve projekta?
- Koji bi bili zahtevi proizvoda?
- Šta biste identifikovali kao procesne zahteve?
- Da li postojali neki eksterni zahtevi?

Obratite posebnu pažnju na način izražavanja zahteva koje ste postavili. Podsetite se prvog predavanja da biste mogli adekvatno da odgovorite na sva pitanja. (30 min)

## ✓ Poglavlje 5

### Domaći zadatak

#### DOMAĆI ZADATAK 2

##### *Tekst domaćeg zadatka*

Za isti sistem koji ste dobili da analizirate za DZ01, identifikujte i navedite sledeće klase zahteva:

1. Zahteve projekta
2. Zahteve proizvoda
3. Procesne zahteve
4. Eksterne zahteva

Za svaku klasu zahteva navedite bar 2 primera zahteva koja nisu navedena u lekcijama.

*Napomene:*

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 6

### Projektni zadatak

#### ZADATAK ZA RAD NA PROJEKTU

*Tekst zadatka za rad na projektu*

Pošaljite asistentu na mejl **predlog teme za projekat** koji ćete raditi na predmetu. Projekat koji asistent usvoji u drugoj nedelji nastave se izrađuje kroz naredne nedelje nastave, paralelno sa domaćim zadacima.

## ✓ Zaključak

### ZAKLJUČAK

*Šta smo naučili u ovoj lekciji?*

1. **Proces razvoja zahteva** nije linearan, već ima i povratne sprege, te se iterativno realizuje.
2. Primjenjuju se različiti načina za **prikupljanje zahteva** od korisnika i drugih aktera.
3. **Analiza zahteva** poboljšava zahteve, otklanja greške i nedostatke u njima, a omogućava bolje razumevanje zahteva od strane svih aktera.
4. Svrha **specifikacije zahteva** je dokumentovanje zahteva različitog tipa na jedan konzistentan, pristupačan i proverljiv način koji je razumljiv akterima sistema
5. **Potvrđivanje** (validacija) **zahteva** obezbeđuje kontrolu da li su zahtevi korektni, i pokazuju karakteristike kvaliteta i zadovoljenje potreba korisnika
6. Ne postoji jedna **kategorija korisika**. Korisnike treba razvrstati u više klase korisnika. Svaka **klasa korisnika** ima svoj skup zahteva u skladu sa zadacima koje članovi klase treba da izvrše.
7. Primena organizacionih šema i poslovnih procesa može da olakša **utvrđivanje klase korisnika**.
8. Neophodno je koristiti sve moguće **vidove komunikacija između korisnika i inženjera razvoja**. Treba biti svestan da posredništvo uvodi rizik, ali je nužno kod većih projekata.
9. **Šampion proizvoda** služi kao primarni interfejs između određene klase korisnika i biznis analitičara. Broj biznis analitičara i broj šampiona proizvoda se određuju zavisno od vrste i veličine projekta.
10. Svi akteri u projektu, a posebno analitičari poslovanja moraju da prođu odgovarajući **obuku za razvoj softvera** u određenom domenu primene.
11. Postoje razlike u **načinu definisanja zahteva** pri primeni različitih metoda razvoja softvera.
12. Zahtevi se menjaju tokom trajanja projekta. Neophodno je **uspostaviti sistem za upravljanje zahtevima** i koristiti odgovarajuće alate za podršku tom sistemu.
13. **Praćenje statusa zahteva**, praćenje pitanja vezanih za zahteve, održavanje matrice povezivanja zahteva, upotreba alata za upravljanje zahtevima.
14. Na osnovu prikupljenih i prihvaćenih zahteva, potrebno je **planirati njihovu primenu**. Izbor vrste projekta, plan razvoja zahteva, procena potrebnog rada, uskladiti plan projekta sa zahtevima.
15. **Određivanje donosioca odluka o zahtevima**, promena zahteva, upravljanje rizicima u vezi sa zahtevima, praćenje uloženog rada, analiza stečenog iskustva

### 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 3 knjige: Karl Wiegers, Joy Beatty, 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.



SE322 - INŽENJERSTVO ZAHTEVA

## Razvoj poslovnih zahteva

Lekcija 03

PRIRUČNIK ZA STUDENTE

# SE322 - INŽENJERSTVO ZAHTEVA

## Lekcija 03

### *RAZVOJ POSLOVNIH ZAHTEVA*

- ✓ Razvoj poslovnih zahteva
- ✓ Poglavlje 1: Definisanje poslovnih zahteva
- ✓ Poglavlje 2: Dokument o viziji i okviru
- ✓ Poglavlje 3: Tehnike određivanja okvira
- ✓ Poglavlje 4: Držanje okvira u fokusu
- ✓ Poglavlje 5: Upotreba poslovnih ciljeva radi određivanja kraja
- ✓ Poglavlje 6: Vežba
- ✓ Poglavlje 7: Domaći zadatak
- ✓ Poglavlje 8: Projektni zadatak
- ✓ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

## ✓ Uvod

# UVOD

### *Uvodne napomene*

Poslovni zahtevi predstavljaju vrh lanca zahteva. Oni definišu viziju rešenja i okvir projekta koji treba da primeni to rešenje. Zahtevi korisnika i funkcionalni zahtevi moraju da budu u saglasnosti sa kontekstom i zahtevima koje postave poslovni zahtevi. Ne treba prihvati zahteve koji ne doprinose primeni postavljenih poslovnih zahteva. Oni omogućavaju zajedničko razumevanje svih aktera o tome šta sistem treba da omogući, a to je neophodno za uspeh projekta.

U ovoj lekciji opisaćemo dokument o viziji proizvoda i okviru projekta i analiziraćemo rezultate koji sadrže poslovne zahteve.

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

## ▼ Poglavlje 1

# Definisanje poslovnih zahteva

## VIDEO PREDAVANJE ZA OBJEKAT "DEFINISANJE POSLOVNIH ZAHTEVA"

*Trajanje video snimka: 37min 16sek*

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

### DEFINICIJA

*Poslovni zahtevi predstavljaju skup informacija koji, u zbiru, opisuje potrebu koje rezultat projekata razvoja softvera mora da zadovolji sa stanovišta zahteva.*

**Poslovni zahtevi** (business requirements) predstavljaju skup informacija koji, u zbiru, opisuje potrebu koje rezultat projekata razvoja softvera mora da zadovolji sa stanovišta zahtevanih poslovnih rezultata.

Poslovne zahteve čine poslovne mogućnosti, poslovni ciljevi, matrica uspeha i iskaz o viziji.

Pitanja poslovnih zahteva mora da budu razrešena pre nego što se u potpunosti definišu funkcionalni i nefunkcionalni zahtevi. Izkaz o okviru projekta znatno pomaže diskusiji o promeni zahteva koji se tokom projekta predlažu i u određivanju ciljnih izdanja softvera. Poslovni zahtevi su referenca pri donošenju odluka o predloženim izmenama zahteva i poboljšanjima zahteva. Preporučljivo je da se pri svakom predlogu zahteva, imaju u vidu poslovni ciljevi, vizija proizvoda i okvir projekta. To olakšava da se brzo uvidi da li je predloženi zahtev u okviru postavljenog okvira ili nije.

### UTVĐIVANJE POSLOVNE KORISTI OD PROJEKTA

*Poslovna korist treba da doneše istinsku vrednost sponzorima projekta i korisnicima proizvoda.*

Poslovni zahtevi postavljaju kontekst i omogućavaju merenje koristi koje biznis može da očekuje od projekta. Organizacije ne bi trebalo da započnu nijedan projekat bez jasnog razumevanja vrednosti koju projekat donosi biznisu, tj. uspešnosti poslovanja. Postavite

merljive ciljeve u vidu poslovnih ciljeva i onda definišite matricu uspeha koja vam omogućava da merite do koje mere ste na putu da zadovoljite ove ciljeve.

Sponzori projekta, koji ga finansiraju, rukovodstvo organizacije, menadžeri marketinga i vizionari proizvoda treba da odrede poslovne ciljeve projekta. Međutim, obično je izazov da se utvrde poslovne koristi koje projekat treba da ostvari, jer mogu postojati različiti stavovi aktera. Zbog toga, biznis analitičar treba da ima pripremljena pitanja i da zna koga treba da pita, da bi mogao da formuliše poslovnu korist koju projekt treba da omogući organizaciji.

**Poslovna korist** treba da donese istinsku vrednost sponzorima projekta i korisnicima proizvoda.

## VIZIJA PROIZVODA I OKVIR PROJEKTA

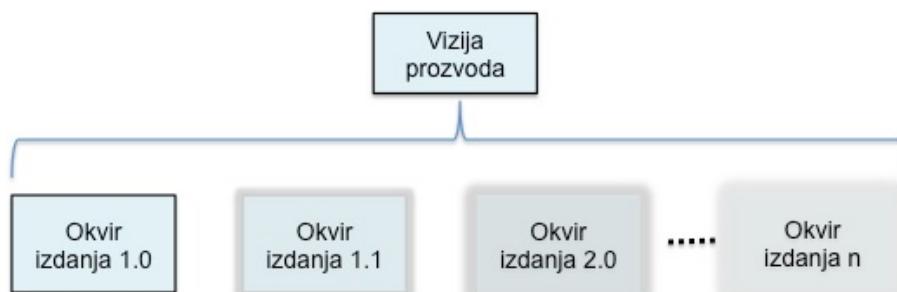
*Vizija proizvoda opisuje konačan proizvod koji će ostvariti poslovne ciljeve i ona je stabilna. Okvir projekta utvrđuje koji deo vizije proizvoda koji će biti predmet rada na projektu.*

Dva ključna elementa poslovnih zahteva su VIZIJA i OKVIR. **Vizija proizvoda** (product vision) opisuje konačan proizvod koji će ostvariti poslovne ciljeve. Ovaj proizvod može da bude kompletно rešenje za poslovne zahteve, ili samo deo rešenja. Vizija opisuje šta je proizvod i šta će biti u budućnosti. Daje kontekst za donošenje odluka za vreme životnog veka proizvoda i usmerava sve aktore u jednom određenom smeru,

**Okvir projekta** (project scope) utvrđuje koji deo vizije proizvoda će biti predmet rada na projektu ili u jednoj njegovoj iteraciji. Iskaz o okviru projekta određuje granicu između projekta i njegovog okruženja.

Vizija se odnosi na ceo projekat i vrlo je stabilna. Može se sporo menjati ukoliko dođe do promena poslovnih ciljeva.

Okvir je vezan za određeni projekat ili njegovu iteraciju i odnosi se na njegov sledeći inkrement i funkcionalnost koju obezbeđuje (slika 1). Okvir je dinamičniji od vizije jer akteri usklađuju njegov sadržaj za svako novo izdanje softvera u skladu sa terminom plana, budžetom, resursima, i ograničenjima kvaliteta. Okvir za sadašnje izdanje mora biti jasno. Međutim, okviri za buduća izdanja mogu biti manje jasni, i sve su manje jasni što su dalji u budućnosti. Cilj tima je da upravlja okvirom određenog projekta, kao definisanog dela strateške vizije proizvoda,



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

Slika 1.1 Okviri budućih izdanja su sve manje jasni što su dalje u budućnosti

## SUPROSTAVLJENI POSLOVNI ZAHTEVI

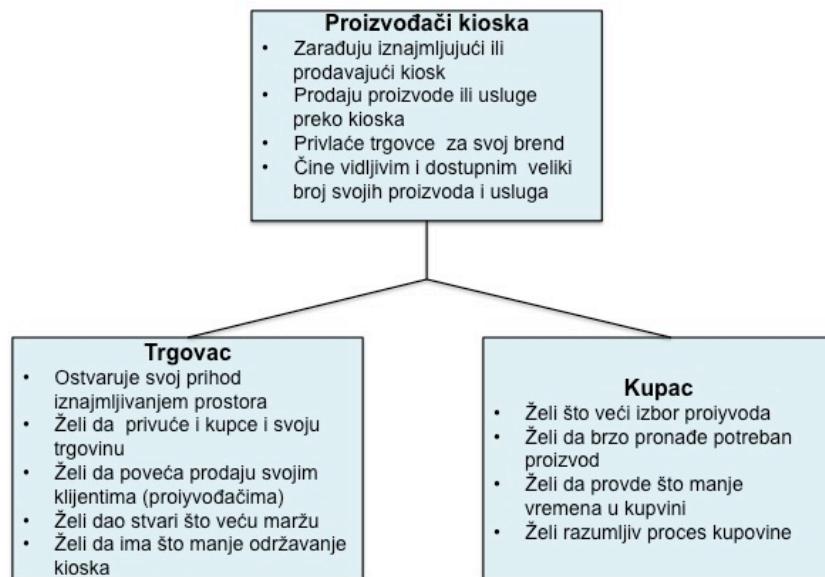
*Poslovni interesi aktera mogu biti i suprotstavljeni. Ako njihovo razrešenje dovede do menjanja poslovnih ciljeva, onda treba da zatražite i promenu plana projekta.*

Poslovni zahtevi dolaze iz različitih izvora i može se desiti da su neki od njih međusobno suprotstavljeni. Na primer, zamislite kiosk (odeljak u trgovini) u kome se prodaju proizvodi nekog brenda u nekoj trgovini .(slika 2). On pokazuje poslovne interese proizvođača kioska, koji nudi svoje proizvode, proizvoda, trgovca u čijoj trgovini se nalazi kiosk i kupca. Svaki od ovih aktera ima svoje poslovne ciljeve vezane za rad kioska.

Neki ciljevi ovih aktera su u međusobnoj saglasnosti. I proizvođači proizvoda i kupci žele da preko kioska imaju što veći broj različitih proizvoda na raspolaganju kupcu. Međutim, neki od poslovnih ciljeva mogu biti i međusobno suprotstavljeni. Na primer, kupac želi da za što kraće vreme obave kupovinu, dok prodavac želi da zadrži kupca što duže u svojoj trgovini (ne bi li još nešto kupio). Donosioci odluka u projektu moraju ovaj konflikt da razreši pre nego što biznis analitičar počne da radi na definisanju detaljnih softverskih zahteva. Fokus bi trebalo da bude obezbeđivanje najveće poslovne vrednosti primarno agentima. Kupca prvenstveno interesuje proizvod koji odgovara njegovim poslovnim ciljevima, ne bi trebalo da mu se nudi nešto van toga.

Ovakav konflikt ne treba da rešava softverski tim koji kreira sistem. Neko mora da kontroliše širenje okvira projekta zbog interesa pojedinih aktera.

Biznis analitičar uočava ovakve konfliktne situacije, ali njihovo razrešavanje je u domenu biznisa, a ne softvera, te i rešavanje takvih situacija se mora razrešiti na biznis nivou. Menjanje okvira i poslovnih ciljeva dolazi češće kod dužih projekata. Ako vam se ovakav slučaj desi, vratite se na početne poslovne zahteve. Zahtevajte da se utvrde razlike i da se, samim tim, izvrše promene u rokovima, budžetu projekta i resursima. Funkcionalni i nefunkcionalni zahtevi takođe onda treba da se promene.



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

Slika 1.2 Akteri jednog kioska nemaju uvek uskladene interese

## VIDEO 3 - THE BUSINESS VALUE OF BETTER REQUIREMENTS (VIDEO)

Trajanje: 7:09

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

## ▼ Poglavlje 2

### Dokument o viziji i okviru

#### ŠTA JE DOKUMENT O VIZIJI I OKVIRU

*Dokument o viziji i okviru projekta sadrži poslovne ciljeve koji su osnova za aktivnosti projekta.*

Dokument o viziji i okviru (vision and scope document) sadrži poslovne ciljeve koji su osnova za naredne aktivnosti na projektu. Organizacije koje razvijaju komercijalni softverski proizvod često ovaj dokument zovu "dokument sa zahtevima marketinga" (marketing requirements document - MRD). MRD ulazi više u detalje ciljnog tržišnog segmenta i pitanjima koje se odnose na poslovni uspeh softvera.

Vlasnik dokumenta o viziji i okviru je sponzor projekta, neko ko finansira projekat ili neko koji ima sličnu ulogu. Biznis analitičar može da pomogne ovima u formulisanju i u pisanju dokumenta o viziji i okviru. Ulaz u definisanje poslovnih ciljeva treba da obezbede oni koji pokreću projekt i koji najbolje znaju razlog za to pokretanje. Pokretač projekta može biti kupac-naručilac softvera ili rukovodstvo organizacije koje planira da proizvod (softver) prodaje na tržištu ili vizionar projekta, menadžer projekta, ekspert u domenu primene softvera, ili član marketing tima.

Na slici 1 je prikazan sadržaj dokumenta o viziji i okviru. Sadržaj se može prilagođavati specifičnostima projekta. Neki delovi dokumenta mogu biti isti kao u nekim sličnim projektima, te se mogu odatle i preuzeti.

Dokument o viziji i okviru definiše okvir samo u jednom opštijem nivou, jer se detaljnije informacije o okviru definišu posebno za svako izdanje softvera. Svako izdanje softvera ili iteracija ili poboljšanje softvera, može da ima svoj iskaz o okviru u dokumentima sa zahtevima, umesto da ima svoj poseban dokument o viziji i okviru.

## 1. Poslovni zahtevi

1. Osnova
2. Poslovna mogućnost
3. Poslovni ciljevi
4. Matrica uspeha
5. Iskaz o viziji
6. Poslovni rizici
7. Poslovne pretpostavke i zavisnosti

## 2. Okvir i ograničenja

1. Glavna svojstva
2. Okvir početnog izdanja
3. Okvir narednih izdanja
4. Ograničenja i isključenja

## 3. Poslovni kontekst

1. Profil sponzora
2. Prioriteti projekta
3. Razmatranju u primeni softvera

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

Slika 2.1 Uzorak sadržaja dokumenta o viziji i okviru

# 1. POSLOVNI ZAHTEVI (ODELJCI 1.1, 1.2, 1.3)

*Poslovni zahtevi opisuju primarne koristi koje novi sistem obezbeđuje sponzorima, kupcima i korisnicima. Oni direktno utiču na prioritet primene zahteva.*

Projekti se pokreću u verovanju da kreiraju softver ili da ga poboljšavaju da bi donele korist nekome i ostvario odgovarajući povraćaj investicije. **Poslovni zahtevi** (**business requirements**) opisuje primarne koristi koje novi sistem obezbeđuju sponzorima, kupcima, i korisnicima. Poslovni zahtevi direktno utiču na primenu zahteva korisnika i na prioritet primene.

### 1.1 Osnova

Sumiraju se razlozi i kontekst za razvoj novog sistema ili za poboljšanje postojećeg. Opisuje se istorijat ili situacija koja je dovela do odluke da se donese odluka o razvoju sistema.

### 1.2 Poslovna mogućnost

U slučaju razvoja informacionog sistema, daje se opis poslovnog problema koji se rešava, ili proces koji se poboljšava, kao i opis okruženja u kome će sistem biti korišćen. U slučaju razvoja komercijalnog softvera, opisuje se poslovna mogućnost i tržište na kome će se softver nuditi. U ovom delu dokumenta se mogu navesti uporedna analiza postojećih proizvoda, uz

navođenja zašto je novi proizvod ima prednost u odnosu na postojeće i u čemu je njegova atraktivnost. Opišite probleme koji se ne mogu rešiti bez primene novog proizvoda. Pokažite kako se on uklapa u tržišne trendove, razvoj tehnologije, i u strateške pravce poslovanja organizacija.

Navode se tehnologije, procesi, i resursi koji su potrebni da bi se ostvarilo rešenje za kupca, Daje se opis tipičnih kupaca ili ciljnog tržišta. Opisuje se problem kupca koje će se novim proizvodom rešiti. Daju se primeri kako će kupci koristiti novi proizvod. Daje se opis kritičnih interfejsa ili zahteva kvaliteta, ali se ne ulazi u specifičnosti projektovanja i implementacije.

**1.3 Poslovni** ciljevi Izložite na kvantifikovani i merljiv način važne koristi koje će proizvod obezbediti. Na slici 2 date prikaz uprošćenih finansijskih i nefinansijskih poslovnih ciljeva,

Finansijski poslovni ciljevi	Nefinansijski poslovni ciljevi
<ul style="list-style-type: none"> <li>• Pridobiti X% tržišnog učešća za Y meseci</li> <li>• Povećati u zemlji W učešće na tržištu od X% na Y% za Z meseci.</li> <li>• Povećati prodaju X jedinica ili prihod za Y\$ za Z meseca</li> <li>• Postići X% povraćaja investicija u roku od Y meseci.</li> <li>• Ostvariti pozitivan tok novca kod ovog proizvoda u roku od Y meseci.</li> <li>• Uštedeti X\$ godišnje koje sada trošimo za održavanje postojećih sistema.</li> <li>• Smanjiti mesечne troškove podrške od X\$ na Y\$ mesečno.</li> <li>• Podiži bruto marginu od postojećeg poslovanja od X% na Y% u toku 1 godine,</li> </ul>	<ul style="list-style-type: none"> <li>• Osvrati meru zadovoljstva kupca X u periodu od Y meseci od izdaha</li> <li>• Povećati produktivnost transakcija za X% i smanjiti broj grešaka na Y%</li> <li>• Razviti proširljivu platformu za familiju povezanih proizvoda</li> <li>• Razviti ključne tehnološke kompetencije</li> <li>• Biti ocenjen kao vrhunski proizvod po puuzdanosti u revijama koje se objavljuju do određenog datuma.</li> <li>• Usaglasiti se sa postojećim propisima.</li> <li>• Ne primiti više od X poziva po jedinici proizvoda i Y poziva za garnijske uskuge po jedinica u roku od Z mesecod isporuke.</li> <li>• Smanjiti vreme odziva od X časova na Y% poziva za podršku.</li> </ul>

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

Slika 2.2 Primeri finansijskih i nefinansijskih poslovnih ciljeva

## 1. POSLOVNI ZAHTEVI (ODELJAK 1.3 NASTAVAK)

*Poslovni zahtevi definišu način za merenje ostvarivanja ovih poslovnih ciljeva.*

Projekt se obično pokreće da bi se rešio neki problem. Problem pokazuje šta sprečava poslovanje da ostvari svoje ciljeve. Poslovni zahtevi definišu način za merenje ostvarivanja ovih ciljeva.

Kada imamo poslovne ciljeve, možemo postaviti pitanje "Šta nas sprečava da ostvarimo ove ciljeve?" da bi identifikovali detaljnije poslovni problem. Možete pitati "Zašto nam je važan ovaj cilj" da bi ustanovili poslovni problem ili mogućnost. Za dati poslovni problem, možemo postaviti pitanje "Kako ćemo oceniti da li je problem rešen?", da bi ustanovili merljive ciljeve. Proces je iterativan, obuhvata probleme i ciljeve (**objectives**) i sprovodi se sve do dobijanja liste svojstava koja pomaže rešavanju problema i zadovoljavanju poslovnih ciljeva.

Pitanje	Odgovor
Šta motiviše vaš interest u CTS?	Ručno upravljanje zalihamama mnogo košta i neefikasno je.
Koliko bi voleli da smanjite vaše troškove?	Za 25% godišnje
Šta vas sprečava da sada ih smanjite za 25%? Šta je razlog visokih troškova i neefikasnosti?	Kupujemo nepotrebne hemikalije jer ne znamo šta imamo na zalihamama. Becamo veliki deo hemikalija sa isteklim rokom.
Da li ima još nešto što treba da znam?	Naručivanje je komplijano. Uzima mnogo vremena korisnicima procesa. Izveštaje za državne organe ručno pripremamo, što dugo traje

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

Slika 2.3 Primer razgovora biznis analitičara i sponzora

Na slici 3 prikazan je dijalog biznis analitičara i sponzora projekta sproveden radi utvrđivanja poslovnih problema i ciljeva koji mogu da izgledaju slični. Ilustracija se odnosi na slučaj Chemical Tracking System projekta firme Contoso Pharmaceuticals. Na osnovu ovog razgovora, biznis analitičar je u stanju da kreira model poslovnih ciljeva projekta Chemical Tracking System , dat na slici 4.

Slika 2.4 Primer modela poslovnih ciljeva Chemical Tracking System projekta

## 1. POSLOVNI ZAHTEVI (ODELJAK 1.4 METRIKE USPEHA)

*Metrike uspeha pokazuju da li je projekat na putu da zadovolji svoje poslovne ciljeve.*

### 1.4 Metrike uspeha

Ovde treba da se specificiraju indikatori koje će agenci projekta da upotreba da bi merili uspeh projekta. Treba utvrditi faktore koji imaju najveći efekat na postizanje uspeh, pri čemu se uzimaju u obzir i faktori koji su pod kontrolom organizacije i faktori van nje.

Poslovni ciljevi, ponekad, ne mogu da se izmere pre nego što se projekat završi. U drugim slučajevima, postizanje poslovnih ciljeva može da zavisi od drugih projekata. Međutim, ipak je važno da se oceni uspeh projekta na kome radite i u ovakvim slučajevima. Metrike uspeha (success metrics) pokazuju da li je projekat na putu da zadovolji svoje poslovne ciljeve. Matrica se može pratiti za vreme testiranja ili ubrzo posle izdanja projekta.

Za slučaj projekta Chemical Tracking System, uspeh projekta se može pratiti preko ostvarenja poslovnog cilja 3 - "Smanjiti vreme naručivanja hemikalija na 10 minuta u 80% slučajeva naloga" jer se vreme obrade naloga može da meri za vreme testiranja ili odmah posle izdanja softvera.

Druga metrika uspeha se može povezati sa poslovnim ciljem 2 čije ostvarenje se može ustanoviti u roku mnogo kraćim od godinu dana od izdanja softvera, ako se poslovni cilj 2 meri sledećom metrikom uspeha : "Pratiti 60% komercijalnih hemijskih kontejnera i 50 % sopstvenih hemikalija unutar perioda od 4 nedelje".

## VAŽNO

Mudro izaberite vaše metrike uspeha. Budite sigurni da one mere ono što je zaista važno za vaš biznis, a ne da merite nešto samo zato što se može izmeriti. Metriku uspeha: "Smanjiti troškove razvoja proizvoda za 20%" je lako izmeriti. Može isto tako da bude lako i ostvariti taj cilj otpuštanjem ljudi iz razvoja ili uložiti manje u inovacije. Međutim, to bi moglo da ne budu željeni ciljevi projekta. Organizacija koja ne razvija nove i inovativne proizvode, pre ili kasnije će doći u krizu. Prema tome, to ne može da bude poslovni cilj projekta, te ni metrika za ocenjivanje tog cilja.

## 1. POSLOVNI ZAHTEVI (ODELJAK 1.5 ISKAZ O VIZIJI)

*Iskaz o viziji sumira dugoročnu svrhu i namenu proizvoda. On treba da odražava uravnotežen pogled koji će da zadovolji očekivanja različitih aktera.*

Napišite kratak iskaz o viziji koji sumira dugoročnu svrhu i proizvoda. Iskaz o viziji treba da odražava uravnotežen pogled koji će da zadovolji očekivana različitih aktera projekta. On može da izgleda idealistički, ali on mora da se oslanja na realnost postojećih tržišta, arhitekture organizacije, strateške smernice organizacije, i na ograničenja resursa. Sledeći uzorak ključnih reči može da se koristi za uspešnu formulaciju iskaza o viziji:

- **Za** (ciljni kupac)
- **Koji** (iskaz o potrebi ili prilici)
- **Naziv proizvoda**
- **Je** (kategorija proizvoda)
- **Koji** (glavne sposobnosti, ključna korist, neodoljiv razlog da se kupi ili upotrebi)
- **Za razliku od** (primarna konkurentna alternativa, sadašnji sistem, sadašnji poslovni proces),
- **Naš proizvod** (iskaz o primarnom razlikovanju i prednosti novog proizvoda).

Primenom ove formule za kreiranje iskaza o viziji, kreiran je iskaz o viziji projekta Chemical Tracking System:

**"Za naučnike koji imaju potrebu za nabavku kontejnera sa hemikalijama, **Chemical Trcking System** je informacioni sistem koji će obezbediti jednu tačku pristupa skladištu hemikalija i isporučiocima. Sistem će sadržati informaciju o lokaciji svakog kontejnera sa hemikalijama u kompaniji, o količini tog materijala koji je još u skladištu, i kompletну istoriju o lokaciji kontejnera i njegove upotrebe. Ovaj sistem će uštedeti troškove za nabavku hemikalija za 25% u prvoj godini korišćenja omogućavajući kompaniji da u potpunosti iskoristi hemikalije koje su raspoložive unutar kompanije, a rasprodati ili izbaciti manji broj delimično upotrebljenih kontejnera ili kontejnera čiji je rok upotrebe istekao, i upotrebije standardni sistem za naručivanje hemikalija. Za razliku od sadašnjih manuelnih procesa naručivanje hemikalija, **naš proizvod** će generisati sve izveštaje neophodne za zadovoljenje državne regulative koja zahteva izveštaje o upotrebi hemikalija, o njihovom skladištenju i otklanjanju."**

## PRIPREMA ISKAZA O VIZIJI

Možete zatražiti od nekoliko ključnih aktera projekta da vam napišu njihovu verziju iskaza o viziji. To će vam pomoći da bolje shvatite njihova očekivanja o projektu i pripremite vaš iskaz o viziji. Iskaz o viziji se može pripremati i paralelno sa radom na projektu, kako bi se dobio pravi i usaglašen iskaz o viziji, koji će usmeravati rad na projektu i držati ga u fokusu, tj. usmerenog na ostvarenje vizije.

# 1. POSLOVNI ZAHTEVI (ODELJCI 1.6, 1.7)

*Sumirajte glavne rizike projekta. Analizirajte realnost korišćenih prepostavki i mogućnosti prekida u zavisnosti u projektu.*

## 1.6 Poslovni rizici

Sumirajte glavne rizike koji su povezani sa razvojem, ili sa odsustvom razvoja, proizvoda, tj. softvera. Kategorije rizika obuhvataju: konkurenčiju na tržištu, rokovi, prihvaćanje rezultata od strane korisnika, pitanja implementacije, i mogući negativni uticaji na biznis. Poslovni rizici nisu sa rizicima projekta, koji obično uključuju i raspoloživost resursa, i tehnološke faktore. Procenite gubitak usled svakog od rizika, verovatnoću njegovog javljanja, i moguće akcije za ublažavanje rizika.

## 1.7 Poslovne prepostavke i zavisnosti

Prepostavka je iskaz za koji se veruje da je tačan u odsustvu dokaza ili konačnog znanja. Poslovne prepostavke (*business assumptions*) su specifično povezane sa poslovnim zahtevima. Tačne prepostavke mogu vas dovesti do neispunjavanja vaših poslovnih ciljeva. Na primer, neki izvršni sponzor može postaviti poslovni cilj da novi veb sajt poveća prihode za 100.000 dolara mesečno. Da bi se ostvario ovaj poslovni cilj, sponzor je postavio nekoliko prepostavki:

- veb sajt će privući još 200 novih posetioca sajta dnevno,
- svaki posetilac sajta će potrošiti u proseku 17 dolara

Ako se ove prepostavke ne ostvare, postavljen poslovni cilj se neće ostvariti.

Ako postanete svesni da su postavljenje prepostavke neostvarljive, možete da promenite opseg projekta, da promenite termin plan projekta, ili da otpočnete druge projekte koji mogu da ostvare ove ciljeve.

Zapišite svaku prepostavku na osnovu koje akteri projekta predlažu svoje verzije iskaza o viziji i okviru projekta. Ako ih analizirate, možete izbeći buduću moguću konfuziju i pogoršavanje u budućnosti.

Zapišite svaku veću zavisnost koju projekat ima i od spoljnog okruženja. Na primer, odlaganje industrijskog standarda, ili zakonske regulative, rezultati drugih projekata, isporuke dobavljača, i partneri razvoja. Neke prepostavke mogu dovesti do rizika, te ih menadžer

proizvoda mora dnevno pratiti. Prekinute zavisnosti su česti uzrok kašnjenja projekata. Važno je da akteri projekta shvate važnost nerealnih prepostavki ili prekinutih zavisnosti.

## 2. OKVIR I OGRANIČENJA (POGLAVLJA 2.1 I 2.2)

*Okvir se definiše nizom funkcionalnih zahteva koji se planiraju za izvršenje (implementaciju) u okviru planiranog izdanja ili iteracije.*

Pri razvoju softvera treba definisati okvir i ograničenja projekta razvoja. Šta je softver, a šta nije. Mnogi projekti imaju problem zbog stalnog širenja okvira projekta, kako se dodaju nove i neplanirane funkcije. Zato je važno da se na početku definiše okvir celog projekta.

Okvir projekta (project scope) opisuje koncept i rang predloženog rešenja. Ograničenja (limitations) konkretizuju određene karakteristike softvera koje on ne sme da ima iako bi neki ljudi očekivali da ih imaju. Okvir i ograničenja pomažu da projektni akteri imaju realistična očekivanja jer ponekad, kupci zahtevaju svojstva čija realizacija može biti preskupa ili koja su van nameravanog okvira projekta.

Okvir projekta se može predstaviti na različite načine. Na najvišem nivou apstrakcije, ograničenje se definiše kada kupac odlučuje o ciljnim poslovnim ciljevima. Na nižim nivoima apstrakcije, ograničenje se definiše na nivou svojstava, priča korisnika, slučajeva korišćenja, ili događaja i odgovora koje treba uključiti. Okvir se definiše nizom funkcionalnih zahteva koji se planiraju za izvršenje (implementaciju) u okviru planiranog izdanja ili iteracije. U svakom nivou, okvir mora da bude u određenim granicama na višim nivoima. Na primer, zahtevi korisnika u granicama okvira se moraju preslikati u poslovne ciljeve, a funkcionalni zahtevi moraju da odražavaju zahteve korisnika koji su unutar definisanog okvira.

### 2.1 Glavna svojstva

Lista glavnih svojstava proizvoda ili karakteristika korisnika, razlikuje ga od prethodnih izdanja ili od konkurentnih proizvoda. Ova lista mora da bude kompletна, ali izbegnite da sadrži svojstva nepotrebna svojstva, koja mogu delovati atraktivno, ali korisniku ne obezbeđuje neku dodatnu vrednost. Dajte svakom glavnom svojstvu jedinstvenu oznaku kako bi ga pratili u drugim elementima sistema. Možete koristiti stablo svojstava, koje ćemo opisati.

### 2.2 Okvir početnog izdanja

Objedinite sva svojstva koja planirate da uključite u početno izdanje softvera. Okvir se često definiše u terminima svojstava, ali se može definisati sa korisničkima pričama, slučajeva korišćenja, toka aktivnosti korisničkih svojstava ili spoljnih događaja koje zadovoljavaju. Takođe opišite karakteristike kvaliteta koje će doprineti da proizvod obezbedi nameravane koristi različitim svojim korisnicima. Da bi što pre dobili početno izdanje (u skladu sa planom projekta), vodite računa da ne preterate se brojem svojstava početnog izdanja. Usredsredite se na svojstva koja će najviše povećati vrednost proizvoda, sa što manjim troškovima, što većoj zajednici korisnika, i u što kraće vreme.

## 2. OKVIR I OGRANIČENJA (POGLAVLJA 2.3 I 2.4)

*Potrebno je da napravite plan funkcionalnosti koje ćete razviti u svakoj fazi, iteraciji ili inkrementu.*

### 2.3 Okvir narednih izdanja

Ukoliko planirate fazni razvoj softvera, ili primenu iterativnog ili inkrementalnog razvoja softvera, potrebno je da napravite plan funkcionalnosti koje ćete razviti u svakoj fazi, iteraciji ili inkrementu. Naredna izdanja imaju svoje dodatne slučajeve korišćenja i svojstava, i time obogaćuju osobenosti početnog izdanja. Kasnija izdanja se manje precizno i jasno planiraju, te će se u budućnosti dopunjavati. Kraći vremenski intervali između izdanja softvera omogućavaju bržu reakciju na primedbe korisnika.

### 2.4 Ograničenja i isključenja

Dajte listu svih svojstava proizvoda koje korisnik može da očekuje ali koji NISU planirani da budu uključeni u određenom izdanju. Navedite svojstva koja su u okviru projekta, ali koja su isključena iz okvira određenog izdanja. Time se daje informacija da ta svojstva nisu zaboravljena i da će kasnije biti uključena u okvir nekih budućih izdanja.

## 3. POSLOVNI KONTEKST (POGLAVLJA 3.1 I 3.2)

*Akteri projekta su pojedinci, grupe i organizacije koji su aktivno uključeni u projekat, pod uticajem su njegovih rezultata ili mogu da utiču na rezultate.*

**3.1 Profili aktera projekta** Akteri projekta (stakeholders) su pojedinci, grupe i organizacije koji su aktivno uključeni u projekat, pod uticajem su njegovih rezultata ili mogu da utiču na rezultate. Ovde ne morate da opištete svakog aktera, već prevenstveno se treba usredosrediti na različite tipove kupaca, ciljne segmente tržišta, i na razne klase korisnika unutar ovih segmenata.

Za svaki profil aktera projekta treba obezbediti sledeće informacije:

- Glavnu vrednost koju će akter steći od proizvoda (softvera, sistema) koji se razvija, pri čemu se vrednost definiše preko:
  - povećanja produktivnosti
  - smanjenja ponovnog rada zbog grešaka u softveru,
  - smanjivanja troškova,
  - poboljšanja poslovnih procesa,
  - automatizacije manuelnih procesa,
  - sposobnosti da izvrši potpuno nove zadatke,
  - usaglašenja sa važećim standardima i propisima,
  - poboljšanje upotrebljivosti u poređenju sa sadašnjim proizvodom.
- Njihove stavove prema proizvodu:
- Glavna svojstva i karakteristike koje ih interesuju

- Neko ograničenje koje se mora ugraditi u sistem.

Možete dati listu ključnih aktera sa imenima za svaki profil aktera ili da date organizacionu strukturu koja pokazuje veze među akterima u organizaciji.

**3.2 Prioriteti projekta** Da bi se ostvarilo efektivno donošenje odluka, akteri moraju da se dogovore oko prioriteta projekta. U tom cilju se oni mogu odrediti u odnosu na pet dimenzija: svojstva, kvalitet, rokova, troškova, i članova tima. Svaka od ovih dimenzija se nalazi u okviru sledeće tri kategorije:

- **Ograničenje (constraint)**: Faktor ograničenja o kom menadžer projekta mora da vodi računa.
- **Pokretač (driver)**: Značajan cilj uspeha sa ograničenom fleksibilnošću podešavanja.
- **Stepen slobode (degree of freedom)**: Faktor koji menadžer projekta može do izvesne mere da podešava u odnosu na druge dimenzije.

Izazov je za menadžera projekta da podesi stepene slobode tako da se postignu pokretači uspeha projekta unutar granica uspostavljenih ograničenjima.

Prilikom zahteva za promenom u projektu, menadžer projekta mora da, posle konsultacija sa ključnim akterima, odredi akcije koje najviše odgovaraju za slične zahtevane promene.

## POSLOVNI KONTEKST (POGLAVLJE 3.3)

*Ovde se daju informacije i navode aktivnosti koje su neophodne za obezbeđenje efektivnog raspoređivanja i primene novog rešenja u operativnom okruženju.*

### 3.3 Razmatranja raspoređivanja

Sumirajte informacije i aktivnosti koje su neophodne da bi se obezbedilo efektivno raspoređivanje rešenja u operativnom okruženju. Opišite pristup koji će korisnici zahtevati radi upotrebe sistema, kao na primer, da li se korisnici nalaze u različitim vremenskim zonama ili se nalazi blizu jedan drugog. Navedite kada korisnici koji su na različitim lokacijama žele da pristupe sistemu.

Ako su potrebne promene infrastrukture da bi odgovorile na zahteva softvera koji se tiču kapaciteta, pristupa mreži, skladištenja podataka, ili u vezi migracije podataka, treba opisati te promene.

Zapišite bilo koju informaciju koja će biti potrebna onima koji treba da pripreme obuku ljudi ili koji treba da vrše promene u poslovnim procesima u vezi sa raspoređivanjem novog rešenja.

## ▼ Poglavlje 3

# Tehnike određivanja okvira

## VIDEO PREDAVANJE ZA OBJEKAT "

*Trajanje video snimka: 5min 17sek*

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

## DIJAGRAM KONTEKSTA

*Dijagram konteksa vizuelno ilustruje granicu između sistema koji se razvija i svega ostalog u univerzumu. Utvrđuje i terminatore, kao interfejse sistema.*

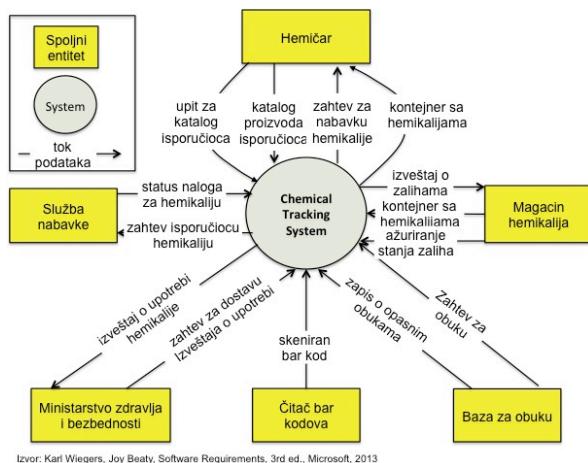
Tehnike, ili alati koji se koriste pri određivanju okvira projekta omogućuju jasniju i precizniju komunikaciju između aktera projekta.

### 1. Dijagram konteksta

Opis okvira postavlja granicu i veze između sistema koji se razvija i svega ostalog u univerzumu. **Dijagram konteksa** (contex diagram) vizuelno ilustruje tu granicu. On utvrđuje tzv. spoljne entitete (ili terminatore) van granica sistema koji su interfejs (sprega) sa sistemom , kao i podatke, kontrolu, i tok materijala između terminadora i sistema.

Slika 1 prikazuje deo dijagrama konteksta za projekat Chemical Tracking System. **Ceo sistem je predstavljen jednim krugom.** Očigledno,dijagram konteksta ne daje nikakvu vidljivost unutrašnjosti sistema, koju čine objekti, procesi ili podaci u sistemu. Sistem predstavljen krugom može da predstavlja bilo koju kombinaciju softvera, hardvera, i ljudskih komponenata. Prema tome, on uključuje i manuelne operacije unutar sistema.

**Spoljni entiteti** ( external entities) mogu da predstavljaju klase korisnika (Hemičar, Kupac), organizacije (Departman zdravlja i bezbednosti), druge sisteme (baza za obuku), hardverske uređaje (čitači bar kodova). Strelice predstavljaju tokove podataka (npr. zahtev za hemikalijom) ili fizičke stvari (kao što je kontejner hemikalija) između sistema i spoljnih entiteta (terminatora).



Slika 3.1 Deo dijagrama konteksta projekta Chemical Tracking System

Proizvođač hemikalija nije prikazan na slici jer nije deo ovog projekta. Naručivanje i plaćanje robe i rad sa proizvođačima se radi u okviru drugog sistema u kompaniji i te poslove obavlja Služba nabavke.

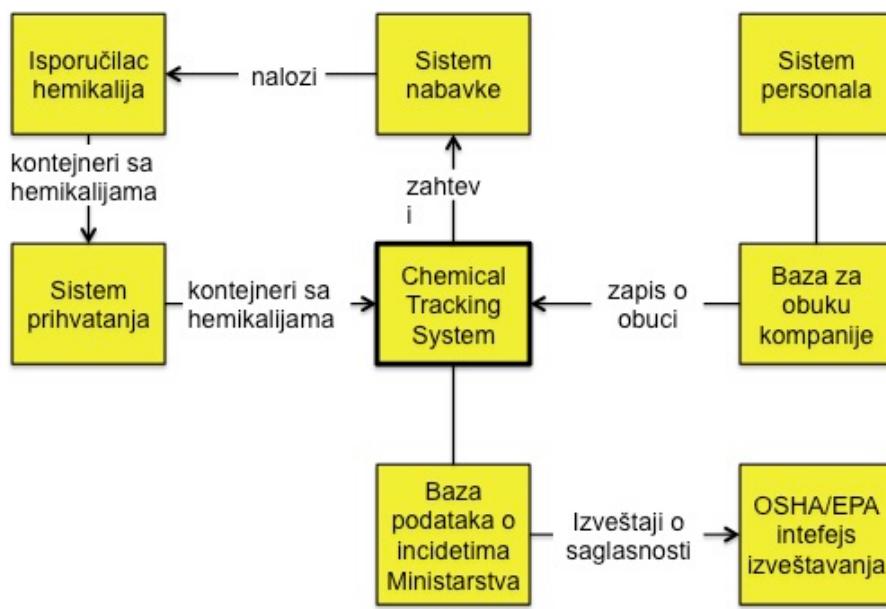
## MAPA EKOSISTEMA

*Mapa ekosistema pokazuje sve sisteme povezane sa vašim sistemom koji su u međusobnoj interakciji i prikazuju prirodu ovih interakcija.*

**Mapa ekosistema** ([ecosystem map](#)) pokazuje sve sisteme povezane sa sistemom od interesa koji su u međusobnoj interakciji i prikazuju prirodu ovih interakcija. Mapa sistema predstavlja okvir prikazujući sve povezane sisteme. Za razliku od dijagrama konteksta, mape ekosistema prikazuju i druge sisteme koji su povezani sa sistemom koji se razvija, uključujući i sisteme sa kojima on nema direktnе interfejsе. Možete utvrditi eksterne sistema prateći podatke, tj. uključujući sisteme koje koriste podatke sistema koji se razvija. Kada dođete do sistema koji ne koristi podatke vašeg sistema, to znači da ste došli do granice vašeg sistema.

Slika 2 pokazuje mapu ekosistema projekta Chemical Tracking System. Strelice sa oznakama pokazuju tok podataka između sistema. Neki od ovih tokova mogu se pojaviti i u dijagramu konteksta.

Ekosistem prikazan na slici 2 pokazuje da Chemical Tracking System nije u direktnoj vezi sa OSHA/EPA sistemom za izveštavanje. Treba da razmotrite da li se neki od zahteva Chemical Tracking System-a javlja zbog podataka koji teku od njega, preko Baze podataka o incidentima Ministarstva zdravlja i bezbednosti i vodi do interfejsa za izveštavanje.



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

Slika 3.2 Deo ekomape sistema Chemical Tracking System

## STABLO SVOJSTAVA

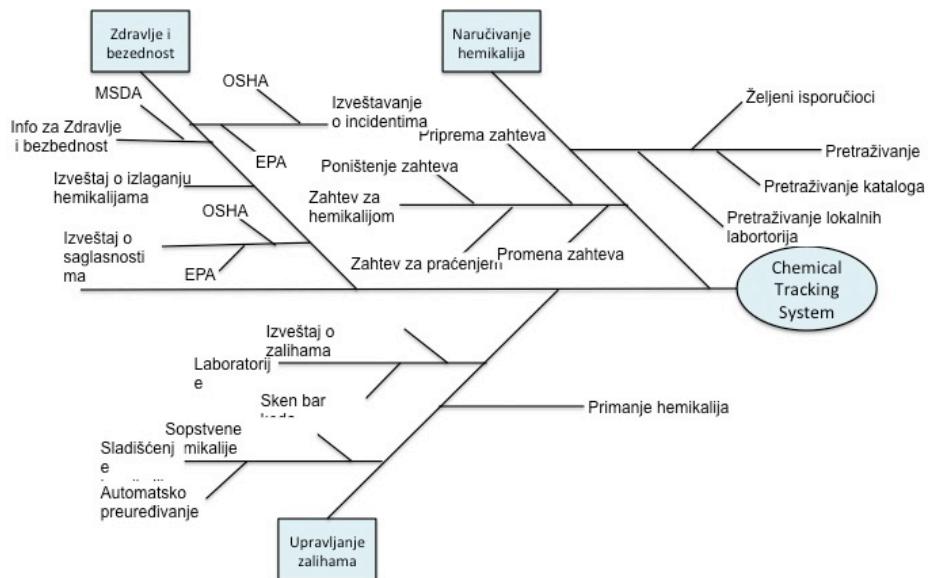
*Stablo svojstava prikazuje logično grupisana svojstva sistema, hijerarhijski podeljena na detaljnije nivoe prikazivanja.*

Stablo svojstava (feature tree) prikazuje logično grupisana svojstva sistema, hijerarhijski podeljena na detaljnije nivoe prikazivanja. Stablo svojstava obezbeđuje koncizan pogled na sva svojstva koja su planirana za projekat. Idealan je za prikaz direktorima koji žele kratak prikaz projekta. Stablo svojstva može da prikaže tri nivoa detaljisanja svojstava, koji se označavaju sa Nivo 1, Nivo 2 i Nivo 3, tj. N1, N2 i N3.

Slika 3 prikazuje delimično stablo projekta Chemical Tracking System. Glavna grana na sredini predstavlja proizvod (Chemical Tracking System). Svako svojstvo ima svoju granu. Pravougaonici označavaju N1 svojstva. Linije koje polaze od N1 grane su N2 svojstva. I konačno, grane koje polaze od N2 grana, predstavljaju N3 svojstva.

Kada planirate neko izdanje ili iteraciju, možete definisati njegov okvir izborom skupa svojstava koje treba izvršiti. Svojstva se biraju u celina, a u slučaju svojstava N2 i N3 moguće ih je izabrati kao deo svojstava glavne grane.

Ako se koriste boje, može se prikazati stablo svojstava svih izdanja ili iteracija, pri čemu jedna boja odgovara jednom izdanju ili iteraciji.



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

Slika 3.3 Deo dijagrama stabla svojstava za slučaj projekta Chemical Tracing System

## LISTA DOGAĐAJA

*Lista događaja utvrđuje spoljne događaje koji mogu da pokrenu neko ponašanje sistema.*

**Lista događaja** (**event list**) utvrđuje spoljnje događaje koji mogu da pokrenu neko ponašanje sistema. Spoljni događaji kao što su: vremenski događaji, signalni događaji, i dr koji dolaze spolja, mogu da pokrenu razna ponašanja sistema. Lista događaja samo imenuje događaje. Funkcionalni zahtevi definišu koje se akcije u sistemu pokreću kada se javi takvi događaji. Njihov opis se daje u dokumentu softverskim zahtevima (SRS) o kojima će u drugim predavanjima biti reći.

Slika 4 daje delimičnu listu spoljnih događaja sistema Chemical Tracking System. Za svaki događaj, pored njegovog naziva, daju se trigeri (pokretači) odgovarajućih akcija, kao i opisi ovih akcija. Lista događaja je alat za određivanje okvira projekta, a na osnovu nje možete utvrditi listu događaja i ona daje zadatak koje događaje treba evidentirati u sistemu za svako izdanje sistema ili njegovu iteraciju.

Lista događaja je komplementarni alat dijagramu konteksta i mapi ekosistema. Dijagram konteksta i mapa ekosistema zajedno opisuju spoljne aktere i sisteme koji su uključeni, dok lista događaja utvrđuje šta može da pokrene ponašanja ovih aktera

Možete proveriti tačnost i kompletност listu događaja u odnosu na dijagram konteksta i mapu ekosistema na sledeći način:

Spoljni događaji za Chemical Tracking System

- Hemičar postavlja zahtev za određenu hemikaliju.
- Bar kod kontejnera sa hemikalijama je skeniran.
- Vreme potrebno za generisanje dobijanje izveštaja saglasnosti OSHA.
- Isporučilac hemikalija objavljuje novi katalog hemikalija.
- Nova sopstvena hemikalija je ubaćena u sistem.
- Isporučilac opreme javlja da je poništen nalog za naručenu hemikaliju.
- Hemičar pita da generiše svoj izveštaj o izloženosti dejstvu hemikalije.
- Primljen je bezbednosti izveštaj o obnovljenom materijalu od EPA.
- Dodat je novi isporučilac opreme na listu isporučioca hemikalija.
- Kontejner sa hemikalijama je isporučen od isporučioca.

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

Slika 3.4 Delimična lista događaja za slučaj projekta Chemical Tracking System

- Vidite da li je svaki spoljni entitet dijagrama konteksta izvor nekog događaja. Na primer "Da li neka akcija Hemičara može da pokaže neko ponašanje sistema?"
- Vidite da li neki sistem u mapi ekosistema dovodi do nekog događaja u sistemu.
- Proverite da li za svaki događaj imate odgovarajuće spoljne entitete u dijagramu konteksta ili sisteme u mapi ekosistema.

Ako pronađete negde prekid, proverite da li u modelu nedostaje neki element. U slučaju Chemical Tracking System-a, Isporučilac hemikalija se ne pojavljuje u dijagramu konteksta jer sistem nema zajednički interfejs sa isporučiocem, što se vidi u mapi ekosistema. Mapa ekosistema pokazuje da Isporučilac opreme nije uključen u mapu.

## ▼ Poglavlje 4

### Držanje okvira u fokusu

#### KAKO ODREDITI PRIORITETE PRI REALIZACIJI ZAHTEVA

*Dokument o viziji i okviru vam pomaže da ocenite predložene nove zahteve. Poslovni ciljevi su najvažniji faktor koji se uzima u obzir prilikom odlučivanja*

Dokument o viziji i okviri vam pomaže da ocenite predložene zahteve i da vidite da li su odgovarajući za projekat, tj. da li su u skladu sa vizijom i okvirom projekta. Međutim, i dokument o viziji i okviru može da se koriguje ako za to postoje jaki razlozi koji vode projekt ka uspehu.

#### **Upotreba poslovnih ciljeva pri donošenju *odлуka o okviru***

Poslovni ciljevi su najvažniji faktor koji se uzima u obzir prilikom odlučivanja o okviru. Treba da odredite koja svojstva ili zahtevi korisnika dodaju najviše vrednosti u odnosu na poslovne ciljeve. Njih treba primeniti u ranijim fazama projekta. Kada neki od aktera želi da doda neku funkcionalnost, razmotrite kako će ta promena uticati na postizanje poslovnih ciljeva. Na primer, u slučaju kioska u trgovinama, ako se želi da ostvari što veći prihod, prednost treba dati prodaji robe koja dovodi do najvećeg broja kupovina, tj. dovodi najveći broj kupaca.

Ako je moguće, kvantifikujte doprinos nekog svojstva postizanju poslovnog cilja, kako bi pri odlučivanju o prioritetima odlučivale činjenice, a ne emocije.

#### **Ocenjivane posledica promena okvira**

Kada se okvir projekta povećava, menadžer projekta obično mora da ponovo dogovori promene u budžetu, ili u planu (rokovima) ili u resursima (ljudima), kako bi mogao da realizuje usvojenu promenu, ukoliko nije unapred ostavljeno prostora za ovakve promene.

Česta posledica promena okvira je nužnost da se već urađene aktivnosti moraju ponovo uraditi ili doraditi, da bi se odgovorilo na promene. Ako se ne povećaju resursi ili vreme pri dodavanju nove funkcionalnosti, onda dolazi do pada kvaliteta rešenja. Dokumentovani poslovni zahtevi olakšavaju upravljanje promenama okvira, koje obično prate promene na tržištu ili promene u poslovnim potrebama. U slučaju dokumentovanih poslovnih zahteva, lakše je menadžeru projekta da se odupre pritiscima i uticaju aktera koji "guraju" neke promene i nove zahteve, jer menadžer projekta lakše može da argumentuje svoje odbijanje ili odlaganje.

## ▼ Poglavlje 5

# Upotreba poslovnih ciljeva radi određivanja kraja

## KADA JE PROJEKAT ZAVRŠEN?

*Urađene iteracije dovode do realizacije vizije proizvoda koji zadovoljava poslovne ciljeve. Neophodno je poslovne ciljeve jasno definisati, da bi znali*

Kada znate da ste završili sa izvršenjem (implementacijom) funkcionalnosti softvera? Tradicionalno, menadžer projekta upravlja projektom do njegovog završetka. Međutim, kako biznis analitičar dobro poznaje postavljene poslovne ciljeve, on može da pomogne u oceni da li su ostvarene željene vrednosti i da je posao završen.

Ako počnete projekat sa jasnom vizijom rešenja, i ako je svako izdanje, ili iteracija, u okviru definisanog okvira i daje deo ukupne funkcionalnosti, onda ste završili projekat kada ste realizovali planirane iteracije. Urađene iteracije dovode do realizacije vizije proizvoda koji zadovoljava poslovne ciljeve.

Međutim, specifičnost iterativnog razvoja je u tome što završna tačka nije precizno definisana. U svakoj iteraciji se određuje novi okvir softvera, odnosno, vrši se njegovo proširenje. Kako se projekat približava kraju, broj nerešenih zahteva polako nestaje. Nije uvek potrebno da se realizuju ceo set preostale funkcionalnosti. Vrlo je važno da imate jasne poslovne ciljeve tako da projekat inkrementalno usmeravate ka njihovom ostvarenju kada informacije postanu raspoložive. Projekat je završen kada metrike uspeha pokazuju da ste ostvarili poslovne ciljeve. **Ako poslovni ciljevi nisu precizno definisani, onda će i kraj projekta biti neprecizno definisan.**

Sponzori koji finansiraju ove projekte ovo ne vole, jer ne znaju kako da obezbede budžet, rokove ili plan takvih projekata. I kupci ne vole ovakve projekte, iako su u okviru budžeta i planiranih troškova, jer se može desiti da ne ostvaruju njihova očekivanja.

**Zato je potrebno da se usmerite ka definisanu jasnih poslovnih zahteva za vaše projekte, da bi znali da li ste uspešno završili posao.**

Evo primera nedovoljno definisanog cilja projekta koji je iniciran da bi se razvio softver koji treba da pomogne menadžmentu pri upravljanju troškovima poslovanja firme. :

- Cilj: Pored direktnih troškova, projekat treba da obuhvati i sve druge posredne troškove i uticajne faktore na troškove poslovanja.

Šta je ovde problem? Cilj nije precizan. Do kog nivoa troškova treba pratiti? U kojim oblastima pratiti, a u kojima nije potrebno? Računovodstveni sistemi prate sve troškove, ali najčešće

njihova struktura izveštaja nije u formi pogodnoj za odlučivanje menadžera. Za one koji odlučuju o bitnim pitanjima, treba dati samo bitne i relevantne troškove koji su vezani za odluke koje oni reba da donose.

## VIDEO 3 - THE BUSINESS VALUE OF BETTER REQUIREMENTS - WIEGERS (VIDEO)

*Trajanje: 7:09 minuta*

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

## VIDEO 7 - THE VISION STATEMENT - WIGERS (VIDEO)

*Trajanje: 2:36 minuta*

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

## VIDEO 8 - DEPICTING PROJECT SCOPE AND THE CONTEXT DIAGRAM (VIDEO)

*Trajanje: 8:14 minuta*

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

## ▼ Poglavlje 6

### Vežba

## POSLOVNI ZAHTEVI INFORMACIONOG SISTEMA PRIVATNE KLINIKE

*Primer sadržaja Dokumenta o viziji i okviru: Poslovni zahtevi, Pozadina, Poslovna prilika*

### **1. Poslovni zahtevi**

U ovom dokumentu su razmotreni zahtevi poslovanja, na osnovu kojih se pokreće projekat razvoja poslovnog informacionog sistema privatne klinike. Poslovni zahtevi su dobijeni od rukovodstva kupca, privatne klinike Hipokrat, i predstavljaju osnovu za razvoj detaljnih zahteva.

#### **1.1. Pozadina**

Privatna klinika Hipokrat poseduje poslovnu aplikaciju koju želi da zameni novim softverom.

#### **1.2. Poslovna prilika**

Poslovni informacioni sistem privatne klinike, čija se izgradnja planira, treba da obezbedi veću kontrolu poslovanja i da podrži automatizovano obavljanje važnih poslovnih procesa. Staru aplikaciju je neophodno zameniti jer ne podržava u potpunosti potrebe poslovanja, s obzirom na to da se obim posla klinike povećao od otvaranja, a zahtevi RFZO su porasli. Problemi koje treba rešiti novim sistemom su sledeći:

1. Evidencija podataka o pacijentima obavlja se delimično ručno, a delimično kroz poslovnu aplikaciju
2. Evidencija podataka o zaposlenima obavlja se delimično ručno, a delimično kroz poslovnu aplikaciju
3. Zakazivanje pregleda obavlja se ručno
4. Vođenje kartona pacijenta obavlja se delimično ručno, a delimično u papirnoj formi
5. Kreiranje i razmena rasporeda rada, pregleda i šifrarnika je putem mejla
6. Vođenje finansija obavlja se kroz elektronsku dokumentaciju
7. Vođenje robno-materijalnog knjigovodstva obavlja se kroz elektronsku dokumentaciju
8. Nema preciznih statističkih, finansijskih izveštaja i izveštaja za donošenje odluka, neki od izveštaja se kreiraju objedinjavanjem podataka iz postojeće aplikacije, elektronske i ručne dokumentacije
9. Ne postoji integracija sa RZFO servisima, što je važno zbog automatskog pristupa šifrarnicima, fakturama i provere osiguranja

# POSLOVNI ZAHTEVI INFORMACIONOG SISTEMA PRIVATNE KLINIKE - NASTAVAK

*Primer sadržaja Dokumenta o viziji i okviru: Poslovni ciljevi*

## 1.3. Poslovni ciljevi

Finansijski	Nefinansijski
Smanjiti potrošnju papira i rashode na papirna sredstva za 30% u prvih mesec dana korišćenja novog sistema.	Smanjiti vremensko trajanje procesa zakazivanja jednog pregleda za 50% u prvih 6 meseci korišćenja novog sistema.
Smanjiti operativne troškove za minimum 15% u prvih 6 meseci korišćenja novog sistema.	Minimizirati čekanje pacijenata koji su rezervisali preglede unapred. Čekanje na pregled duže od 15 minuta sme da se dogodi u svega 2% vremena.
Dobiti sistem izveštavanja o finansijskom stanju, odnosno alat za osmišljavanje načina za uvećanje prihoda od minimum 7% u prvih godinu dana korišćenja novog sistema.	Povećati zadovoljstvo pacijenata smanjenjem ukupnog vremena provedenog na klinici u proseku za 10 minuta.
Sprovesti siguran i pouzdan sistem naplate	Povećati zadovoljstvo medicinskih radnika zbog prestanka vođenja evidencije duplo. Povećanje zadovoljstva administrativnih radnika zbog lakše komunikacije sa menadžmentom i veće kontrole rada.

Slika 6.1 Tabela poslovnih ciljeva

# POSLOVNI ZAHTEVI INFORMACIONOG SISTEMA PRIVATNE KLINIKE - IZJAVE O VIZIJI

*Primer sadržaja Dokumenta o viziji i okviru: Izjave o viziji (medicinski radnik i lekar)*

## 1.4. Izjave o viziji

Za medicinske radnike, **koji** imaju potrebu da evidentiraju pacijente, zakazuju preglede i naplaćuju obavljene usluge, **Poslovni sistem** klinike je informacioni sistem **koji** predstavlja jedinstvenu tačku pristupa podacima o pacijentima i trenutnim cenovnicima. Sistem treba da obezbedi čuvanje informacija o svakom pacijentu koji se ikada lečio na klinici, kao i pregled istorije poseta. Poslovni sistem ima za cilj da smanji vreme koje se utroši na zakazivanje pregleda za 50%, već u prvih 6 meseci korišćenja. Izdavanje računa i plaćanje karticama treba da bude podržano kroz naplatu usluga prema definisanom cenovniku u okviru sistema. Za razliku od trenutne aplikacije za evidenciju, koja nema mogućnost čuvanja svih neophodnih podataka o pacijentima, pa se deo evidencije vodi ručno, **poslovni sistem klinike** će omogućiti brže zakazivanje u skladu sa preciznim rasporedom slobodnih

lekara i termina. Izdavanje računa se trenutno vrši direktno preko fiskalne kase, dok se od novog sistema očekuje da obezbedi siguran i integriran sistem naplate.

**Za** lekare, **koji** imaju zadatak da evidentiraju dijagnoze i obavljene specijalističke pregledе, kao i da izdaju upute za pregledе drugih lekara u okviru klinike, **Poslovni sistem** klinike **je** informacioni sistem **koji** im daje uvid u elektronski karton svakog pojedinačnog pacijenta i mogućnost ažuriranja podataka koji se u njemu čuvaju. Elektronski karton treba da bude privatno skladište informacija o pacijentu, uključujući faktore rizika, prethodna oboljenja, dijagnoze i svu prateću medicinsku dokumentaciju. Elektronski karton svakog pacijenta treba da bude zaštićen od neovlašćenog pristupa i treba da omogući smanjenje potrošnje papira minimum za trećinu u odnosu na čuvanje papirnih kartona i fascikli. **Za razliku od** trenutnog decentralizovanog vođenju rasporeda pregledа, koji se prosleđuje zaposlenima putem mejla, što dovodi do čestih gužvi i preklapa termina, **poslovni sistem klinike** će obezbediti tačnost i pravovremenost kada je reč o pristupu informacijama o značaju.

## POSLOVNI ZAHTEVI INFORMACIONOG SISTEMA PRIVATNE KLINIKE - IZJAVE O VIZIJI - NASTAVAK

*Primer sadržaja Dokumenta o viziji i okviru: Izjave o viziji  
(administrativni radnik i menadžer)*

**Za** administrativne radnike, **koji** imaju zadatak da vode kadrovsku evidenciju, definišu radne periode i rasporede pregledа, kreiraju šifrarnike i vode robno-materijalno knjigovodstvo za potrebe klinike, **Poslovni sistem** klinike je informacioni sistem **koji** im omogućava da obavljaju navedeni skup administrativnih poslova. Vođenje kadrovske evidencije za administrativne radnike predstavlja upravljanje ljudskim resursima i informacijama o njihovim zaposlenjem. Administrativnim radnicima je zadatak i da vode računa o stanju materijalnih sredstava za rad i njihovom pravovremenom poručivanju od dobavljača. **Za razliku od** trenutnog toka informacija od značaja, koji je podložan velikim propustima jer se sprovodi putem mejla, gde se deo informacija prikuplja iz poslovne aplikacije, a deo vodi ručno u Excel fajlovima; kreiranje šifrarnika, rasporeda i praćenje stanja kroz **sistem** će omogućiti da relevantne informacije budu dostupne lekarima, medicinskim radnicima i menadžmentu u bilo kom trenutku. Očekuje se da novi informacioni sistem smanji korišćenje mejla za 60% i podigne nivo informisanosti zaposlenih.

**Za** menadžere, **koji** imaju potrebu za statističkim izveštajima i praćenjem finansijskog stanja klinike, **Poslovni sistem** klinike **je** informacioni sistem **koji** im omogućava da dobiju podatke koji govore o poslovanju klinike tako da mogu po potrebi da reaguju i kreiraju planove za dalji razvoj preduzeća. Njima je jako važno da sistem ispoštuje standarde propisane od strane Republičkog fonda za zdravstveno osiguranje (RFZO) i da koristi njihove servise. **Za razliku od** trenutne kontrole prihoda i rashoda, koja se obavlja ručno posredstvom Excel fajlova i podložna je greškama visokog rizika, **poslovni sistem klinike** će obezbediti strožu kontrolu finansija, a široki spektar izveštaja će omogućiti analizu podataka za kreiranje uspešnih poslovnih strategija. Menadžment privatne klinike ima u planu povećanje prihoda minimum za 5% u toku prve godine korišćenja poslovnog sistema i smanjenje celokupnih rashoda minimum za 15%.

# POSLOVNI ZAHTEVI INFORMACIONOG SISTEMA PRIVATNE KLINIKE - POSLOVNI RIZICI

*Primer sadržaja Dokumenta o viziji i okviru: Poslovni rizici, Poslovne pretpostavke i zavisnosti*

## 1.5. Poslovni rizici

Ako se novi proizvod ne pusti u operativan rad u roku od godinu dana od početka prikupljanja poslovnih zahteva, postoje sledeći rizici:

RI-1: Privatna klinika će pretrpeti poslovne gubitke zbog toga što nije na vreme ugradila poslovni sistem koji će automatizovati poslovne procese i omogućiti bolje upravljanje poslovanjem.

RI-2: Privatna klinika će pretrpeti poslovne gubitke zbog toga što nije u korak sa konkurentima koji već koriste benefite ugradnje informacionih tehnologija u poslovanje.

RI-3: Privatna klinika će morati da poveća broj ljudskih resursa, što će dovesti do finansijskih gubitaka, jer rashodi ne mogu biti smanjeni bez uvođenja novog sistema.

RI-4: Rast poseta u privatnoj klinici može stati ili broj poseta može opasti, ukoliko klijenti postanu nezadovoljni zbog brzine rada ili problema do kojih dolazi zbog decentralizovanog vođenja podataka.

Rizici puštanja novog proizvoda:

RI-5: Izvestan broj zaposlenih će pokazati otpor prema novom sistemu tokom perioda privikavanja.

RI-6: Izvesnom broju starijih radnika će biti potrebna duža obuka za korišćenje.

## 1.6. Poslovne pretpostavke i zavisnosti

AS-1: Svi podaci koji su do sada u nekoj formi čuvani u poslovnoj aplikaciji, biće preneti u novi poslovni sistem.

DE-1: RFZO može promeniti svoje servise ili pravila u toku razvoja poslovног sistema, koji se mora prilagoditi novonastalim promenama.

# POSLOVNI ZAHTEVI INFORMACIONOG SISTEMA PRIVATNE KLINIKE - OBIM I OGRANIČENJA

*Primer sadržaja Dokumenta o viziji i okviru: Obim i ograničenja*

## 2. Obim i ograničenja

Definisani obim predstavlja referentni okvir na osnovu koga se procenjuju korisnički i softverski zahtevi. Zahtevi koji nisu obuhvaćeni predviđenim proizvodom moraju biti odbijeni, osim ako nisu toliko korisni da se razmotri povećanje obima radi njihovog uključenja.

### 2.1. Glavne karakteristike

FE-1: Evidenciranje pacijenata - kreiranje elektronskog kartona

FE-2: Zakazivanje pregleda kod lekara

FE-3: Naplata usluga

FE4: Evidenciranje dijagnoza, terapija, evidentiranje specijalističkih pregleda (bolovanja,

pregledi medicine rada, lekarska uverenja), medicinske dokumentacije, uputa za druge preglede na klinici; sve kroz elektronski karton pacijenta

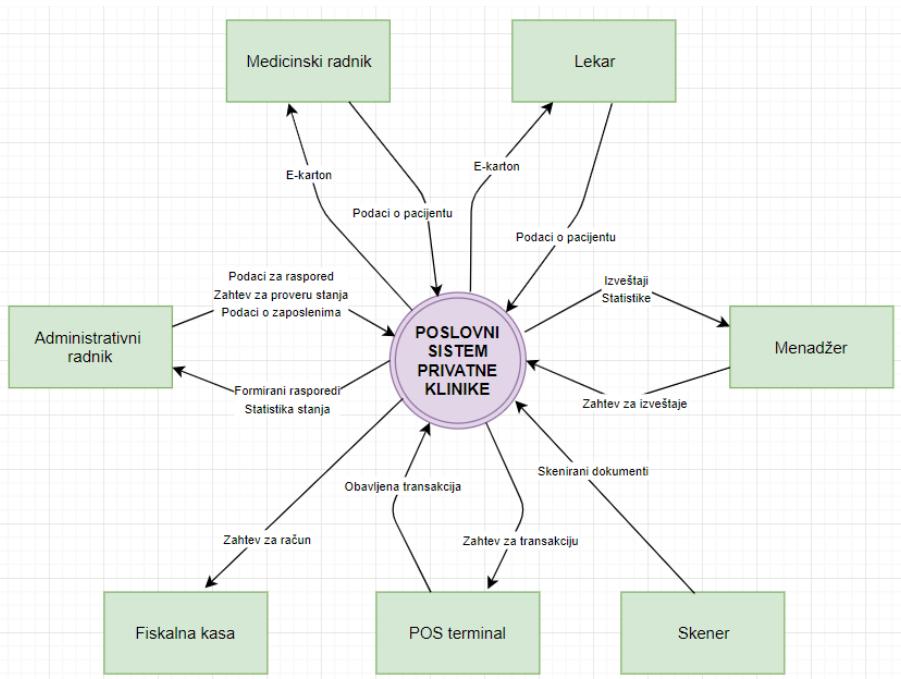
FE-5: Administracija rada klinike

FE-6: Robno-materijalno knjigovodstvo

FE-7: Upravljanje ljudskim resursima

FE-8: Upravljanje finansijama

FE-9: Izveštavanje

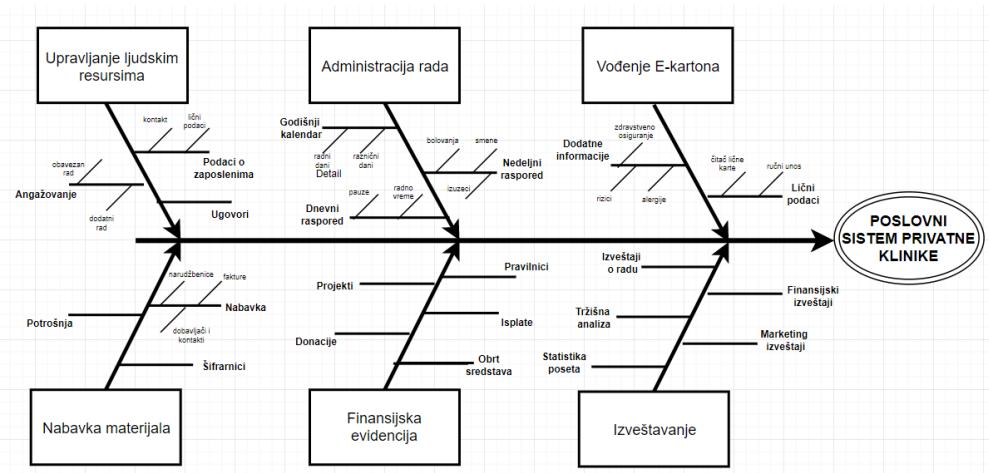


Slika 6.2 Predstavljanje okvira putem kontekstnog dijagrama na primeru poslovnog sistema privatne klinike

## POSLOVNI ZAHTEVI INFORMACIONOG SISTEMA PRIVATNE KLINIKE - ODREĐIVANJE OBIMA

*Primer sadržaja Dokumenta o viziji i okviru: Određivanje obima*

Osim dijagrama konteksta, dobar model za reprezentaciju obima je Stablo svojstava. Primer takvog modela je prikazan na slici 3:



Slika 6.3 Predstavljanje okvira putem Stabla svojstava na primeru poslovnog sistema privatne klinike

## ZADATAK ZA SAMOSTALNI RAD

### Tekst zadatka za samostalni rad

Na primeru sistema za upravljanje radom taksi udruženja koji je opisan u prvoj vežbi, uraditi sledeće:

1. Identifikovati poslovne ciljeve (10 min)
2. Predvideti poslovne rizike i verovatnoću da se oni ispolje (10 min)
3. Napisati izjavu o viziji (ili više njih ako je potrebno) (15 min)
4. Popisati glavne karakteristike novog sistema (10 min)
5. Primeni jednu od tehnika za određivanje okvira objašnjениh u predavanju (na primer Dijagram konteksta ili Stabla svojstava) (15 min)

## ✓ Poglavlje 7

### Domaći zadatak

#### DOMAĆI ZADATAK 3

##### *Tekst domaćeg zadatka*

Za sistem koji ste dobili za DZ01 uraditi sledeće:

1. Opisati poslovnu pozadinu i priliku
2. Napisati izjavu o viziji
3. Primeniti jednu od tehnika za reprezentaciju obima (na primer Dijagram konteksta ili Stablo svojstava)

*Napomene:*

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

## ZADATAK ZA RAD NA PROJEKTU

### *Tekst zadatka za rad na projektu*

Kreirajte Dokument o viziji i okviru za sistem koji ste odabrali kao temu svog projektnog zadatka. Popunite prva dva poglavlja.

## ✓ Poglavlje 9

### Zaključak

## ZAKLJUČAK

*Rezime glavnih poruka lekcije.*

1. Poslovni zahtevi predstavljaju skup informacija koje, u zbiru, opisuju potrebu koje rezultat projekata razvoja softvera mora da zadovolji sa stanovišta zahtevanih poslovnih rezultata.
2. Poslovna korist treba da donese istinsku vrednost sponzorima projekta i korisnicima proizvoda.
3. Vizija proizvoda opisuje konačan proizvod koji će ostvariti poslovne ciljeve i ona je stabilna. Okvir projekta utvrđuje koji deo vizije proizvoda će biti predmet rada i on je promenljiv.
4. Poslovni interesi aktera mogu biti i suprotstavljeni. Ako njihovo razrešenje dovede do menjanja poslovnih ciljeva, onda treba da zatražite i promenu poslovnih zahteva.
5. Dokument o viziji i okviru projekta sadrži poslovne ciljeve koji su osnova za aktivnosti planiranja projekta.
6. Poslovni zahtevi opisuju primarne koristi koje novi sistem obezbeđuje sponzorima, kupcima i korisnicima. Oni direktno utiču na implementaciju zahteva korisnika i na njen prioritet. Poslovni zahtevi definišu način za merenje ostvarivanja ovih poslovnih ciljeva.
7. Metrike uspeha pokazuju da li je projekat na putu da zadovolji svoje poslovne ciljeve.
8. Iskaz o viziji sumira dugoročnu svrhu i namenu proizvoda. On treba da odražava uravnotežen pogleda koji će da zadovolji očekivanja različitih aktera projekta.
9. Sumirajte glavne rizike projekta. Analizirajte realnost korišćenih prepostavki i mogućnosti prekida zavisnosti u projektu.
10. Okvir se definiše nizom funkcionalnih zahteva koji se planiraju za izvršenje (implementaciju) u okviru planiranog izdanja ili iteracije. Definisanje okvira narednih izdanja ima ograničenja i isključenja.
11. Potrebno je da napravite plan funkcionalnosti koje ćete razviti u svakoj fazi, iteraciji ili inkrementu.
12. Akteri projekta su pojedinci, grupe i organizacije koji su aktivno uključeni u projekat, pod uticajem su njegovih rezultata ili mogu da utiču na rezultate.
13. Dijagram konteksta vizualno ilustruje granicu između sistema koji se razvija i svega ostalog u univerzu. Utvrđuje i terminatore, kao sprege (interfejse) sistema sa spoljnjim svetom.
14. Mapa ekosistema pokazuje sve sisteme povezane sa vašim sistemom koji su u međusobnoj interakciji i prikazuju prirodu ovih interakcija.

15. Stablo svojstava prikazuje logično grupisana svojstva sistema, hijerarhijski podeljena na detaljnije nivoe prikazivanja.

16. Lista događaja utvrđuje spoljne događaje koji mogu da pokrenu neko ponašanje sistema.

## REFERENCA

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 5** 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.



## SE322 - INŽENJERSTVO ZAHTEVA

Značaj korisnika softvera u  
inženjerstvu zahteva

Lekcija 04

PRIRUČNIK ZA STUDENTE

# SE322 - INŽENJERSTVO ZAHTEVA

## Lekcija 04

### **ZNAČAJ KORISNIKA SOFTVERA U INŽENJERSTVU ZAHTEVA**

- ✓ Značaj korisnika softvera u inženjerstvu zahteva
- ✓ Poglavlje 1: Klase korisnika
- ✓ Poglavlje 2: Veze sa predstavnicima korisnika
- ✓ Poglavlje 3: Šampion proizvoda
- ✓ Poglavlje 4: Predstavljanje korisnika u agilnim projektima
- ✓ Poglavlje 5: Rešavanje sukobljenih zahteva
- ✓ Poglavlje 6: Vežba
- ✓ Poglavlje 7: Domaći zadatak
- ✓ Poglavlje 8: Projektni zadatak
- ✓ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

## ❖ Uvod

### UVOD

*Uspeh u softverskim zahtevima, a samim tim i u razvoju softvera, zavisi od približavanja glasa korisnika uhu programera.*

Ako delite naše uverenje da je učešće kupaca presudan faktor u pružanju odličnog softvera, osiguraćete da će poslovni analitičar (BA) i menadžer projekta za vaš projekt naporno raditi na angažovanju odgovarajućih predstavnika kupaca. Uspeh u softverskim zahtevima, a samim tim i u razvoju softvera, zavisi od približavanja glasa korisnika uhu programera. Da biste pronašli glas korisnika, poduzmite sledeće korake:

- Identifikujte različite klase korisnika vašeg proizvoda.
- Izaberite i radite sa pojedincima koji predstavljaju svaku klasu korisnika i ostale interesne grupe.
- Dogovorite se ko su donosioci odluka za vaš projekat.

Zahtevi iz perspektive kupca, neusklađenost proizvoda koji kupci očekuju da dobiju i onoga što programeri grade. Nije dovoljno jednostavno pitati nekoliko kupaca ili njihovog menadžera šta žele jednom ili dva puta, a zatim započeti kodiranje. Ako programeri naprave upravo ono što kupci prvobitno zahtevaju, verovatno će ih morati ponovo izraditi, jer kupci često ne znaju šta im zaista treba. Pored toga, BA možda ne razgovaraju sa pravim ljudima ili postavljaju prava pitanja.

Karakteristike koje korisnici predstavljaju kao "žele" ne moraju se izjednačiti sa funkcionalnošću koja im je potrebna da bi izvršili svoje zadatke novim proizvodom. Da bi stekao tačniji prikaz korisničkih potreba, poslovni analitičar mora prikupiti širok spektar korisničkih uloga, analizirati ga i razjasniti i precizirati šta je potrebno izgraditi kako bi korisnici mogli da rade svoj posao. BA ima glavnu odgovornost za evidentiranje potrebnih mogućnosti i svojstava novog sistema i za prenošenje tih informacija drugim zainteresovanim stranama. Ovo je iterativni proces za koji je potrebno vreme. Ako ne uložite vreme da postignete ovo zajedničko razumevanje - ovu zajedničku viziju željenog proizvoda - određeni ishodi su prepravljeni, propušteni rokovi, prekoračenja troškova i nezadovoljstvo kupaca.

### UVODNI VIDEO

*Trajanje video snimka: 2min 30sek*

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

## ✓ Poglavlje 1

### Klase korisnika

## VIDEO PREDAVANJE ZA OBJEKAT "KLASE KORISNIKA"

*Trajanje video snimka: 17min 12sek*

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

## KLASIFIKACIJA KORISNIKA

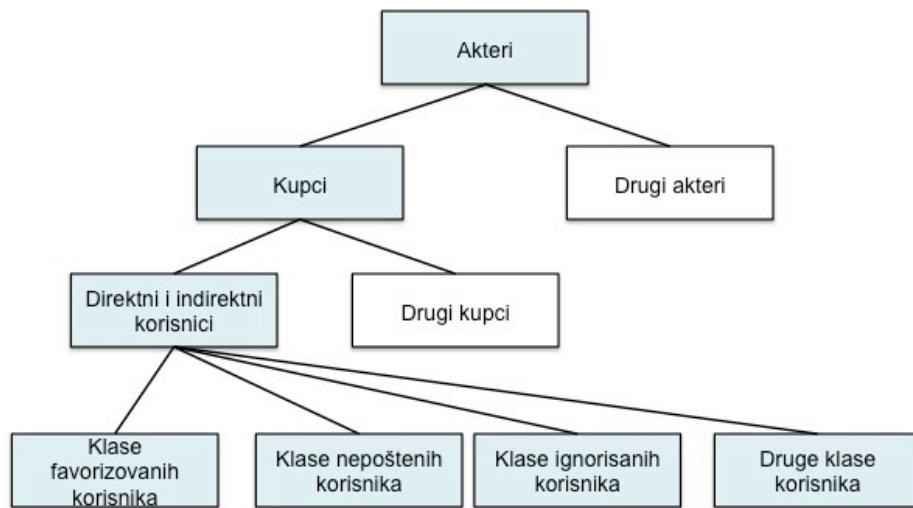
*Klasa korisnika je podklasa korisnika proizvoda, koji je podskup kupaca proizvoda, a koji je podskup njegovih interesnih grupa.*

Ljudi često govore o „korisniku“ za softverski sistem kao da svi korisnici pripadaju monolitnoj grupi sa sličnim karakteristikama i potrebama. U stvarnosti, većina proizvoda bilo koje veličine se sviđa raznolikosti korisnika sa različitim očekivanjima i ciljevima. Umesto da razmišljate o „korisniku“ u jednini, provedite neko vreme identificirajući više korisničkih klasa i njihove uloge i privilegije za vaš proizvod.

### Klasifikacija korisnika

Kao što je prikazano na slici 1, klasa korisnika je podklasa korisnika proizvoda, koji je podskup kupaca proizvoda, a koji je podskup njegovih interesnih grupa. Pojedinac može pripadati u više klasa korisnika. Na primer, administrator aplikacije takođe može ponekad komunicirati s njom kao običan korisnik. Korisnici proizvoda se mogu razlikovati - između ostalog - u sledećim pogledima, a možete grupisati korisnike u brojne različite klase korisnika na osnovu ovih vrsta razlika:

- Njihova pristupna privilegija ili nivoi sigurnosti (kao što su obični korisnik, gost korisnik, administrator)
- Zadaci koje obavljaju tokom svog poslovanja
- Funkcije koje koriste
- Učestalost kojom koriste proizvod
- Njihovo iskustvo u domenu primene i znanje o računarskim sistemima
- Platforme koje će koristiti (desktop računari, laptop računari, tableti, pametni telefoni, specijalizovani uređaji)
- Njihov maternji jezik
- Da li će komunicirati sa sistemom direktno ili indirektno



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

Slika 1.1 Hjерархија актера, купача, корисника и класа корисника

## KAKO IDENTIFIKOVATI I KLASIFIKOVATI KORISNIKE SOFTVERA?

*Pri identifikaciji korisničkih klasa razmišljajte o zadacima koje će različiti korisnici obavljati sa sistemom*

Pri identifikaciji korisničkih klasa razmišljajte o zadacima koje će različiti korisnici obavljati sa sistemom. Sve vrste finansijskih institucija imajuće sagovornike, zaposlene koji obrađuju zahteve za kredit, poslovne bankare i tako dalje. Pojedinci koji obavljaju takve aktivnosti - bilo da su u pitanju zvanja ili jednostavno uloge - imajuće slične funkcionalne potrebe za sistem u svim finansijskim institucijama. Svi prodavači moraju više ili manje da rade iste stvari, poslovni bankari rade manje ili više iste stvari, i tako dalje. Logičnija imena korisničkih klasa za bankarski sistem stoga mogu uključivati prodavača, kreditnog službenika, poslovnog bankara i menadžera filijala. Možete otkriti dodatne korisničke klase razmišljajući o mogućim slučajevima upotrebe, pričama korisnika i tokovima procesa i ko ih može izvoditi.

Određene klase korisnika mogu biti važnije od drugih za određeni projekat. **Favorizovane korisničke klase** su one čije je zadovoljstvo usko povezano sa postizanjem poslovnih ciljeva projekta. Prilikom rešavanja sukoba između zahteva iz različitih korisničkih klasa ili donošenja prioritetnih odluka, favorizovane korisničke klase imaju preferencijalni tretman. To ne znači da bi kupci koji plaćaju sistem (koji možda uopšte nisu korisnici) ili oni koji imaju najviše političkih uticaja nužno trebalo da budu favorizovani. To je pitanje usklađivanja sa poslovnim ciljevima.

**Klase neomiljenih korisnika** su grupe koje ne treba da koriste proizvod iz pravnih i bezbednosnih razloga. Možete njima namerno da otežate da urade ono što oni ne treba da rade. Primeri uključuju mehanizme zaštite pristupa, nivoe privilegija korisnika, funkcije antimalvera (za korisnike koji nisu ljudi) i evidenciju upotrebe. Zaključavanje korisničkog naloga nakon četiri neuspešna pokušaja prijavljivanja štiti od neovlaštene korisničke klase „korisnika impersonatora“, iako postoji rizik da će sistem neprijatno zaboraviti zakonite korisnike. Ako moja banka ne prepozna računar koji koristim, šalje mi e-poruku sa

jednokratnim pristupnim kodom koji moram da unesem pre nego što se mogu prijaviti. Ova funkcija je primenjena zbog nepoštene klase korisnika „ljudi koji su možda ukrali moje bankarske podatke“.

Možda ćete izabратi da zanemarite ostale klase korisnika. Da, koristiće proizvod, ali vi ne razvijate sistem u skladu sa njihovim potrebama. . Ako postoji bilo koje druge grupe korisnika koje nisu favorizovane, nepoštene ili zanemarene, one su od jednakog važnosti za definisanje potreba proizvoda.

## POSEBNI ZAHTEVI RAZLIČITIH KLASA KORISNIKA

*Svaka klasa korisnika će imati svoj vlastiti skup zahteva za zadatke koje pripadnici klase moraju da obavljaju.*

Svaka klasa korisnika će imati svoj vlastiti skup zahteva za zadatke koje pripadnici klase moraju obavljati. Moglo bi doći do preklapanja između potreba različitih klasa korisnika. Na primer, prodavači, bankarski bankari i kreditni službenici možda će morati da provere stanje na računu klijenta u banci. Različite klasе korisnika takođe bi mogle očekivati različit kvalitet, poput upotrebljivosti, koje će potaknuti izbor dizajna korisničkog interfejsa. Novi ili povremeni korisnici brinu se o tome koliko je sistem jednostavan za učenje. Takvi korisnici traže elemente poput menija, jednostavnih grafičkih korisničkih interfejsa, čarobnjaka i ekrana za pomoć. Kako korisnici stiču iskustvo sa sistemom, oni postaju više zainteresovani za efikasnost. Tada vrednuju prečice na tastaturi, mogućnosti prilagođavanja, trake sa alatkama i mogućnosti skripti.

Korisničke klase ne moraju biti ljudska bića. To bi mogli biti softverski agenti koji obavljaju uslugu u ime ljudskog korisnika, kao što su botovi. Softverski agenti mogu skenirati mreže radi informacija o robama i uslugama, sastaviti prilagođene feedove vesti, obraditi vašu dolaznu e-poštu, nadgledati fizičke sisteme i mreže zbog problema ili upada ili izvršiti rudarenje podataka. Internet agenti koji pretražuju veb lokacije zbog ranjivosti ili generišu neželjenu poštu su vrsta nepoštene korisničke klase

Ako identifikujete ove vrste nepoštenih klasa korisnika, možete navesti određene zahteve koji ne zadovoljavaju njihove potrebe, a radije će ih sprečiti. Na primer, alati za veb lokacije poput CAPTCHA koji potvrđuju da li je korisnik čovek ili robot koji pokušava da omete rad web aplikacije. Ovaj alat sprečava pristup tim ometačima, tj. lažnih korisnika.

Zapamtite, korisnici su podskup kupaca, koji su podskup zainteresovanih strana. Trebaće da razmotrite mnogo širi spektar potencijalnih izvora zahteva nego samo direktnе i indirektnе klase korisnika. Na primer, iako članovi razvojnog tima nisu krajnji korisnici sistema koji grade, potreban vam je vaš doprinos internim atributima kvaliteta kao što su efikasnost, izmenljivost, prenosivost i upotrebljivost. Jedna kompanija je otkrila da je svaka instalacija njihovog proizvoda bila skupa noćna mora sve dok nisu uveli „instalacijsku“ korisničku klasu kako bi se mogli fokusirati na zahteve poput razvoja arhitekture prilagođavanja njihovog proizvoda. Gledajte mnogo šire od gledanja samo očiglednih krajnjih korisnika kada pokušavate da identifikujete zainteresovane strane čiji su zahtevi potrebni.

# UTVRĐIVANJE VAŠIH KLASA KORISNIKA

*Prepoznajte i okarakterišite različite klase korisnika za svoj proizvod već na početku projekta.*

Prepoznajte i okarakterišite različite klase korisnika za svoj proizvod već na početku projekta kako biste mogli da postavljate zahteve od predstavnika svake važne klase. Korisna tehnika za to je sledeći obrazac saradnje. Započnite sa pitanjem sponzora projekta "Ko očekuje da će koristiti sistem". Zatim utvrdite onoliko korisničkih klasa koliko možete da se setite. Ne nervirajte se ako ih u ovoj fazi ima na desetine; kasnije ćete ih kondenzovati i kategorisati. Važno je ne zanemariti korisničku klasu, što može kasnije dovesti do problema kada se neko žali da dostavljeno rešenje ne zadovoljava njene potrebe. Zatim potražite grupe sa sličnim potrebama koje možete kombinovati ili tretirati kao glavnu korisničku klasu sa nekoliko podklasa. Pokušajte da svedete listu na oko 15 ili manje različitih klasa korisnika.

Različiti modeli analiza mogu vam pomoći da identifikujete klase korisnika. Spoljni entiteti prikazani izvan vašeg sistema na kontekstnom dijagramu su kandidati za korisničke klase. Organizaciona karta organizacije takođe vam može pomoći da otkrijete potencijalne korisnike i ostale zainteresovane strane. Slika 2 ilustruje deo organizacione šeme Contoso Pharmaceuticals. Skoro svi potencijalni korisnici sistema će se verovatno naći negde na ovoj tabeli. Dok izvodite analizu zainteresovanih strana i korisnika, proučite organizacioni grafikon da potražite:

- Odeljenja koja učestvuju u poslovnom procesu.
- Odeljenja na koja utiče poslovni proces.



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

Slika 1.2 Deo organizacione strukture firme Contoso Pharmaceuticals

- Odeljenja ili imena uloga u kojima se mogu naći direktni ili indirektni korisnici.
- Korisničke klase koje obuhvataju više odeljenja.
- Odeljenja koja mogu imati interfejs za spoljne zainteresovane strane izvan kompanije.

Analiza organizovanog grafikona smanjuje verovatnoću da ćete prevideti važnu klasu korisnika unutar te organizacije. Pokazuje vam gde treba da tražite potencijalne predstavnike za određene korisničke klase, kao i pomaže vam da odredite koji bi mogli da budu ključni zahtevi za donošenje odluka. U jednom odeljenju možete pronaći više korisničkih klasa sa

različitim potrebama. Suprotno tome, prepoznavanje iste korisničke klase u više odeljenja može pojednostaviti iznošenje zahteva.

## PRIMER KORISNIČKIH KLASA - CHEMICAL TRACKING SYSTEM

*Dokumentujte klase korisnika i njihove karakteristike, odgovornosti i fizičke lokacije u specifikaciji softverskih zahteva (SRS) ili u planu zahteva za svoj p*

Proučavanje organizacionog grafikona pomaže vam da procenite sa koliko predstavnika korisnika ćete te morati da sarađujete da biste bili sigurni da temeljno razumete potrebe široke korisničke zajednice. Takođe pokušajte da shvatite koju vrstu informacija mogu da daju korisnici iz svakog odeljenja na osnovu njihove uloge u organizaciji i perspektive njihovog odeljenja za projekat.

Dokumentujte klase korisnika i njihove karakteristike, odgovornosti i fizičke lokacije u specifikaciji softverskih zahteva (SRS) ili u planu zahteva za svoj projekat. Proverite te podatke u odnosu na bilo kakve informacije koje možda već imate o profilima zainteresovanih strana u viziji i dokumentu o dometu da biste izbegli sukobe i dupliranje. Uključite sve relevantne informacije o svakoj korisničkoj klasi, poput njihove relativne ili apsolutne veličine i koje klase su favorizovane. Ovo će pomoći timu da prioretizuje zahteve za promenom i kasnije sprovede procene uticaja. Procene obima i vrste sistemskih transakcija pomažu ispitivačima da razviju profil upotrebe sistema kako bi mogli planirati svoje aktivnosti verifikacije. Rukovodilac projekta i poslovni analitičar sistema za praćenje hemikalija o kome se razgovaralo u prethodnim poglavljima identifikovalo je klase korisnika i karakteristike prikazane u tabeli na slici 3.

Razmislite o izradi kataloga korisničkih klasa koje se ponavljaju u više aplikacija. Definisanje korisničkih klasa na nivou preduzeća omogućava vam ponovnu upotrebu opisa korisničkih klasa u budućim projektima. Sledeći sistem koji gradite može služiti potrebama nekih novih korisničkih klasa, ali verovatno će ih koristiti i klase korisnika iz vaših ranijih sistema. Ako uključite opise korisničke klase u SRS projekta, možete uključiti stavke iz kataloga korisničke klase za višekratnu upotrebu referencom i samo napisati opise novih grupa koje su specifične za tu aplikaciju.

Naziv	Broj	Opis
Hemičari (favorizovani)	Oko 1000 lociranih u 6 zgrada	Hemičari će tražiti hemikalije od dobavljača i iz skladišta hemikalija. Svaki hemičar koristiće sistem nekoliko puta dnevno, uglavnom za traženje hemikalija i praćenje hemijskih kontejnera u i iz laboratorija. Hemičari moraju da pretražuju kataloge dobavljača za određene hemijske strukture uvezene iz alata koji koriste za crtanje konstrukcija.
Kupci	5	Kupci u odeljenju za nabavku obraduju hemijske zahteve. Oni naručuju i prate narudžbe kod eksternih dobavljača. Oni malo znaju o hemiji i trebaju im jednostavne mogućnosti za pretragu kataloga dobavljača. Kupci neće koristiti funkcije sistema za praćenje kontejnera. Svaki kupac će sistem koristiti u proseku 25 puta dnevno.
Zaposleni u skladištima hemikalija	6 tehničara 1 nadzornik	Osoblje hemijskog skladišta upravlja inventarom više od 500.000 kontejnera za hemikalije. Oni će isporučivati kontejnere iz tri skladišta, tražiti nove hemikalije od dobavljača i pratiti kretanje svih kontejnera u i van skladišta. Oni su jedini korisnici funkcije prijavljivanja zaliba. Zbog velikog obima transakcija, funkcije koje koristi samo osoblje skladišta hemijskih proizvoda moraju biti automatizovane i efikasne.
Zaposleni u odeljenjima za zdravlje i bezbednost	1 menadžer	Osoblje Odeljenja za zdravlje i bezbednost koristiće sistem samo za generisanje unapred definisanih kvartalnih izveštaja koji su u skladu sa federalnim i državnim propisima o izveštavanju o korišćenju hemikalija i odlaganja. Direktor Odeljenja za zdravlje i bezbednost periodično će zahtevati izmene u izveštajima kako se promene vladine uredbe. Ove promene izveštaja su od najvećeg prioriteta, a primena će biti kritična za vreme.

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

Slika 1.3 Korisničke klase u Chemical Tracking System

## PERSONA - HIPOTETIČKI REPREZENT KORISNIČKE KLASE

*Persona je opis hipotetičke, generičke osobe koja služi tipičan represent grupi korisnika koja imaju slične karakteristike i potrebe.*

Da biste lakše oživeli vaše korisničke klase, razmislite o kreiranju personalizacije za svaki od njih, opisu reprezentativnog člana korisničke klase. **Persona** je opis hipotetičke, generičke osobe koja služi kao tipičan reprezentant grupe korisnika koja imaju slične karakteristike i potrebe. Možete da koristite persone kako bi vam pomogle da razumete zahteve i da dizajnirate korisničko iskustvo na način koji najbolje ispunjava potrebe određenih zajednica korisnika.

Persona može poslužiti kao rezervno mesto kada BA nema pri ruci stvarnog predstavnika korisnika. Umesto da se napredak zaustavi, BA može zamisliti osobu koja obavlja određeni zadatak ili pokušati da proceni kakve će biti perspektive osobe i tako izraditi početnu tačku zahteva koja treba biti potvrđena kada je stvarni korisnik dostupan. Detalji o personi za komercijalnog kupca uključuju socijalne i demografske karakteristike i ponašanje, preferencije, smetnje i slične informacije. Proverite da lični likovi koje zaista stvarate predstavljaju njihove klase korisnika, zasnovane na tržišnim, demografskim i etnografskim istraživanjima.

Evo primera persona za jednu korisničku klasu u sistemu za praćenje hemikalija:

*"Fred (41) je hemičar kompanije Contoso Pharmaceuticals otkad je doktorirao. Pre 14 godina. On nema puno strpljenja prema kompjuterima. Fred obično radi na dva projekta istovremeno u povezanim hemijskim oblastima. Njegova laboratorijska soba sadrži oko 300 boca hemikalija i benzinskih boca. U proseku će mu trebati četiri nove hemikalije iz skladišta. Dve od njih će biti komercijalne hemikalije na zalihama, jedna će se morati naručiti, a jedna će se isporučiti iz privatnih uzoraka Contoso hemikalija. Ponekad će Fredu biti potrebna opasna hemikalija za koju je potrebna posebna obuka za sigurno rukovanje. Kad prvi put kupuje hemikalije, Fred želi da mu podaci o sigurnosti materijala budu automatski poslati. Svake godine Fred*

će sintetizovati oko 20 novih zaštićenih hemikalija da bi otišlo u skladište. Fred želi da se izveštaj o njegovoj upotrebi hemikalija za prethodni mesec automatski generiše i pošalje mu e-poštom kako bi mogao da prati svoju izloženost hemikalijama."

Dok poslovni analitičar (BA) istražuje zahteve hemičara, on može razmišljati o Fredu kao o arhetipu ove korisničke klase i pitati se: „Šta bi Fred trebalo da uradi?“ Rad sa personom čini da procesi misaonih zahteva postanu opipljiviji nego ako jednostavno razmislite o tome šta bi čitava bezlična grupa ljudi želela. Neki ljudi biraju nasumično ljudsko lice odgovarajućeg pola kako bi persona izgledala još stvarnije.

## ▼ Poglavlje 2

# Veze sa predstavnicima korisnika

## PREDSTAVNICI KORISNIKA

### *Predstavnici korisnika pružaju "glas korisnika"*

Svaka vrsta projekta - korporativni informacioni sistemi, komercijalne aplikacije, ugrađeni sistemi, veb lokacije, ugovoreni softver - trebaju odgovarajuće predstavnike koji bi mogli da pruže glas korisniku. Ovi korisnici trebalo bi da budu uključeni tokom čitavog životnog ciklusa razvoja, a ne samo u fazi izolovanih potreba na početku projekta. Svakoj korisničkoj klasi potreban je neko ko će govoriti za nju.

Najlakše je dobiti pristup stvarnim korisnicima kada razvijate aplikacije za upotrebu u sopstvenoj kompaniji. Ako razvijate komercijalni softver, možete angažovati ljude sa vašeg beta testiranja ili veb lokacija sa ranim izdanjem da biste uneli zahteve mnogo ranije u procesu razvoja. Razmislite o postavljanju fokus grupe trenutnih korisnika vaših proizvoda ili proizvoda vaših konkurenata. Umesto da samo nagadate šta bi vaši korisnici možda želeli, pitajte neke od njih.

Jedna kompanija je zatražila od fokus grupe da izvrši određene zadatke pomoću različitih digitalnih kamera i računara. Rezultati su pokazali da je kompanijskom softveru za fotoaparate bilo potrebno predugo da bi se obavljala najčešća operacija zbog dizajnerske odluke koja je doneta da se uklopi i manje verovatni scenariji. Kompanija je promenila svoju sledeću kameru kako bi umanjila pritužbe kupaca na brzinu. Budite sigurni da fokus grupa predstavlja vrste korisnika čije potrebe bi trebalo da pokreću vaš razvoj proizvoda. Uključite i ekspertske i manje iskusne kupce. Ako vaša fokus grupa predstavlja samo mlade osobe, možda ćete naići na brojne sofisticirane i tehnički teške zahteve za koje je malo kupaca zainteresovano.

Slika 1 ilustruje neke tipične komunikacijske puteve koji povezuju glas korisnika sa uhom programera. Jedno istraživanje pokazalo je da je korišćenje više vrsta komunikacijskih veza i direktnije veze između programera i korisnika dovelo do uspešnijih projekata. Najdirektnija komunikacija događa se kada programeri mogu sami razgovarati s odgovarajućim korisnicima, što znači da programer takođe obavlja ulogu poslovnog analitičara. Ovo može da radi na vrlo malim projektima, pod uslovom da programer koji radi ima odgovarajuću BA veština, ali ne obuhvata velike projekte sa hiljadama potencijalnih korisnika i desetinama programera.

Kategorija	Aktivnosti
Planiranje	Poboljšava okvir i granice sistema Utvrđuje druge sisteme sa kojim sistem treba da bude u interakciji Ocjenjuje efekat novog sistema na poslovne operacije kupca Definiše tranzicioni put od sadadnjih aplikacija ili ručnih operacija Utvrđuje relevantne standarde i zahteve sertifikacije
Zahtevi	Prikuplja zahteve ostalih korisnika u klasi koju predstavlja Razvija scenarije korišćenja, slučajeva korišćenja, i priče korisnika Rešava konflikte između zahteva iz svoje klase korisnika Određuje prioritete primene zahteva Obezbeđuje unos potrebnih performansi i drugih zahteva kvaliteta Ocjenjuje prototipove U saradnji sa drugima, rešava konflikte između zahteva različitih klasa korisnika Obezbeđuje specifične algoritme
Potpovrđivanje (validation) i proveravanje (verification)	Recenzija specifikacije zahteva Definiše kriterijume prihvatanja Razvija testove prihvatanje korisnika iz scenarija korišćenja Obezbeđuje skup podataka za testiranja od poslovnog dela firme Realizuje beta testiranje ili testiranja prihvatanja od strane korisnika
Pomoć korisnicima	Piše delove teksta namenjen korisnicima sistema Učestvuje u pripremi materijala za obuku ili kurseve

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

Slika 2.1 Putevi komunikacija korisnika i programera

## ▼ Poglavlje 3

# Šampion proizvoda

## ULOGA ŠAMPIONA PROIZVODA

*Šampion proizvoda služi kao primarni interfejs između članova jedne korisničke klase i poslovnog analitičara projekta*

Šampion proizvoda služi kao primarni interfejs između članova jedne korisničke klase i poslovnog analitičara projekta. U idealnom slučaju, šampioni proizvoda će biti stvarni korisnici, a ne surogati kao što su sponzori finansiranja, marketinško osoblje, menadžeri korisnika ili programeri softvera koji sebe zamišljaju kao korisnike. Šampioni proizvoda prikupljaju zahteve od ostalih članova korisničke klase koje predstavljaju i usklađuju neusaglašenosti. Razvoj zahteva je stoga zajednička odgovornost BA i odabranih korisnika, mada BA zapravo treba da napiše dokumentaciju o zahtevima. Dovoljno je teško napisati dobre zahteve ako to učinite za život; nije realno očekivati od korisnika koji nikada ranije nisu napisali zahteve da rade dobar posao.

Najbolji šampioni proizvoda imaju jasnu viziju novog sistema. Oduševljeni su jer vide kako će to koristiti njima i njihovim vršnjacima. Šampioni bi trebali biti efikasni komunikatori koje poštuju njihove kolege. Potrebno im je temeljno razumevanje domena aplikacije i operativnog okruženja rešenja. Odlični šampioni proizvoda traže i druge zadatke, tako da ćete morati da izradite ubedljiv slučaj zbog čega su određeni pojedinci kritični za uspeh projekta. Na primer, šampioni proizvoda mogu da usvoje aplikaciju od strane korisničke zajednice, što bi mogao biti pokazatelj uspeha koji će menadžeri ceniti.

Pristup šampionu proizvoda najbolje funkcioniše ako je svaki šampion u potpunosti ovlašten da donosi obavezujuće odluke u ime korisničke klase koju predstavlja. Ako odluke šampiona rutinski nadjačavaju drugi, njegovo vreme i dobra volja se gube. Međutim, šampioni moraju imati na umu da oni nisu jedini kupci. Problemi nastaju kada pojedinac koji ispunjava ovu kritičnu ulogu veze ne komunicira na odgovarajući način sa svojim kolegama i iznosi samo svoje želje i ideje.

Otkrili smo da su dobri šampioni proizvoda napravili veliku razliku u našim projektima, pa im nudimo javnu nagradu i priznanje za njihov doprinos. Naši timovi za razvoj softvera uživali su dodatnu korist od pristupa proizvodnom šampionu. Na nekoliko projekata imali smo odlične šampione koji su u naše ime razgovarali sa kolegama kada su se kupci pitali zašto softver još nije gotov. "Ne brinite oko toga", poručili su šampioni svojim vršnjacima i njihovim menadžerima. „Razumem i slažem se sa pristupom softverskog tima softverskom inženjeringu. Vreme koje trošimo na njihove potrebe pomoći će nam da dobijemo sistem

koji nam je zaista potreban i dugoročno ćemo uštedeti vreme.” Takva saradnja pomaže u razbijanju napetosti koja može nastati između kupaca i razvojnih timova.

## SPOLJNI ŠAMPIONI

*Ako imate raznovrsnu korisničku bazu, prvo identifikujte osnovne zahteve koji su zajednički za sve kupce. Zatim definišite dodatne zahteve koji su specifični*

Kada razvijate komercijalni softver, može biti teško pronaći šampiona proizvoda izvan vaše kompanije. Kompanije koje razvijaju komercijalne proizvode ponekad se oslanjaju na interne stručnjake za teme ili spoljne konsultante koji će služiti kao surogati za stvarne korisnike, koji su možda nepoznati ili ih je teško angažovati. Ako imate bliske radne odnose sa nekim većim korporativnim kupcima, oni će možda pozdraviti priliku da učestvuju u iznošenju zahteva. Možete dati spoljnim šampionima proizvoda ekonomске podsticaje za njihovo učešće. Razmislite o tome da im ponudite popuste na proizvod ili da platite za vreme koje provode radeći s vama na zahteve. I dalje se suočavate sa izazovom kako izbeći da čujete samo zahteve šampiona i prevideti potrebe drugih zainteresovanih strana. Ako imate raznovrsnu korisničku bazu, prvo identifikujte osnovne zahteve koji su zajednički za sve kupce. Zatim definišite dodatne zahteve koji su specifični za pojedinačne korporativne kupce, segmente tržišta ili klase korisnika.

Kad god je šampion proizvoda bivši ili simulirani korisnik, pripazite da ne postoji veza između percepcija šampiona i trenutnih potreba stvarnih korisnika. Neki se domeni brzo menjaju, dok su drugi stabilniji. Bez obzira na to, ako ljudi više ne rade u ulozi, jednostavno bi zaboravili sitnice svakodnevnog posla. Suštinsko pitanje je da li šampion proizvoda, bez obzira na pozadinu ili trenutni posao, može tačno da predstavlja potrebe današnjih stvarnih korisnika.

Druga alternativa je angažovanje odgovarajućeg šampiona proizvoda koji ima pravu pozadinu. Jedna kompanija, koja je razvila maloprodajni i back-office sistem za određenu industriju, angažovala je tri menadžera prodavnica koji će služiti kao prvaci proizvoda sa punim radnim vremenom. Kao još jedan primer, moj dugogodišnji porodični lekar, Art, napustio je medicinsku praksu da postane glas lekara u kompaniji za medicinski softver. Art-ov novi poslodavac verovao je da je vredno tog troška angažovati doktora koji će kompaniji pomoći da izradi softver koji bi drugi lekari prihvatali. Treća kompanija je zaposlila nekoliko bivših radnika jednog od svojih glavnih kupaca. Ovi ljudi su pružili dragocenu pomoć domenu ekspertize kao i uvid u politiku organizacije korisnika. Za ilustraciju alternativnog modela angažovanja, jedna kompanija je imala nekoliko korporativnih kupaca koji su intenzivno koristili njihove sisteme fakturisanja. Umesto da dovede kupce šampiona proizvoda, kompanija u razvoju послala je BA na lokacije za kupce. Kupci su svojevoljno posvetili neko vreme svog osoblja pomaganju BA-ima da dobiju prave zahteve za novi sistem fakturisanja.

## OČEKIVANJA ŠAMPIONA PROIZVODA

*Koristite ovu tabelu aktivnosti šampiona kao polaznu tačku za pregovaranje o obavezama svakog šampiona.*

Da biste pomogli šampionima proizvoda da uspeju, dokumentujte šta očekujete od šampiona. Ova pismena očekivanja mogu vam pomoći da izgradite slučaj da specifični pojedinci ispunjavaju ovu kritičnu ulogu. Tabela na slici 1 prikazuje neke aktivnosti koje šampioni proizvoda mogu obavljati. Neće svaki šampion uraditi sve ovo; koristite ovu tabelu kao polaznu tačku za pregovaranje o obavezama svakog šampiona.

Kategorija	Aktivnosti
Planiranje	Poboljšava okvir i granice sistema Utvrđuje druge sisteme sa kojim sistem treba da bude u interakciji Ocenjuje efekat novog sistema na poslovne operacije kupca Definiše tranzicioni put od sadašnjih aplikacija ili ručnih operacija Utvrđuje relevantne standarde i zahteve sertifikacije
Zahtevi	Prikuplja zahteve ostalih korisnika u klasi koju predstavlja Razvija scenarije korišćenja, slučajeva korišćenja, i priče korisnika Rešava konflikte između zahteva iz svoje klase korisnika Određuje prioritete primene zahteva Obezbeđuje unos potrebnih performansi i drugih zahteva kvaliteta Ocenjuje prototipe U saradnji sa drugima, rešava konflikte između zahteva različitih klasa korisnika Obezbeđuje specifične algoritme
Potvrđivanje (validation) i proveravanje (verification)	Recenzija specifikacije zahteva Definiše kriterijume prihvatanja Razvija testove prihvatanje korisnika iz scenarija korišćenja Obezbeđuje skup podataka za testiranja od poslovnog dela firme Realizuje beta testiranje ili testiranja prihvatanja od strane korisnika
Pomoć korisnicima	Piše delove teksta namenjen korisnicima sistema Učestvuje u pripremi materijala za obuku ili kurseve

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

Slika 3.1 Komunikacija između šampiona proizvoda i inženjera razvoja proizvoda

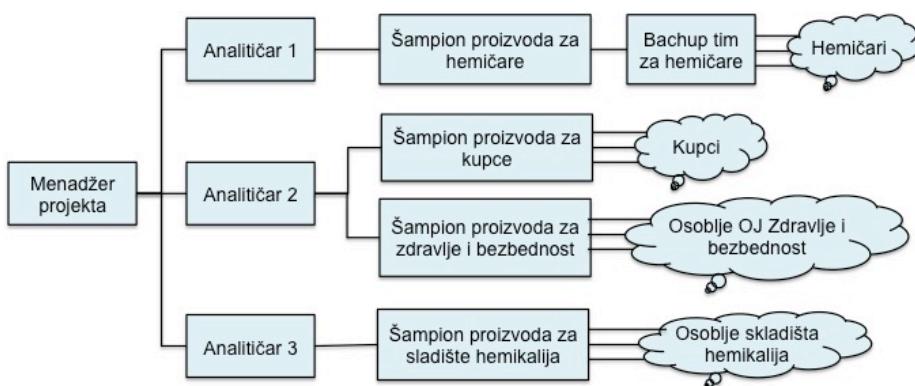
## VIŠESTRUKI ŠAMPIONI PROIZVODA

*Za četiri glavne klase korisnika, bila su potrebna četiri šampiona. Njima nije dodeljeno puno radno vreme, ali svaki je proveo nekoliko sati nedeljno radeći*

Jedna osoba retko može opisati potrebe za sve korisnike aplikacije. Sistem praćenja hemikalija imao je četiri glavne klase korisnika, pa su mu bila potrebna četiri šampiona proizvoda izabranih iz interne zajednice korisnika u kompaniji Contoso Pharmaceuticals. Slika 2 ilustruje kako je rukovodilac projekta osnovao tim BA i šampiona proizvoda da bi iz pravih izvora izvukao prave zahteve. Ovim šampionima nije dodeljeno puno radno vreme, ali svaki je proveo nekoliko sati nedeljno radeći na projektu. Tri BA su radila sa četiri šampiona proizvoda

na zahtevu, analiziranju i dokumentovanju njihovih zahteva. (Jedan BA je radio sa dva šampiona proizvoda, jer su klase korisnika i odeljenja za zdravlje i bezbednost kupca bile male i imale malo zahteva.) Jedan od BA je sakupio sve podatke u objedinjenu SRS.

Nismo očekivali da će jedna osoba pružiti sve raznolike zahteve stotinama hemičara kompanije Contoso. Don, šampion proizvoda za hemijsku klasu korisnika, okupio je rezervni tim od pet hemičara iz drugih delova kompanije. Predstavljali su potklase unutar široke korisničke klase hemičara. Ovaj hijerarhijski pristup uključio je dodatne korisnike u razvoj potreba, izbegavajući na taj način masovne radionice ili desetine pojedinačnih intervjua. Don uvek teži ka konsenzusu. Međutim, voljno je donosio potrebne odluke kada sporazum nije postignut da bi projekat mogao da napreduje. Nije potreban rezervni tim kada je klasa korisnika bila dovoljno mala ili kohezivna da jedan pojedinac uistinu može predstavljati potrebe grupe.



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

Slika 3.2 Model šampiona proizvoda za Chemical Tracking System

## KAKO IDEJU O ŠAMPIONU PROIZVODA PRODATI NARUČIOCU?

*Podsetite menadžment da je šampion proizvoda ključni saradnik koji može da pomogne projektu da postigne svoje poslovne ciljeve*

Očekujte da ćete naići na otpor kada predložite ideju da u svojim projektima imate šampiona proizvoda. "Korisnici su previše zauzeti." "Menadžment želi da donosi odluke." "Usporiće nas." "Ne možemo to da priuštimo." „Ne znam šta bih trebao da radim kao šampion proizvoda.“ Neki korisnici neće želeti da sarađuju na projektu koji će ih naterati da promene način rada ili čak da im preti posao. Rukovodioci ponekad nerado delegiraju svoju odgovornost za zahteve na obične korisnike.

Odvajanje poslovnih zahteva od potreba korisnika ublažava neke od tih neprijatnosti. Kao stvarni korisnik, šampion proizvoda donosi odluke na nivou korisničkih zahteva u granicama opsega koje postavljaju poslovni zahtevi. Sponzor menadžmenta zadržava ovlašćenje da donosi odluke koje utiču na viziju proizvoda, obim projekta, prioritete u vezi sa poslovanjem, raspored ili budžet. Dokumentiranje i pregovaranje o ulozi i odgovornosti svakog šampiona proizvoda pružaju šampionima nivo komfora o onome što se od njih traži. Podsetite menadžment da je šampion proizvoda ključni saradnik koji može da pomogne projektu da postigne svoje poslovne ciljeve.

Ako nađete na otpor, istaknite da je nedovoljno učešće korisnika vodeći uzrok neuspeha softverskog projekta. Podsetite demonstrante na probleme koje su iskusili na prethodnim projektima koji sežu u nedovoljan unos korisnika. Svaka organizacija ima grozne priče o novim sistemima koji nisu udovoljili potrebama korisnika ili nisu ispunili očekivanja u upotrebljivosti ili performansama. Ne možete sebi da priuštite da ponovo izgradite ili odbacite sisteme koji se ne mere jer niko ne razume zahteve. Šampioni proizvoda pružaju jedan od načina da blagovremeno dobiju taj najvažniji unos kupca, a ne na kraju projekta kada su kupci razočarani, a programeri umorni.

## ZAMKE ŠAMPIONA PROIZVODA KOJE TREBA IZBEGAVATI

*Model šampiona proizvoda uspeo je u mnogim okruženjima. Deluje samo kada šampioni razumeju svoje odgovornosti, imaju autoritet da donose odluke*

Model šampiona proizvoda uspeo je u mnogim okruženjima. Deluje samo kada šampioni proizvoda razumeju i prijave se za svoje odgovornosti, imaju autoritet da donose odluke na nivou korisničkih potreba i imaju na raspolaganju vreme za obavljanje posla. Pazite na sledeće potencijalne probleme:

- Menadžeri nadjačavaju odluke koje donosi kvalifikovani i propisno ovlašćeni šampion proizvoda. Možda menadžer ima neku novu ideju u poslednjem trenutku ili misli da zna šta korisnicima treba. Ovo ponašanje često rezultira nezadovoljnim korisnicima i frustriranim prvacima proizvoda koji smatraju da im uprava ne veruje.
- Šampion proizvoda koji zaboravi da zastupa druge kupce i predstavlja samo svoje zahteve, neće raditi dobro. Možda je zadovoljan ishodom, ali drugi verovatno neće biti.
- Stariji korisnik može imenovati manje iskusnog korisnika kao šampiona, jer on nema vremena da sam obavi posao. To može dovesti do toga da stariji korisnik "iz pozadine" utiče na odvijanje projekta.

Pazite na korisnike koji nameravaju da govore za korisničku klasu kojoj ne pripadaju. Retko, pojedinac iz nekog razloga može aktivno pokušati da blokira BA da radi sa idealnim kontaktima. Što se tiče sistema za praćenje hemikalija, šampion proizvoda za osoblje skladišta hemijskih proizvoda - ona bivša hemičarka - u početku je insistirao na pružanju onoga što je smatrao potrebama hemičke klase korisnika. Nažalost, njen stav o trenutnim potrebama hemičara nije bio tačan. Bilo je teško uveriti je da to nije njen posao, ali BA nije dozvolio da ga zastraši. Rukovodilac projekta sastavio je zasebnog šampiona proizvoda za hemičare, koji su uradili sjajan posao prikupljanja, procene i prenošenja potreba zajednice.

## ▼ Poglavlje 4

# Predstavljanje korisnika u agilnim projektima

## VIDEO PREDAVANJE ZA OBJEKAT "PREDSTAVLJANJE KORISNIKA U AGILNIM PROJEKTIMA"

*Trajanje video snimka: 11min 59sek*

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

## VLASNIK PROIZVODA

*Vlasnik proizvoda definiše viziju proizvoda i odgovoran je za razvoj i određivanje prioriteta sadržaja zaostalih proizvoda*

Česti razgovori između članova projektnog tima i odgovarajućih kupaca su najefikasniji način za rešavanje mnogih problema sa zahtevima i preciziranje specifičnosti zahteva kada su potrebni. Pismena dokumentacija, koliko god bila detaljna, nepotpuna je zamena za ove tekuće komunikacije. Temeljno načelo Ekstremnog programiranja, jedne od ranih metoda agilnog razvoja, je prisustvo stalnog kupca na terenu za diskusije. Neke agilne metode razvoja uključuju jednog predstavnika aktera koji se zove vlasnik proizvoda u timu koji će služiti kao glas kupca. Vlasnik proizvoda definiše viziju proizvoda i odgovoran je za razvoj i određivanje prioriteta sadržaja bekloggingu proizvoda. Beklogging (**backlogging**) je prioritetna lista korisničkih priča - zahtevi - za proizvod i njihovo dodeljivanje narednim iteracijama, koje se zovu sprintovi u agilnom metodu razvoja zvanim Scrum. **Vlasnik proizvoda obuhvata sva tri nivoa zahteva: posao, korisnik i funkcionalnost.** On u suštini razvlači šampione proizvoda i funkcije poslovnog analitičara, predstavljajući kupca, definisanjući karakteristike proizvoda, postavljajući im prioritete i tako dalje. Konačno, neko mora donositi odluke o tome koje će tačno mogućnosti isporučiti u proizvodu i kada. U Scrum-u je to odgovornost vlasnika proizvoda.

Idealno stanje vlasništva jednog proizvoda nije uvek praktično. Znamo za jednu kompaniju koja je implementirala paketno rešenje za vođenje svog osiguranja. Organizacija je bila prevelika i složena da bi imala jedna osoba koja je sve razumela dovoljno detaljno da bi mogla doneti sve odluke o implementaciji. Umesto toga, kupci su odabrali vlasnika proizvoda iz svakog odeljenja da poseduje prioritete funkcije koju taj odeljak koristi. CIO kompanije poslužio je kao vodeći vlasnik proizvoda. CIO je razumeo celokupnu viziju

proizvoda, tako da je mogao osigurati da odeljenja idu na put da isporuče tu viziju. Bio je odgovoran za donošenje odluka kada je došlo do sukoba između vlasnika proizvoda na nivou odeljenja.

Dobro je koristiti prostorije kupca na licu mesta i blisku saradnju korisnika sa programerima koji agilni metodi podržavaju. U stvari, snažno osećamo da svi razvojni projekti garantuju ovaj naglasak na uključivanju korisnika. Kao što ste videli, svi projekti, osim najmanjih, imaju više korisničkih klasa, kao i brojne dodatne aktere čiji interesi moraju biti zastupljeni. U mnogim slučajevima nije realno očekivati da će pojedinac moći adekvatno razumeti i opisati potrebe svih relevantnih korisničkih klasa, niti doneti sve odluke povezane sa definicijom proizvoda. Posebno sa internim korporativnim projektima, generalno će biti bolje koristiti reprezentativnu strukturu poput modela šampiona proizvoda kako bi osigurali adekvatno angažovanje korisnika.

## ODNOS VLASNIK PROIZVODA I ŠAMPION PROIZVODA

*“Nema zamene za to da imate prave ljudе, u pravoj ulozi, na pravom mestu, sa pravim stavom”.*

Šeme vlasnika proizvoda i šampiona proizvoda se međusobno ne isključuju. Ako vlasnik proizvoda deluje u ulozi poslovnog analitičara, a ne kao sam predstavnik aktera, mogao bi da uspostavi strukturu sa jednim ili više šampiona proizvoda da bi video da li najprikladniji izvori daju doprinos. S druge strane, vlasnik proizvoda mogao bi sarađivati sa jednim ili više poslovnih analitičara, koji potom rade sa zainteresovanim stranama kako bi razumeli njihove zahteve. Vlasnik proizvoda bi tada služio kao krajnji donositelj odluka.

Jednom sam napisao programe za naučnika istraživanja koji je sedeо desetak stopa od mog stola. John bi mogao da pruži trenutne odgovore na moja pitanja, pruži povratne informacije o dizajnu korisničkog interfejsa i razjasni naše neformalno napisane zahteve. Jednog dana Džon se preselio u novu kancelariju, iza ugla na istom spratu iste zgrade, udaljene oko 100 stopa. Osetio sam trenutni pad produktivnosti programiranja zbog kašnjenja vremena ciklusa u dobijanju Džon-ovog unosa. Provodio sam više vremena u rešavanju problema jer sam ponekad krenuo pogrešnim putem pre nego što sam uspeo da popravim kurs. Ne možete zameniti prave kupce koji su neprekidno dostupni programerima i na licu mesta i "vidljiva na licu mesta". Pazite, ipak, na prečeste prekide koji ljudima otežavaju da preusmere pažnju na njihov rad. Trajanje može potrajati i do 15 minuta u visoko produktivnom, fokusiranom stanju uma koje se naziva protok

Kupac na licu mesta ne garantuje željeni ishod. Moj kolega Chris, rukovodilac projekta, uspostavio je okruženje razvojnog tima sa minimalnim fizičkim barijerama i angažovao dva šampiona proizvoda. Chris je ponudio ovaj izveštaj: „lako se čini da blizina deluje

na razvojni tim, rezultati sa proizvodnim šampionima su pomešani. Jedan je sedeо u našoj sredini i još uvek je uspeо da nas izbegne. Novi šampion obavlja dobar posao interakcije sa programerima i zaista je omogućio brzi razvoj softvera. "Nema zamene za to da imate prave ljude, u pravoj ulozi, na pravom mestu, sa pravim stavom".

## ▼ Poglavlje 5

# Rešavanje sukobljenih zahteva

## KO REŠAVA PROBLEM SUKOBLJENIH ZAHTEVA?

*Pri sukobu zahteva odluke treba donositi što niže u hijerarhiji organizacije od strane dobro informisanih ljudi koji su bliski problemima*

Neko mora da reši sukobljene zahteve iz različitih korisničkih klasa, izmiri nedoslednosti i arbitrira pitanja koja se pojavljuju. Šampioni i proizvoda ili vlasnik proizvoda to mogu rešiti u mnogim, ali verovatno ne u svim slučajevima. Na početku projekta odredite ko će biti donosioci odluka u vezi sa zahtevima. Ako nije jasno ko je odgovoran za donošenje tih odluka ili ako ovlašćeni pojedinci odustanu od svojih odgovornosti, odluke će pasti na programere ili analitičari podrazumevano. Međutim, većina njih nema potrebno znanje i perspektivu za donošenje najboljih poslovnih odluka. Analitičari ponekad se priklanjaju onom koji ima najglasniji glas koji čuju ili osobi koja je najviša u lancu rukovođenja. Iako je razumljivo, to nije najbolja strategija. Odluke treba donositi što niže u hijerarhiji organizacije od strane dobro informisanih ljudi koji su bliski problemima.

Tabela na slici 1 identificuje neke konflikte sa zahtevima koji mogu nastati na projektima i predlaže načine njihovog rešavanja. Lideri projekta treba da odrede ko će odlučiti šta treba raditi kada se pojave takve situacije, ko će uputiti poziv ako se ne postigne dogovor i kome se značajna pitanja moraju uvećavati po potrebi.

Ovi pregovori ne ispadaju uvek onako kako se analitičar može nadati. Određeni kupci mogu odbiti sve pokušaje da razmotre razumne alternative i druga gledišta. Videli smo slučajeve u kojima marketing nikada nije odgovorio "ne" zahtevu kupca, bez obzira koliko neizvodljiv ili skup.

Tim treba da odluči ko će donositi odluke o zahtevima projekta pre nego što se suoči sa ovim vrstama problema. U suprotnom, neodlučnost i revizija prethodnih odluka mogu zaustaviti projekat u beskonačnom sukobu. Ako ste BA koji je uhvaćen u ovoj dilemi, oslonite se na svoju organizacionu strukturu i procese za rešavanje neslaganja. Ali, kao što smo ranije upozorili, nema lakih rešenja ako radite sa zaista nerazumnim ljudima.

Neslaganje između	Kako rešiti
Pojedinih korisnika	Šampion poizvoda ili vlasnik proizvoda odlučuje
Klasa korisnika	Prednost dobija preferirana klasa korisnika
Segemenata tržišta	Prednost dobija segment tržišta koji ima veći efekat na poslovni uspeh organizacije.
Kupaca korporacije	Poslovni ciljevi određuju pravac
Korisnika i menadžera korisnika	Odlučuje vlasnik proizvoda i šampion proizvoda za klasu korisnika
Razvoja i kupca	Kupci imaju prednost, ali ako su u saglasnosti sa poslovnim ciljevima.
Razvoja i marketinga	Marketing ima prednost

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

Slika 5.1 Preporuke za rešavanje sukoba zahteva

## VIDEO 9 - USER CLASSES - WIEGRES (VIDEO)

*Trajanje: 8:06 minuta*

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

## VIDEO 10 - PRODUCT CHAMPIONS / WIEGERS (VIDEO)

*Trajanje: 6:20 minuta*

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

## ▼ Poglavlje 6

### Vežba

#### UČESNICI NA PROJEKTU

*Pojedinci, grupe ili organizacije koje su aktivno uključene u projekat, pogođene njegovim rezultatom ili mogu uticati na njegov rezultat.*

Učesnici su pojedinci, grupe ili organizacije koje su aktivno uključene u projekat, pogođene njegovim rezultatom ili mogu uticati na njegov rezultat. Potrebno je identifikovati kupce ovog proizvoda i druge zainteresovane strane i navesti njihove glavne interese u proizvodu. Okarakterišite kupce na poslovnom nivou, ciljne tržišne segmente i različite korisničke klase, kako biste smanjili verovatnoću da se kasnije pojave neočekivani zahtevi koji se ne mogu prilagoditi zbog ograničenja rasporeda ili obima. Za svaku kategoriju zainteresovanih strana, profil uključuje glavnu vrednost ili koristi koje će dobiti od proizvoda, njihov verovatni odnos prema proizvodu, glavne karakteristike i karakteristike koje vas zanimaju i sva poznata ograničenja koja moraju biti zadovoljena. Primeri vrednosti zainteresovanih strana:

- poboljšana produktivnost
- smanjena prerada
- uštede
- pojednostavljeni poslovni procesi
- automatizacija prethodno ručnih zadataka
- sposobnost izvršavanja potpuno novih zadataka ili funkcija
- usklađenost sa važećim standardima ili propisima
- poboljšana upotrebljivost ili smanjen nivo frustracije u poređenju sa trenutnim aplikacijama

#### POSLOVNI ZAHTEVI INFORMACIONOG SISTEMA PRIVATNE KLINIKE - PROFIL ZAINTERESOVANIH STRANA

*Odnosi se na treće podglavlje u Dokumentu o viziji i okviru, koji nosi naziv Poslovni kontekst*

U tabeli na slici 1 su izloženi profili zainteresovanih strana za razvoj novog poslovnog sistema klinike.

Za svakog stejkholdera je navedena glavna vrednost koju njemu nov sistem donosi, kakav stav imaju prema razvoju novog sistema, šta ih najviše interesuje da nov sistem omogući i na sa kakvim ograničenjima se susreću.

STEJKHOLDERI	GLAVNA VREDNOST	STAVOVI	GLAVNI INTERESI	OGRANIČENJA
<b>MEDICINSKI RADNICI I LEKARI</b>	Automatizacija ručno rađenih aktivnosti  Poboljšana upotrebljivost	Očekuju visoku upotrebljivost, ali su sumnjičavi po pitanju uspešnosti projekta	Lako za učenje; jednostavno za korišćenje	Obuka za korišćenje
<b>PACIJENTI</b>	Smanjen nivo frustracije zbog zadržavanja	Entuzijastični, ali očekuju drastično veću operativnost	Brže obavljanje pregleda; pouzdane arhive informacija	Nisu identifikovana
<b>ADMINISTRATIVNI RADNIK</b>	Brz pristup podacima  Pojednostavljeni poslovni procesi	Imaju razumevanja, ali očekuju visoku upotrebljivost	Mogućnost rada sa mnogo većom bazom podataka nego sa sadašnjim sistemom; lako za učenje	Obuka za korišćenje
<b>UPRAVNI ODBOR</b>	Uštede  Usklađenost sa standardima	Puni ideja i očekivanja, ali oprezni	Bogatiji skup svojstava u odnosu na konkurenčiju	Planiran budžet \$7.000,00
<b>DOBAVLJAČI</b>	Manje grešaka u porudžbenicama  Brži protok informacija	Zabrinuti da nivo saradnje ne opadne zbog uvođenja promena u poslovanju	Tačnost dokumentacije na mnogo višem nivou	Nisu identifikovana
<b>PREDSTAVNICI MARKETINGA</b>	Konkurentkska prednost  Sredstvo privlačenja klijenata	Vide proizvod kao sredstvo povećanja od 15% udelna na tržištu	Bogatiji skup svojstava u odnosu na konkurenčiju	Nije im predviđen direktni pristup sistemu

Slika 6.1 Profili zainteresovanih strana za razvoj sistema privatne klinika

## POSLOVNI ZAHTEVI INFORMACIONOG SISTEMA PRIVATNE KLINIKE - PRIORITETI PROJEKTA

*Odnosi se na treće podglavlje Dokumenta o viziji i okviru, koji nosi naziv Poslovni kontekst*

U ovom poglavlju se javlja prilika da se opišu prioriteti među zahtevima, rasporedom i budžetom projekta. Tabela sa slika 2 može biti od koristi za identifikaciju parametara oko glavnih pokretača projekta (ciljevi najvišeg prioriteta), ograničenja u kojima se radi i dimenzije koje se mogu uravnotežite jedne s drugima kako bi se postigli ciljevi unutar poznatih ograničenja.

DIMENZIJA	POKRETAČ (UNOS CILJEVA)	OGRANIČENJE (UNOS OGRANIČENJA)	STEPEN SLOBODE (UNOS DOZVOLJENOG OPSEGA)
<b>PLAN I ROKOVI</b>	Prvo izdanje treba da bude pušteno u roku od godinu dana od prihvatanja predloga projekta		
<b>SVOJSTVA (OBIM, ZAHTEVI)</b>	Upravni odbor mora da bude uključen u definisanje svojstava budućeg sistema		95% svojstva sa najvećim prioritetom moraju se uključiti u izdanje 1
<b>KVALITET</b>			Testovi prihvatanja od strane kupca moraju da prođu 95%
<b>ČLANOVI TIMA</b>		Nema ograničenja, dokle god se to uklapa u budžet	
<b>TROŠAK</b>		Planiran budžet \$7.000,00	Prihvatljivo probijanje budžeta za 5%

Slika 6.2 Prioriteti projekta razvoja poslovnog softvera za privatnu kliniku

## PITANJA ZA DISKUSIJU

*Tekst pitanja za diskusiju*

### **PITANJE 1.**

Da li razumete razliku između korisnika proizvoda, kupca, vlasnika i interesnih grupa? Predstavite je opisno ili vizuelno.

Takođe, koga vidite kao donosioca odluka? (5 min)

### **PITANJE 2.**

Ko su favorizovane klase korisnika na projektu razvoja poslovnog sistema klinike? Ko su ignorisane klase korisnika. (5 min)

## ZADACI ZA VEŽBU

*Tekst zadataka za vežbu*

### **ZADATAK 1.**

Na osnovu stejkholdera sistema privatne klinike, koji su razmatrani u prethodnim vežbama, odaberite stvarne aktere (učesnike, direktnе korisnike) sistema privatne klinike. Za svakog od tih korisnika, uradite personalizaciju. (15 min)

### **ZADATAK 1.**

Na primeru sistema za upravljanje radom taksi udruženja koji je opisan u prvoj vežbi, uraditi sledeće:

- Identifikujte različite klase korisnika i kreirati profile zainteresovanih strana (10 min)

- Odredite favorizovanu klasu korisnika (ili klase, ako ih ima više) (5 min)
- Odredite prioritete projekta (10 min)

## ZADATAK 2.

Na primeru modula E-Student, uraditi sledeće:

- Identifikujte različite klase korisnika i kreirati profile zainteresovanih strana
- Odredite prioritete projekta

## ✓ Poglavlje 7

### Domaći zadatak

#### DOMAĆI ZADATAK 4

##### *Tekst domaćeg zadatka*

Za sistem koji ste dobili za DZ01 identifikujte različite klase korisnika, odredite koje su favorizovane i na osnovu toga kreirajte profile zainteresovanih strana.

*Napomene:*

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

### ZADATAK ZA RAD NA PROJEKTU

*Tekst zadatka za rad na projektu*

U kreiranom **Dokument o viziji i okviru** dodajte treće poglavlje.

## ✓ Poglavlje 9

### Zaključak

## ZAKLJUČAK

1. Klasa korisnika je podklasa korisnika proizvoda, koji je podskup kupaca proizvoda, a koji je podskup njegovih interesnih grupa.
2. Pri identifikaciji korisničkih klasa razmišljajte o zadacima koje će različiti korisnici obavljati sa sistemom.
3. Svaka klasa korisnika će imati svoj vlastiti skup zahteva za zadatke koje pripadnici klase moraju da obavljaju.
4. Prepoznajte i okarakterišite različite klase korisnika za svoj proizvod već na početku projekta.
5. Dokumentujte klase korisnika i njihove karakteristike, odgovornosti i fizičke lokacije u specifikaciji softverskih zahteva (SRS) ili u planu zahteva za svoj projekat.
6. Persona je opis hipotetičke, generičke osobe koja služi tipičan reprezentant grupe korisnika koja imaju slične karakteristike i potrebe.
7. Predstavnici korisnika pružaju "glas korisnika".
8. Šampion proizvoda služi kao primarni interfejs između članova jedne korisničke klase i poslovnog analitičara projekta.
9. Ako imate raznovrsnu korisničku bazu, prvo identifikujte osnovne zahteve koji su zajednički za sve kupce. Zatim definišite dodatne zahteve koji su specifični za pojedinačne korporativne kupce.
10. Koristite datu tabelu aktivnosti šampiona kao polaznu tačku za pregovaranje o obavezama svakog šampiona.
11. Podsetite menadžment da je šampion proizvoda ključni saradnik koji može da pomogne projektu da postigne svoje poslovne ciljeve.
12. Model šampiona proizvoda uspeo je u mnogim okruženjima. Deluje samo kada šampioni razumeju svoje odgovornosti, imaju autoritet da donose odluke na nivo potreba korisnika.
13. Vlasnik proizvoda definiše viziju proizvoda i odgovoran je za razvoj i određivanje prioriteta sadržaja backlog-a (zaostalih proizvoda).
14. "Nema zamene za to da imate prave ljude, u pravoj ulozi, na pravom mestu, sa pravim stavom."
15. Pri sukobu zahteva odluke treba donositi što niže u hijerarhiji organizacije od strane dobro informisanih ljudi koji su bliski problemima.

## REFERENCE

Nastavi 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 6** 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.



## SE322 - INŽENJERSTVO ZAHTEVA

Prikupljanje zahteva

Lekcija 05

PRIRUČNIK ZA STUDENTE

# SE322 - INŽENJERSTVO ZAHTEVA

## Lekcija 05

### **PRIKUPLJANJE ZAHTEVA**

- ✓ Prikupljanje zahteva
- ✓ Poglavlje 1: Tehnike utvrđivanja zahteva
- ✓ Poglavlje 2: Planiranje za izazivanje zahteva
- ✓ Poglavlje 3: Priprema za izazivanje zahteva
- ✓ Poglavlje 4: Izvršenje aktivnosti prikupljanja zahteva
- ✓ Poglavlje 5: Aktivnosti posle izazivanja zahteva
- ✓ Poglavlje 6: Klasifikovanje predloga i stavova kupaca
- ✓ Poglavlje 7: Još o zahtevima
- ✓ Poglavlje 8: Vežba
- ✓ Poglavlje 9: Domaći zadatak
- ✓ Poglavlje 10: Projektni zadatak
- ✓ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

## ❖ Uvod

### UVOD

*Izazivanje zahteva nije isto što je i prikupljanje zahteva. Izazivanje je kolaborativan i analitičan proces koji uključuje aktivnosti prikupljanja, otkriva*

Izazivanje zahtava (*requirements elicitation*) je srž razvoja zahteva, procesa utvrđivanja potreba i ograničenja različitih aktera projekta razvoja softvera. Izazivanje zahteva nije isto što je i prikupljanje (*gathering*) zahteva. Izazivanje je kolaborativan i analitičan proces koji uključuje aktivnosti prikupljanja, otkrivanja, izvlačenja i definisanja zahteva. Nama je stran termin "izazivanje zahteva". Umesto da koristimo, što neki rade, i ubacimo u naš žargon reč "elicitacija", koja je još manje razumljiva od reči "izazivanje", mi ćemo koristiti termin "izazivanje zahteva", a ne što nam prvo pada na pamet - "prikupljanje zahteva". Kao što smo naveli, **izazivanje zahteva** jeste prikupljanje zahteva, ali i mnogo više od toga. Pre nego što ih prikupimo (preuzmemos), moramo da ih otkrijemos, da vidimo gde da ih nađemo. Onda moramo i da ih izvučemos - jer korisnici ih implicitno koriste, ali ih eksplisitno često ne navode ("podrazumevaju se"). Nigde na nekoj tabli i papiru nećete naći ispisane sve zahteve korisnika. I na kraju treba da ih formalno definišemos kako bi bili jasni i precizni. Kao što vidite, pojam "izazivanje zahteva" je dosta složen i širi od pojma "prikupljanja zahteva", te treba sada da se naviknemo na upotrebu termina "izazivanje zahteva" (*requirements elicitation*). Da ne bi unosili odmah zabunu, do sada smo u prethodnim lekcijama koristili termin "prikupljanje zahteva", ali smo u stvari mislili na "izazivanje zahteva". U ovoj lekciji ćemo detaljno objasniti šta sve to znači.

Izazivanje zahteva se koristi radi otkrivanja poslovnih, korisničkih, funkcionalnih i nefunkcionalnih zahteva, zajedno sa ostalim informacijama. Izazivanje zahteva je možda najizazovniji, kritičan i komunikaciono intenzivan aspekt razvoja softvera.

U toku procesa izazivanja zahteva, vi treba i da pridobijete podršku korisnika za kasnije prihvatanje softverskog proizvoda. Pratite procese rada i donošenja odluka korisnika i izvucite iz toga logiku njihovog rada i odlučivanja. Pokušajte da svi shvate zašto sistem treba da ima određene funkcije. One zahteve koji su rezultat loših i neefektivnih sadašnjih poslovnih procesa nemojte uzimati u obzir.

Kao analitičar sistema, koristite terminologiju domena, ne namećite vašu i stručnu terminologiju. Napravite rečnik termina koje korisnici koriste i definišite ih kako bi sa korisnicima uspostavili saglasnost oko jezika koji ćete koristiti i kako bi se bolje razumeli. Kada razumete potrebne korisnika, onda možete razmišljati o alternativnim načinima za njihovo zadovoljenje. U procesu izazivanja zahteva nemojte se opredeljivati za projektna rešenja novog sistema. Isuviše je rano za to. Prvo morate da dobro razumete sistem. U suprotnom, može se desiti da projektna rešenja morate da menjate kada saznate nove zahteve, a to znači više rada. Usredsreditte potrebe korisnika na njihove zadatke, a manje na njihove želje.

## UVODNI VIDEO

*Trajanje video snimka: 3min 11sek*

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

## ✓ Poglavlje 1

### Tehnike utvrđivanja zahteva

#### VIDEO PREDAVANJE ZA OBJEKAT "TEHNIKE UTVRĐIVANJA ZAHTEVA"

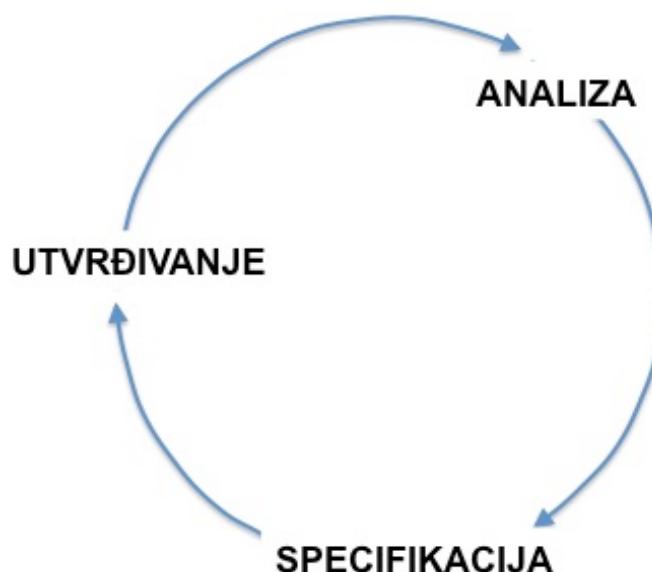
*Trajanje video snimka: 21min 52sek*

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

#### CIKLIČNI NAČIN RAZVOJA ZAHTEVA

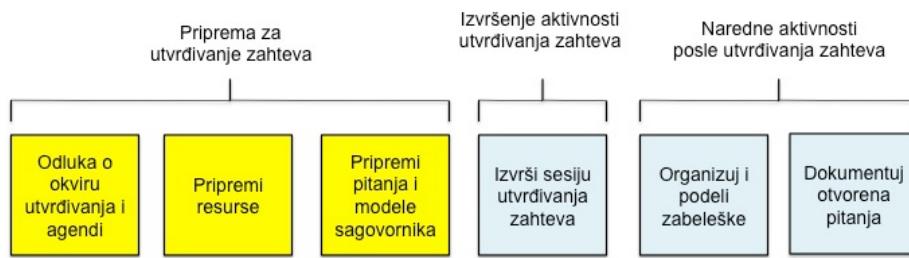
*Vi prvo vršite utvrđivanje zahteva, analizirate šta ste naučili i onda zapisujete zahteve.*

Slika 1 prikazuje cikličan način razvoja zahteva. Vi prvo vršite utvrđivanje zahteva, analizirate šta ste naučili i onda zapisujete neke zahteve. Tada i utvrđujete šta vam od informacija nedostaje i onda vršite dodatno utvrđivanje zahteva i tako se ciklus utvrđivanje-analiza-specifikacija zahteva ponavlja. To je težak posao koji traje. Ne očekujte da ćete posle par radionica završiti proces utvrđivanja zahteva, tj. da ćete ih prikupiti, otkriti, izvući i definisati. Morate to uraditi primenom više različitih tehnika utvrđivanja zahteva.



Slika 1.1 Cikličan razvoj zahteva

U narednim lekcijama ćemo izučavati faze analize i specifikacije u procesu razvoja zahteva. U ovoj lekciji ćemo se isključivo baviti fazom utvrđivanja zahteva. Slika 2 prikazuje jednu tipičnu sesiju utvrđivanja zahteva koja sadrži više aktivnosti koje se koriste pri izazivanju zahteva.



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

Slika 1.2 Aktivnosti jedne sesije izazivanja zahteva

Sve ove aktivnosti biće izučavane u ovoj lekciji, ali pre toga, navećemo tehnike koje se koriste u procesu utvrđivanja zahteva.

Pošto se prikupljaju različite informacije tokom sesije izazivanja zahteva, ne primenjuje se samo jedna, već više tehnika izazivanja zahteva, zbog specifičnosti informacija koje se prikupljaju.

## TEHNIKE UTVRĐIVANJA ZAHTEVA - INTERVJU

*Ustvarite odnos, držite se okvira, pripremite pitanja, sugerirajte ideje, aktivno slušajte.*

Tehnike utvrđivanja zahteva (elicitation techniques) se koriste:

- **olakšane tehnike**, u kojima ste vi u interakciji sa akterima, i
- **nezavisne tehnike**, u kojima vi samostalno radite otkrivanja informacija.

Olakšane tehnike (facilitated techniques) prvenstveno su usmerene ka otkrivanju poslovnih i korisničkih zahteva. Ovde je neophodan direktni rad sa korisnicima jer zahtevi korisnika obuhvataju zadatke koje korisnici moraju da izvrše sa sistemom. Da bi izazvali poslovne zahteve, treba da razgovarate prvenstveno sa sponzorima projekta.

Nezavisne tehnike utvrđivanja zahteva (independent elicitation techniques) dopunjuju olakšane tehnike jer se otkrivaju potrebe i zahtevi kojih korisnici možda ni svesni.

U praksi, kombinujemo obe vrsta tehnika

**Intervju** Vode se individualni ili intervjuji sa malim grupama korisnika. Jednostavnije ih je organizovati od radionica, koje su rad sa većom grupom korisnika. Intervjuji se dobri za izazivanje poslovnih zahteva od rukovodioca koji imaju malo vremena za razgovor. Ovde se daju neki saveti za vođenje intervjeta:

- **Ustvarite odnos:** Predstavite se, upoznajte sagovornika sa sadržajem razgovora, ukažite mu na ciljeve razgovora i pitajte da li on ima neka pitanja.
- **Držite se okvira:** Usmeravajte razgovor ka njegovim ciljevima.

- **Pripremite pitanja i model sagovornika:** Pripremite se za razgovor pripremom pitanja i možda neke konstatacije. To daje dobar početak razgovora.
- **Sugerište ideje:** Na osnovu odgovora koje dobijete, kreirajte ideje i alternative za vreme izazivanja zahteva. Ako se korisnik teško izražava o svojim potrebama, onda idite na tehniku osmatranja njegovog rada.
- **Aktivno slušajte:** Nagnite se napred, pokažite strpljenje, verbalno reagujte kada treba i pitajte kada vam nešto nije jasno. Rezimirajte šta mislite da je korisnik rekao, ako vam nije najjasnije, da bi sa njim to usaglasili.

Možete snimati razgovor da bi preslušali nejasne delove razgovora, a i da bi imali i dokaz šta vam je sagovornik rekao, mada je njegova potvrda beleške merodavnija. Odmah posle intervjuja, dok vam sećanje ne izbledi, sačinite belešku sa intervjuja i dostavite osobi sa kojom ste razgovarali, da bi proverili da li ste njene stavove i odgovore pravilno protumačili i zapisali. Tražite i da sagovornik to potvrdi potpisom na dostavljen dokument - beleška sa razgovora.

## TEHNIKE IZAZIVANJA ZAHTEVA - FOKUS GRUPE, POSMATRANJA I UPITNICI

*Fokus grupa je reprezentativna grupa korisnika koja se saziva radi usmeravane aktivnosti izazivanja zahteva. Posmatranjem otkrivate neizrečene zahteve.*

**Fokus grupa** Fokus grupa (*focus group*) je reprezentativna grupa korisnika koja se saziva radi usmeravane aktivnosti izazivanja zahteva a radi davanja predloga i ideja za funkcionalne zahteve i zahteve kvaliteta. One su interaktivne i očekuje se da svi učesnici govore. Fokus grupe su zgodne za izražavanje stavova korisnika, impresija, preferencija, i potreba. Važne su pri razvoju komercijalnog proizvoda kada nemate direktnе buduće korisnike.

Izbor članova grupe treba pažljivo uraditi. Ili ćete držati jednu grupu sa predstavnicima svih klasa korisnika, ili ćete držati više fokus grupe, sa svakom klasom korisnika posebno.

Radom fokus grupe se rukovodi, ali na način da svako može slobodno da govori. Možete snimati diskusiju. One daju subjektivna mišljenja koja kasnije morate da analizirate i da im određujete prioritet. Učesnici fokus grupe obično ne donose nikakve odluke.

**Posmatranja** Neki učesnici iz raznih razloga ne iskazuju sve što znaju i žele. Zato ih treba posmatrati dok rade. Posmatranja traju te ih treba pažljivo koristiti. Da ne bi ometali u radu ljudi, ograničite posmatranja na najviše dva sata. Izaberite važne i visokorizične zadatke za posmatranje.

Osmatranja mogu biti tiha (kada se korisnici ne mogu uznemiravati dok rade), a mogu biti i interaktivna. (kada je to moguće radi razjašnjavanja zadataka koji radi). Dokumentujte vaša opažanja tokom posmatranja. Možete i snimiti rad zaposlenog, ako je to dozvoljeno.

**Upitnici** Upitnici (*questionairs*) su jeftini i dobri u slučaju velikog broja ispitanika. Njavažnije je dobro pripremiti pitanja. Evo nekih preporuka:

- Ponudite odgovore koji pokrivaju sve opcije mogućih odgovora.

- Ponuđeni odgovori ne smeju da budu međusobno isključivi.
- Ne postavljajte pitanja na koje se može samo potvrđno odgovoriti.
- Ako koristite ocene, onda ih koristite u svim pitanjima.
- Koristite pitanja sa mogućim odgovorima ako rezultat ide u statističku obradu i analizu.
- Možete koristiti savete stručnjaka za pripremu pitanja, kako bi postavili prava pitanja pravim ljudima.
- Uvek testirajte pitanja pre nego što ih distribuirate.
- Ne pitajte mnogo, ako želite da ljudi odgovore na upitnik.

## TEHNIKE UTVRĐIVANJA ZAHTEVA - ANALIZA INTERFEJSA SISTEMA

*Analiza interfejsa sistema, interfejsa sa korisnicima i relevantne dokumentacije može ubrazati izazivanje zahteva .*

### **Analiza interfejsa sistema**

**Analiza interfejsa sistema** (system interface analysis) ispituje sistem sa kojim je vaš sistem povezan. Analiza interfejsa sistema otkriva funkcionalne zahteve u vezi sa podacima i servisima koje sistem razmenjuje sa drugim sistemima. Dijagrami konteksta i mape ekosistema su dobra priprema za ovu analizu. Ako pronađete neki interfejs koji nije predstavljen u ovim dijagramima, onda su oni nekompletni.

Za svaki sistem sa kojim je vaš sistem u vezi, utvrdite funkcionalnost drugog sistema koji može da dovodi do zahteva u vašem sistemu. Ti zahtevi mogu da definišu podatke koje treba poslati u drugi sistem, koje treba preuzeti, i pravila vezana za ove podatke, kao što su pravila provere podataka. Možete i otkriti da imate funkcionalnost koja vam nije potrebna.

### **Analiza interfejsa sa korisnikom**

**Analiza interfejsa sa korisnikom** (user interface analysis) služi za otkrivanje zahteva korisnika i funkcionalnih zahteva. Ako se radi o novom sistemu, analizirajte korisničke interfejse sličnih sistema. Ako sistem postoji, možete snimiti sve izglede ekrana u okviru njegovog korisničkog dizajna.

Analiza interfejsa može da dovede do liste svih svojstava softvera. Takođe, možete utvrditi koje slučajeve korišćenja podržava, kao i podatke koje korisnik mora da unese ili da vidi. Na ovaj način, umesto da pitate korisnike kako rade sa postojećim sistemom, možete lično utvrditi.

### **Analiza dokumenata**

**Analiza dokumenata** (document analysis) podrazumeva analizu sve postojeće dokumentacije koja može da pomogne utvrđivanju softverskih zahteva. Najkorisnija dokumenta su: specifikacija zahteva, poslovni procesi, zbirke naučenih lekcija i korisnička uputstva za postojeće a slične sisteme. Dokumenta mogu da opisuju i industrijske standarde ili interne standarde organizacije ili dokumenta sa propisima koje sistem mora da poštue.

Možete analizirati dokumenta koja daju upoređivanje sličnih proizvoda, zatim izveštaje o primedbama korisnika ili zahteve za poboljšanje sistema.

Analiza dokumenata omogućava ubrzavanje razvoja novog sistema ili poboljšanje starog. Što više se upoznate sa starim sistemom, ili sa primedbama korisnika starog sistema, brže ćete razviti zahteve za novi sistem. Rizik sa ovom analizom je upotreba neažurne dokumentacije.

## TEHNIKE UTVRĐIVANJA ZAHTEVA - RADIONICE

*Radionica je strukturisan sastanak na kome grupa aktera i eksperata rade zajedno na definisanju, kreiranju, poboljšanju i postizanju saglasnosti o zahtevima korisnika.*

Radionice podržavaju učešće korisnika u definisanju zahteva. **Radionica** (*workshop*) je strukturisan sastanak na kome pažljivo izabrana grupa aktera i eksperata domenu rade zajedno na definisanju, kreiranju, poboljšanju i postizanju saglasnosti o rezultatima (kao što su modeli ili dokumenti) koji predstavljaju zahteve korisnika. Radionice obično sadrže nekoliko tipova aktera, od korisnika do inženjera razvoja i testera. Rad u grupi je efikasniji u rešavanju konfliktnih situacija, nego individualni rad. Takođe, primena radionica ubrzava proces izazivanja zahteva.

Kritičnu ulogu u radu radionice ima **fasilitator** koji vodi učesnike ka uspešnim rezultatima. Dobra je praksa da pomoćnik biznis analitičara igra ulogu fasilitatora, kako bi se biznis analitičar usredsredio da sluša i izvlači ideje i stavove učesnika radionice.

Radionice treba dobro planirati da se ne bi gubilo vreme, jer mogu da traju i nekoliko dana. Na primer, možete unapred pripremiti nacrt slučajeva korišćenja i diskutovati ih na radionici, umesto da ih svi grupno kreirate za vreme rada radionice. Time se štedi vreme. Upotrebite druge tehnike izazivanja zahteva da pripremite radionicu, tako da se radionica bavi samo određenim oblastima, kada pokazuje svoju efikasnost.

Ovde ćemo dati neke savete za efektivno vođenje i olakšavanje rada radionice:

- **Postavite osnovna pravila:** Učesnici treba da se slože oko nekoliko principa rada. Na primer, vreme početka i kraja, povratak sa pauza u zakazano vreme, isključenje mobilnih uređaja, očekivanje da svako aktivno učestvuje, i usmerenje primedbi na sadržaj a ne na pojedince.
- **Popunite sve uloge članova tima:** Vođenje beleške, vođenje računa o vremenu, upravljanje okvirom, upravljanje osnovnim pravilima, i omogućavanje svakom da bude saslušan.
- **Planirajte agendu:** Unapred obavestite učesnike o planu rada radionice i agendi.
- **Držite se okvira:** Proverite da su predlozi u skladu sa poslovnim zahtevima. Držite potreban nivo apstraktnosti diskusija.
- **Zapisivanje ideja:** Na papirićima zapишite ideje za kasnije razmatranje.
- **Vremensko ograničenje diskusija:** Za svaku temu odredite vreme za razmatranje. Na kraju sumirajte diskusiju i pređite na sledeću temu.
- **Napravite malu grupu, ali sa pravim akterima:** Optimum je 5'6 učesnika. Vidite paralelne radionice sa različitim klasama korisnika.

- **Neka svako bude angažovan:** Neki učesnici iz raznih razloga ne učestvuju u diskusijama. Fasilitator treba da ih angažuje. Možete direktno pitati čutljive učesnike šta misle o vođenoj diskusiji. Fasilitator treba da obezbedi da se svako čuje i da može da se izrazi.

## ✓ Poglavlje 2

# Planiranje za izazivanje zahteva

## PLAN UTVRĐIVANJA ZAHTEVA

*Plan izazivanja uključuje tehnike koje koristite, kada planirate da ih koristite i sa kojom svrhom.*

Na početku projekta, poslovni analitičar treba da isplanira pristup projektu zahtevima. Čak i jednostavan plan akcije povećava šansu za uspeh i postavlja realna očekivanja zainteresovane strane. Samo ako ste izričito zauzeti za resurse, raspored i rezultate, možete izbeći da se učesnici povuku da rade druge poslove. Plan privlačenja uključuje tehnike koje ćete koristiti, kada ih planirate da koristite i u koje svrhe. Kao i bilo koji plan, koristite ga kao vodič i podsetnik tokom celog projekta, ali shvatite da ćete možda morati da menjate plan tokom celog projekta. Vaš plan treba da sadrži sledeće stavke:

- **Ciljevi za pronalaženje** Planirajte ciljeve za prikupljanje zahteva za ceo projekat i ciljeve za svaku planiranu aktivnost izvlačenja.
- **Strategija za izvlačenje i planirane tehnike** Odlučite koje ćete tehnike koristiti sa različitim interesnim grupama. Možete koristiti neku kombinaciju upitnika, radionica, poseta kupcima, individualnih intervjuja i drugih tehnika, u zavisnosti od pristupa koji imate zainteresovanim stranama, vremenskih ograničenja i vašeg znanja o postojećem sistemu.
- **Procena rasporeda i resursa** Identifikujte i kupca i učesnike u razvoju za različite aktivnosti prikupljanja zahteva, zajedno sa procenama potrebnog napora i kalendarskog vremena. Možda ćete moći samo da identifikujete korisničke klase, a ne neke konkretnе pojedince, ali to će menadžerima omogućiti da započnu planiranje nadolazećih potreba za resursima. Procenite vreme BA, uključujući vreme za pripremu za izuzeće i za obavljanje naknadnih analiza.
- **Dokumenti i sistemi potrebni za nezavisno prikupljanje podataka** Ako sprovodite analizu dokumenata, sistemskog interfejsa ili korisničkog interfejsa, identifikujte materijale potrebne da biste ih osigurali kada su vam potrebni.
- **Rad na očekivanom prikupljanju zahteva** Znajući da ćete kreirati listu slučajeva upotrebe, SRS, analize rezultata upitnika ili specifikacije kvaliteta, pomažu vam da ciljate na prave aktere projekta, teme i detalje tokom iznošenja.
- **Rizici utvrđivanja zahteva:** Identifikujte faktore koji bi mogli da ometu vašu sposobnost da dovršite aktivnosti utvrđivanja zahteva kako je planirano. Procenite ozbiljnost svakog rizika i odlučite kako ga možete ublažiti ili kontrolisati.

## PREPORUČENE TEHNIKE ZA UTVRĐIVANJE ZAHTEVA

*Izaberite nekoliko tehnika utvrđivanja zahteva u skladu sa specifičnostima vašeg projekta.*

Mnogi BA-i imaju svoju tehniku vođenja „uobičajenih intervjeta i radionica“ i ne misle da koriste druge tehnike koje mogu umanjiti potrebe za resursima ili povećati kvalitet otkrivenih informacija. Korisnici i akteri najčešće ne mogu direktno da vam kažu eksplizitno svoje zahteve. Zbog toga, utvrđivanje zahteva se svodi na izazivanje i izvlačenje zahteva.

Retko će BA dobiti najbolje rezultate koristeći samo jednu tehniku utvrđivanja zahteva na projektu. Tehnike utvrđivanja zahteva primenjuju se u čitavom spektru razvojnih pristupa. Izbor tehnika utvrđivanja zahteva treba da se zasniva na karakteristikama projekta.

Na slici 1 predložene su tehnike zahteva utvrđivanja koje će najverovatnije biti korisne za razne vrste projekata. Izaberite red ili redove koji predstavljaju karakteristike vašeg projekta i pročitajte s desne strane da biste videli koje tehnike utvrđivanja zahteva mogu biti najviše korisne (označene sa X). Na primer, ako razvijate novu aplikaciju, verovatno ćete dobiti najbolje rezultate kombinacijom intervjeta sa zainteresovanim stranama, radionicama i analizom sistemskog interfejsa. U većini projekata mogu se koristiti intervjeti i radionice. Fokus grupe su pogodnije od radonica softvera za masovno tržište jer imate veliku spoljnu korisničku bazu, ali ograničen pristup predstavnicima. Ovi predlozi za tehnike utvrđivanje zahteva su upravo to - predlozi. Na primer, mogli biste zaključiti da želite da примените analizu korisničkog interfejsa na softverskim projektima za masovno tržište.

	Intervjeti	Radionice	Fokus grupe	Posmatranja	Upitnici	Analiza interfejsa sistema	Analiza interfejsa korisnika	Analiza dokumenata
Softver namenjen tržištu	x		x		x			
Interni softver organizacije	x	x	x	x		x		x
Zamena postojećeg sistema	x	x		x		x	x	x
Poboljašnje postojećeg sistema	x	x				x	x	x
Nova aplikacija	x	x				x		
Upakovano softversko rešenje	x	x		x		x		x
Ugrađen sistem	x	x				x		x
Geografski distribuirani akteri	x	x			x			

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

Slika 2.1 Preporučene tehnike izazivanja zahteva po kategorijama softvera

## ✓ Poglavlje 3

# Priprema za izazivanje zahteva

## KAKO SE PRIPREMITI?

*Planirajte okvir sesije i agendu. Pripremite resurse. Pripremite pitanja o modelu sagovornika.*

Sesije utvrđivanja zahteva zahtevaju pripreme kako bi najbolje iskoristili nečije vreme. Što je veća grupa koja učestvuje u sesiji, važnija je priprema. Na slici 1 prikazane su aktivnosti za pripremu jedinstvene sesije ispunjavanja zahteva..



Slika 3.1 Tri aktivnosti priprema za izazivanje zahteva softvera

Pripremite pitanja i izradite materijale koji bi mogli biti korisni tokom sesije. Sledeći saveti će vam pomoći da se pripremitе za utvrđivanje zahteva.

**Planirajte okvir i dnevni red sednice** Odlučite o okviru sesije za utvrđivanju zahteva, uzimajući u obzir koliko je vremena na raspolaganju. Okvir sesije možete definisati upotrebom skupa tema ili pitanja ili možete navesti određeni skup tokova procesa ili koristiti slučajeve koje treba istražiti. Uskladite okvir sesije sa celokupnim obimom projekta definisanim u poslovnim zahtevima, tako da možete voditi razgovor o temi.

Na dnevnom redu bi trebalo da bude precizirano koje će teme biti obuhvaćene, raspoloživo vreme za svaku temu i ciljevi. Unapred podelite dnevni red sednice sa akterima, tj. učesnicima sednice.

**Pripremite resurse** Zakažite potrebne fizičke resurse, kao što su sobe, projektori, brojevi telekonferencija i oprema za video konferencije. Takođe, zakažite učesnike, osetljivi na razlike u vremenskoj zoni ako niste svi na istoj lokaciji. Za geografski raštrkane grupe menjajte raspored svaki put kada se sastanete kako seanse ne bi uvek dovele do neugodnosti kod istih ljudi u određenom delu sveta. Prikupite dokumentaciju iz različitih izvora. Po potrebi steknite pristup sistemima. Krenite na obuku putem interneta da biste saznali više o postojećim sistemima.

**Saznajte više o akterima** Prepoznajte relevantne aktere za sesiju. Saznajte o kulturnim i regionalnim preferencijama zainteresovanih strana za sastanke. Ako neki od učesnika nisu izvorni govornici jezika na kojem će sesija biti održana, razmislite o tome da im obezbedite prateću dokumentaciju, poput slajdova, pre vremena kako bi mogli da čitaju unapred ili prate dalje. U slajdovima se mogu navesti konkretna pitanja koja ćete postavljati ili jednostavno dati kontekst za sesiju koji biste takođe mogli usmeno objasniti. Izbegavajte stvaranje „nas“ nasuprot „njima“ napetosti.

## PRIPREMA PITANJA

*Uđite u svaku sesiju utvrđivanja zahetva, sa setom pripremljenih pitanja.*

**Priprema pitanja** Uđite u svaku sesiju utvrđivanja zahteva, sa setom pripremljenih pitanja. Koristite područja nesigurnosti u modelima UML aktera kao izvor pitanja. Ako se pripremate za intervju ili radionicu, koristite rezultate drugih tehnika iznošenja da biste identifikovali nerešena pitanja. Postoje brojni izvori predloženih pitanja za utvrđivanje zahteva.

Izrazite svoja pitanja kako biste izbegli da kupce vodi nemernim putem ili ka određenom odgovoru. Kao analitičar morate da proverite ispod površine zahteva koje postavljaju kupci da bi razumeli njihove istinske potrebe. Jednostavno pitajući korisnike, „Šta želite?“ Generiše masu slučajnih informacija koje ostavljaju analitičare koji lebdi. „Šta treba da uradite?“ Je mnogo bolje pitanje. Pitajući „zašto“ nekoliko puta može prebaciti diskusiju sa predstavljenog rešenja na čvrsto razumevanje problema koji treba rešiti. Postavljajte otvorena pitanja koja će vam pomoći da razumete trenutne poslovne procese korisnika i da vidite kako bi novi sistem mogao da poboljša njihove performanse.

Zamislite da učite korisnikov posao ili ga zapravo radite pod uputstvom korisnika. Koje biste zadatke obavljali? Koja pitanja biste imali? Drugi pristup je igrati ulogu naučnika koji uči od glavnog korisnika. Tada korisnik s kojim razgovorate vodi diskusiju i opisuje šta on smatra važnim temama za diskusiju.

**Ispitati oko izuzetaka.** Šta može sprečiti korisnika da uspešno ispuni zadatak? Kako sistem treba da reaguje na različite uslove greške? Postavljajte pitanja koja započinju sa „Šta bi drugo moglo. . . ,“ „Šta se dogodi kada . . . ,“ „Da li bi ti ikad trebalo. . . ,“ „Gde ćete dobiti. . . ,“ „Zašto (ili ne). . . ,“ „I“ Da li neko ikad. . . “Dokumentirajte izvor svakog zahteva tako da možete dobiti dodatna pojašnjenja ako je potrebno i pratiti razvojne aktivnosti do određenog porekla kupca.

Kao i kod svake aktivnosti na poboljšanju, nezadovoljstvo trenutnom situacijom daje odličnu hranu za novu i poboljšanu buduću državu. Kada radite na projektu zamene za postojeći sistem, pitajte korisnike: „Koje tri stvari vas najviše nerviraju u odnosu na postojeći sistem?“ Ovo pitanje pokriva očekivanja koja korisnici imaju za sistem praćenja. Nećete imati - niti vam treba - savršeni scenario koji ide u intervju ili radionicu.

Pripremljena pitanja će vam pomoći ako se zaglavite. Pitanja bi trebalo da izgledaju prirodno i prijatno - poput razgovora, a ne ispitivanja. Pet minuta nakon sesije, mogli biste shvatiti da ste propustili važno područje za diskusiju. Budite spremni da napustite svoja pitanja ako

je potrebno. Na kraju sesije, pitajte „Postoji li još nešto što ste očekivali da vas pitam?“ Da pokušam da iznesem probleme na koje jednostavno niste razmišljali.

## PRIMENA I UML MODELA AKTERA

*UML model aktera služi kao polazna osnova koja vam pomaže da saznate više o temi i podstakne korisnike da razmišljaju o idejama.*

**Pripremite UML model aktera:** Modeli analiza mogu se koristiti tokom sesija za utvrđivanje zahteva kako bi korisnicima pomogli da obezbede bolje zahteve. Neki od najkorisnijih modela su slučajevi upotrebe i tokovi procesa jer se usko usklađuju s načinom na koji ljudi razmišljaju o obavljanju svog posla. Kreirajte UML modele slučajeva korišćenja ili nacrte, pre nego što započnete sesije. UML model aktera služi kao polazna osnova koja vam pomaže da saznate više o temi i podstakne korisnike da razmišljaju o idejama. Lakše je revidirati nacrt modela nego stvoriti ga ispočetka.

Ako ste novi u domenu projekta, možda će biti teško da sami napravite nacrt modela. Pomoću drugih tehnika izvlačenja, izvucite dovoljno znanja za rad. Pročitajte postojeće dokumente, proučite postojeće sisteme za modele koje možete ponovo da koristite kao polaznu tačku ili vodite razgovor pojedinac sa stručnjakom za teme kako biste naučili dovoljno za početak. Zatim recite grupi sa kojom radite: „Ovaj model je možda pogrešan. Molim vas, kritikujte ga i nađite mu slabe strane, ali mi recite kako bi trebalo da izgleda. Meni to ne smeta.“

## ▼ Poglavlje 4

# Izvršenje aktivnosti prikupljanja zahteva

## VIDEO PREDAVANJE ZA OBJEKAT "IZVRŠENJE AKTIVNOSTI PRIKUPLJANJA ZAHTEVA"

*Trajanje video snimka: 24min 49sek*

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

## AKTIVNOST IZVRŠENJA PRIKUPLJANJA ZAHTEVA

*Obrazujte aktere. Dobro vodite beleške. Iskoristite fizički prostor za sesiju.*

Slika 1 prikazuje aktivnost u kojoj se realizuje izazivanje zahteva u okviru jedne sesije izazivanja.



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

Slika 4.1 Izvršenje aktivnosti utvrđivanja zahteva u okviru jedne sesije izazivanja

Izvođenje same aktivnosti utvrđivanja zahteva relativno je očigledno - ako intervjuјete, razgovirate sa ljudima; ako vršite analizu dokumenta, pročitajte dokument. Sledeći saveti mogu biti korisni u sledećim savetima.

**Obučite zainteresovane strane** Učite svoje zainteresovane strane o vašem pristupu iznošenju i zašto ste ga izabrali. Objasnite tehnike istraživanja koje ćete koristiti, kao što su slučajevi upotrebe ili tokovi procesa, i kako oni mogu pomoći akterima da obezbede bolje zahteve. Takođe opišite kako ćete koristiti njihove podatke i poslati im materijale na pregled nakon sesije.

**Vodite dobre beleške** Dodelite nekoga ko aktivno ne učestvuje u raspravi za pisca, odgovornog za tačne beleške. Sednice treba da sadrže listu polaznika, pozive koji nisu prisustvovali, donesene odluke, akcije koje treba poduzeti i ko je odgovoran za svako, nerešena pitanja i najvažnije tačke ključnih diskusija. Nažalost, BA-i ponekad održavaju sesije za olakšano izazivanje bez posebnog pisca i sami moraju popuniti ulogu. Ako ste u ovoj situaciji, budite spremni da pišete stenografijom, brzo kucate ili koristite uređaj za snimanje (ukoliko se učesnici slažu). Audio olovke mogu prevesti rukom pisane beleške u elektronski oblik i vezati ih za snimljeni audio razgovor. Takođe možete koristiti bele table i papiре na zidovima i fotografisati ih.

Unapred pripremite pitanja za uklanjanje nekih razmišljanja na licu mesta, potrebnih za nastavak razgovora. Dođite sa skraćenicom za snimanje pitanja koje vam padne na pamet dok neko razgovara, tako da možete brzo da se vratite na to kad imate priliku. Ne pokušavajte da snimite dijagrame u komplikovanom softveru za modeliranje; samo fotografišite skicirane dijagrame ili brzo crtajte rukom.

## KORIŠĆENJE PROSTORIJE ZA SASTANAK

*Koristite i zidove za crtanje (prekriveni papirom, ako nisu pripremljeni za crtanje) i podstaknite kolaborativnu učešće učesnika sednice.  
Koristite i alate za video konferencije.*

**Iskoristite fizički prostor** Većina prostorija ima četiri zida, pa ih koristite za vreme crtanja dijagrama ili pravljenja lista. Ako ne postoje bele table, pričvrstite velike listove papira na zidove. Imajte na raspolaganju lepljive beleške i markere. Pozovite ostale učesnike da ustanu i doprinesu zidu; kretanje oko pomaže održavanju ljudi angažovanim. Ako postoje artefakti koje treba pogledati projektujte ih na zid.

Olakšavanje kolaborativnih sesija sa učesnicima na više lokacija zahteva više kreativnosti. Možete koristiti alate za onlajn konferencije da biste delili slajdove i dozvolili interakcije. Ako je nekoliko učesnika u istoj sobi, koristite alate za video konferencije da biste pokazali udaljenim učesnicima šta je na zidovima i tabli.

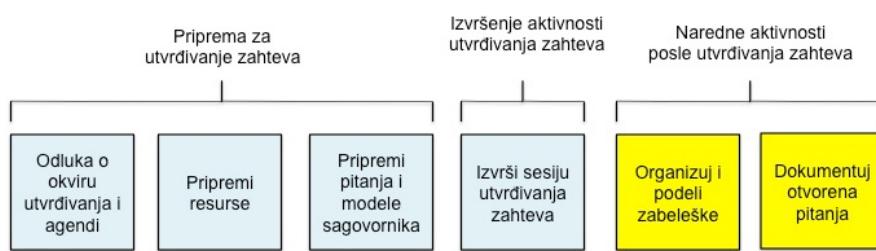
## ▼ Poglavlje 5

# Aktivnosti posle izazivanja zahteva

## ŠTA URADITI POSLE SESIJE PRIKUPLJANJA ZAHTEVA?

*Organizovanje i deljenje beleške sa sesije. Dokumentovati otvorena pitanja.*

Posle izvršenja aktivnosti prikupljanja (ili utvrđivanja) zahteva, ima puno preostalog posla. Treba da organizujete i podelite beleške sa sesija, da dokumentujete otvorena pitanja, da klasifikujete prikupljene informacije. Slika 1 označava te naredne aktivnosti posle jedne sesije izazivanja zahteva.



Slika 5.1 Aktivnosti koje slede izvršenje kativnosti izazivanja zahteva

**Organizovanje i deljenje beleški sa sesija** Odmah posle završene sesije izazivanja zahteva, dok vam su informacije još sveže, napravite belešku o održanoj sesiji. Ubeležite sve predloge. Dajte na recenziju beleške učesnicima sesije (intervjua, radionice, fokus grupe, i dr.) i u skladu sa tim izvršite izmene u beleški. Sačuvajte i sirove beleške, koje ste vodili na sastanku. Možda će vam trebati kasnije, da se na njih pozovete. Što podelite pripremljenu belešku na recenziju učesnicima, tražeći od njih da potvrde tačnost onoga što su izjavili i što je raspravljano na sesiji. Ako je potrebno, razjasnite razgovorom sa pojedinim učesnicima neka neslaganja. Konačnu verziju beleški pošaljite i ostalim akterima.

### Dokumentujte otvorena pitanja

Za vreme izvršenja aktivnosti izazivanja zahteva, primetili ste neka nerazjašnjena pitanja koja treba kasnije razjasniti da bi se popunio jaz u informacijama. Zbog toga, možete pripremiti i nova pitanja. Zapišite ih u alatu za praćenje otvorenih pitanja. Za svako pitanje, dajte vezu sa odgovarajućom beleškom. Zapisujte napredak u razmatranju i rešavanju otvorenih pitanja, Evidentirajte vlasnika otvorenog pitanje, tj. odgovornog da to pitanje reši, i rok do kada se pitanje mora razjasniti. Najbolje da koristite isti alat za praćenje otvorenih pitanja koje koristi tim za razvoj i testiranje softvera.

## ✓ Poglavlje 6

# Klasifikovanje predloga i stavova kupaca

## KAKO KLASIFIKOVATI INFORMACIJE DOBIJENE OD KUPCA?

*Analitičar mora da klasificuje pregršt zahteva koje je čuo u više kategorija zahteva, da ih dokumentuje i da ih koristi kako treba.*

Ne očekujte da će vam kupci (ili korisnici) dati jednu jezgrovitu, kompletну i dobro organizovanu listu njihovih potreba. Analitičar mora da klasificuje pregršt zahteva koje je čuo u više kategorija zahteva, da ih dokumentuje i da ih koristi kako treba. Slika 1 pokazuje 9 takvih kategorija. Za vreme aktivističkog izazivanja zahteva upišite kratke opaske u beležnicu, ako primetite da neka od informacija koju dobijate spada u jednu od ovih kategorija. Na primer, zapišite DP u krug pored informacije, da bi označili "definicija podataka".



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

Slika 6.1 Klasifikacija informacija dobijenih od korisnika ili iz priča korisnika

Informacija koja ne pripadne ni jednoj od navedenih 9 kategorija može biti:

- zahtev projekta koji nije vezan za razvoj softvera, kao što je obučavanje korisnika softvera,
- ograničenje projekta, kao što su ograničenja budžeta ili rokova
- neka prepostavka ili zavisnost,
- neka dodatna informacija istorijskog, kontekstualnog, ili deskriptivnog karaktera,
- irelevantna informacija koja ne dodaje vrednost sistemu.

Korisnici vam neće reći "ovo je naš poslovni zahtev". Kao analitičar, vi morate da odredite kom tipu informacija pripada iskaz koji čujete od učesnika procesa izazivanja zahteva. Sledеće fraze vam mogu da pomognu u ovom klasifikacionom procesu:

- **Poslovni zahtevi:** Sve što opisuje finansijsku, marketinšku i neku drugu poslovnu korist, koju organizacija-korisnik ili organizacija-proizvođač softvera želi da ostvari od proizvoda je poslovni zahtev.
- **Zahtev korisnika:** Uopšteni iskazi o tome šta su ciljevi korisnika, ili poslovni zadaci koje korisnik mora da ostvari, su zahtevi korisnika. To su najčešće slučajevi korišćenja, scenariji ili priče korisnika.

## SUGESTIJE ZA KLASIFIKACIJU INFORMACIJA U JEDNU OD 9 KATEGORIJA

*Morate da spojite informacije u jednu jasno prikazanu i dobro organizovanu kolekciju zahteva.*

- **Poslovna pravila:** Kada korisnik kaže da samo određeni korisnici mogu da realizuju neku aktivnost, onda on ne govori o nekom poslovnom pravilu. To nisu softverski zahtevi, kao što izgledaju, već funkcionalna pravila koji se dobijaju iz poslovnih pravila.
- **Funkcionalna pravila:** Funkcionalna pravila opisuju ponašanja koja se vide a koja sistem ostvaruje, pod određenim uslovima, i usled dejstva određenih akcija koje sistem dozvoljava korisniku da preduzme. Najčešće analitičar mora da preformuliše funkcionalni zahtev korisnika i da mu da veću preciznost.
- **Atributi kvaliteta:** Iskazi koji opisuju kako dobro sistem radi nešto su atributi kvaliteta. Ovi atributi obično sadrže sledeće reči: brz, lak, prijateljski prema korisniku, pouzdan, siguran. Morate sa korisnikom da razjasnite nejasne termine koje često koriste, kako bi ih zapisali preformulisali u jasne attribute kvaliteta.
- **Zahtevi spoljnog interfejsa:** Zahtevi u ovoj kategoriji opisuju veze vašeg sistema i drugih softverskih sistema.
- **Ograničenja:** Projektna i implementaciona ograničenja ograničavaju opcije inženjeru razvoja softvera. Uređaji sa ugrađenim softverom, obično imaju fizička ograničenja, kao što su veličina, težina, i veze interfejsa. Kao i u slučaju sa funkcionalnim zahtevima, ne prihvativte odmah ograničenje koje vam daje kupac. Pitajte ga zašto to ograničenje postoji, potvrdite njegovu validnost, i zapišite razloge zbog kojih se ovo ograničenje uključuje u sistem.
- **Zahtevi podataka:** Kupci govore o zahtevu podataka uvek kada opisuju format, tip podataka, dozvoljene vrednosti, ili unapred određenu vrednost nekog elementa podataka. Isto to važi ako govori o nekoj složenoj poslovnoj strukturi podataka, ili kada govori o nekom izveštaju koje se mora generisati.
- **Ideje za rešenja:** Mnogi "zahtevi" o kojima korisnici govore, su u stvari ideje za rešenje. Kada neko govori o specifičnom načinu interakcije sa sistemom, radi izvršenja neke akcije, on u stvari predlaže neko rešenje. Analitičar mora da "zagrebe ispod površine", ispod ideje rešenja, da bi došao da stvarnog zahteva. Ako analitičar stalno ponavlja pitanje "zašto", to će naterati korisnika da na kraju iskaže svoju stvarnu potrebu.

Klasifikacija informacija dobijenih od korisnika je samo početak kreiranja specifikacije zahteva. Morate da spojite informacije u jednu jasno prikazanu i dobro organizovanu kolekciju zahteva. Kako prolazite kroz ove informacije, uobičavajte jasne pojedinačne zahteve i stavite ih u odgovarajuću sekciju dokumenta i repozitorijuma tima. Prođite nekoliko puta kroz prikupljene informacije, da bi proverili da li klasifikovani zahtevi imaju karakteristike zahteva visokog kvaliteta.

## ▼ Poglavlje 7

### Još o zahtevima

## KAKO ZNATE DA STE ZAVRŠILI SA ZAHTEVIMA?

*Ovde se ukazuje na pojave koje su indikator da možete završiti sa izazivanjem zahteva i da pređete u sledeću fazu razvoja softvera.*

Ne postoji jasan signal koji označava da ste završili sa izazivanjem zahteva. Ako razvijate softver inkrementalno, onda će izazivanje zahteva da se stalno odvija, pri svakoj iteraciji. Sledeće pojave označavaju da ste praktično završili sa izazivanje zahteva:

- Korisnici ne mogu da iznesu još neki slučaj korišćenja niti priču korisnika.
- Korisnici predlažu nove scenarije, ali oni ne dovode do novih funkcionalnih zahteva.
- Korisnici ponavljaju pitanja koja su već ranije prodiskutovana.
- Preporučena nova svojstva, zahtevi korisnika, ili funkcionalni zahtevi su van okvira projekta.
- Predloženi novi zahtevi imaju nizak prioritet.
- Korisnici predlažu sposobnosti sistema koje će "nekad" imati, umesto da to bude sada primenljivo.
- Inženjeri razvoja i testeri koji pregledavaju zahteve u nekom domenu, postavljaju malo pitanja.

Spajanje zahteva od većeg broja korisnika je teško bez upotrebe organizacione šeme strukture sistema, kao što su slučajevi korišćenja. Nemoguće je dobiti sve zahteve. Zato, očekujte da će se neki javiti u fazi konstrukcije softvera.

Ne zaboravite da je vaš zadatak da obezbedite zajedničko razumevanje zahteva koje je dovoljno dobro za konstrukciju sledećeg izdanja softvera, ili za sledeći inkrement sa prihvatljivim nivoom rizika.

## NEKA UPOZORENJA U VEZI IZAZIVANJA ZAHTEVA

*Uravnotežite uticaj aktera. Definišite odgovarajući okvir. Izbegnite sukob analize i projektovanja. Istražujte moguće zahteve sa razlogom.*

Ovde se daju neka upozorenja koje bi trebalo da imate u vidu kada radite intervjuje sa korisnicima sistema:

- **Uravnotežite uticaj aktera:** Ako imate malo aktera sa kojima razgovarate, ili imate jednog koji je najagilniji i najglasniji, imate opasnost da su samo njihovi ili njegovi stavovi ušli u razmatranje, a ne i drugih. Na taj način, možete izostaviti neke druge potrebe i zahteve aktera. Zato kontaktirajte sa šampionima proizvoda, tj. predstavnikom svake klase korisnika, a on neka razgovaraju sa članovima svoje klase. Tako ćete izbegići izostavljanje nekih klasa korisnika iz analize potreba.

- **Definišite odgovarajući okvir:** Za vreme izazivanja zahteva, možete primetiti da je definisan okvir projekta neadekvatno definisan. Ili je suviše veliki, ili je suviše mali. Ako je isoviše veliki, prikupljate nepotrebno mnogo zahteva koji nisu potrebni i ne daju poslovnu vrednost kupcu. Ako je isoviše mali, i neki značajni zahtevi korisnika biće van tako postavljenog okvira, i trebalo bi ih po tome odbaciti. Tokom procesa izazivanja zahteva može se korigovati i vizija i okvir projekta, ako se utvrди da je to potrebno.
- **Izbegnite sukob analize i projektovanja:** Smatra se da zahteva treba da kažu ŠTA sistem treba da obezbedi, a da projektovanje i konstrukcija softvera treba da da rešenje KAKO sistem realizuje te zahteve, i kako on radi. To je malo uprošćena podela, jer ima i tzv. "sivih zona" između analize i projektovanja. Da bi korisnicima objasnili i poboljšali razumevanje pojedinih njihovih potreba, vi ćete ponekad koristiti i KAKO pri analizi modela, te koristiti različite prototipove (na primer, korisničkog interfejsa), i ukazati na hipotetički rad sistema. Međutim, treba korisnicima reći da je to samo radi ilustracije, i da prototipovi koje koristite ne moraju da budu konačno rešenje koje će dobiti.
- **Istražujte sa razlogom:** Tokom procesa izazivanja zahteva često istražujete neke stvari. Ako to isoviše radite, vi time usporavate proces. Koristite prototipove da abi proverili da li su ta istraživanja vredna pažnje. Ako projekat zahteva mnogo istraživanja, primeniti inkrementalni razvoj softvera. Na taj način ćete istraživati zahteve u okviru samo jedne iteracije, "u malom", i videti da li se to isplati.

## PRETPOSTAVLJENI I PODRAZUMEVANI ZAHTEVI

*Neki zahtevi nisu definisani jer su predpostavljeni ili podrazumevani i to kasnije napravi problem.*

Nikada nećete dokumentovati 100% zahteva vašeg sistema. Ali to može do rizika da rezultat projekta nije ono što se očekuje. Krivci za to su:

- **Prepostavljeni zahtevi:** Njih korisnici očekuju i bez formalne specifikacije. Ono što je korisniku očigledan zahtev, inženjeru razvoja ne mora da bude.
- **Podrazumevani zahtevi:** Oni moraju da postoje jer ih zahtevaju drugi zahtevi, ali nisu eksplicitno definisan u projektu. Inženjeri razvoja ne mogu da primene funkcionalnost za koju ne znaju.

Da bi smanjili rizik javljanja ovih jazova u zahtevima, postavite pitanje: "Šta mi prepostavljamo?" za vreme sesija izazivanja zahteva da abi utvrdili ove skrivene zahteve. Odgovore zapišite i proverite važnost toga.

Da bi utvrdili da li postoje podrazumevani zahtevi, analizirajte rezultate poletnih sesija izazivanja zahteva, da bi videli da bi utvrdili područja koja su nekompletна. Ako to utvrdite, ponovite razgovor sa odgovarajućim akterom da bi dodali ove nedostajuće zahteve.

Da bi ustanovili svojstva i potrebe koje korisnici nisu eksplicitno naveli, postavite pitanja van konteksta, uopštena i sa otvorenim odgovorom. Odgovor može da ukaže na karakteristike i poslovnog problema i potencijalnog rešenja. Na primer, na pitanje "Koja se preciznost očekuje od proizvoda?", može vas dovesti do pozadine, koju niste obuhvatili odgovorima topa "da" i "ne".

## NALAŽENJE ZAHTEVA KOJI NEDOSTAJU

*Primenite ove predložete tehnike radi nalaženja nedostajućih zahteva.*

Često se desi da nam neki zahtevi nedostaju. Koristite sledeće tehnike da bi ih otkrili i uključili u specifikaciju zahteva:

- **Dekomponujte uopštene zahteve u konkretnije, detaljnije zahteve**, kako bi bilo jasnije šta se stvarno zahteva. Uopšteni zahtevi koji se mogu različito tumačiti dovode do izostavljanja nekih zahteva. Analitičar ima jedno shvatanje tih zahteva, a inženjer razvoja - drugo.
- **Obezbedite da sve klase korisnike daju svoje ulaze u analizu.** Vidite da svaki potiče bar od jedne klase korisnika.
- **Pratite zahteve korisnika**, zahteve sistema, liste događaj-odgovor, i poslovna pravila do odgovarajućih funkcionalnih zahteva da bi proverili da li dobijate traženu funkcionalnost.
- **Proverite granične vrednosti zbog nedostajućih zahteva.** Slične slučajevе. Da li na primer poslovna pravila obuhvataju sve granične slučajevе.
- **Predstavite informacije o zahtevima na više načina.** Ono što promakne u nekom tekstu, neće u grafičkom modelu iste informacije.
- **Skupovi zahteva sa logičkim operatorima su često nekompletni.** Ako takvi uslovi nemaju odgovarajući funkcionalni zahtev, inženjer razvoja će prepostaviti odgovor sistema. Zato koristiti tablice odlučivanja ili stabla odlučivanja da bi pokrili sve slučajevе odlučivanja.
- **Napravite listu zajedničkih funkcionalnih oblasti radi provere (checklist) vašeg projekta.** Povremeno uporedite tu listu sa funkcijama koje ste specificirali da bi uočili eventualno postojanje razlika, tj. "rupa".
- **Model podataka može da otkrije funkcionalnost koja nedostaje.** Svi podaci moraju da imaju neku vezu sa nekom funkcijom njihovog kreiranja, menjanja, čitanja iz spoljnog izvora, i dodavanja ili brisanja. Skraćenica CRUD (create, read, update, delete) često označava ove operacije. Proverite da li se ove funkcije nalaze u vašoj aplikaciji za podatke koje koristite.

## ELICITATION TECHNIQUES | CABA (VIDEO)

*Trajanje: 8:15 minuta*

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

## REQUIREMENTS ENGINEERING - ELICITATION TECHNIQUES (VIDEO)

*Trajanje: 10:09 minuta*

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

## ▼ Poglavlje 8

### Vežba

## PITANJA ZA DISKUSIJU

*Tekst pitanja za diskusiju*

### **PITANJE 1.**

Razmislite o zahtevima koji su kasno otkriveni na vašem poslednjem projektu. Zašto su zanemareni tokom iznošenja? Kako ste mogli da otkrijete svaki od ovih zahteva ranije? Šta bi to značilo za vaš projekat?

(Asistenti iznose zaključke na bazi iskustva u razvoju ISUM-a ili LAMS-a.

Zatim, studenti iznose zaključke na bazi svojih projekata iz predmeta SE201, SE211 ili CS324) (20 min)

### **PITANJE 2.**

Navedi tehnike za utvrđivanje zahteva koje su se koristile u vašem prethodnom projektu. Koje su od njih dobro funkcionalne? Zašto? Koje nisu dobro funkcionalne? Što nisu?

(Studenti i asistenti iznose zaključke, svako u vezi sa svojim projektom) (10 min)

### **PITANJE 3.**

Identifikujte tehnike utvrđivanja za koje mislite da bi bolje funkcionalne na tom projektu i objasnite kako biste ih primenili da imate priliku. Identifikujte sve prepreke na koje biste mogli naići tokom primene te tehnike i razmislite o načinima za prevazilaženje tih prepreka.

(Studenti i asistenti iznose zaključke, svako u vezi sa svojim projektom) (15 min)

## ZADACI ZA VEŽBU

*Tekst zadataka za vežbu*

### **ZADATAK 1.**

Pronađite odeljak koji se odnosi na zahteve iz dokumenta koji predstavlja izveštaj o urađenom projektnom zadatku za jedan od ponuđenih predmeta. Izdvojite tekst koji se odnosi na zahteve u poseban dokument pod nazivom *SE322-V04-Utvrdjivanje zahteva*, koji će vam poslužiti za analizu. Klasifikujte svaki od zahteva u kategorije prikazane na slici 1 u poglavljju 6. Ako pronađete stavke koje su nepravilno organizovane, premestite ih na odgovarajuće mesto u dokumentaciji sa vašim zahtevima. Ako otkrijete da neki tip zahteva nedostaje, dopišite ih.

(Zadatak asistenata je da pokažu studentima dokument koji predstavlja predlog zahteva za sistem koji su razvijali, a koji su dobili od korisnika ili kreirali na osnovu razgovora sa njima.

Zadatak asistenta je da uradi istovetnu analizu na svom primeru prikupljenih zahteva.) (20 min)

**ZADATAK 2.**

Pronađite koja su najčešća pitanja za intervjuje koji se sprovode sa stejkholderima, radi utvrđivanja zahteva. (10 min)

**ZADATAK 3.**

Selekcijom pitanja koja ste pronašli radeći zadatak 2, dajte svoj predlog pitanja za intervju sa klijentom koji bi naručio sistem za upravljanje radom taksi udruženja. (5 min)

**ZADATAK 4.**

Zamislite da je potrebno da prikupite i analizirate zahteve za potrebe razvoja modula Studentska služba Univerziteta Metropolitan. Selekcijom pitanja koja ste pronašli radeći zadatak 2, dajte svoj predlog pitanja za intervju sa referentom službe Univerziteta Metropolitan. (10 min)

## ▼ Poglavlje 9

### Domaći zadatak

#### DOMAĆI ZADATAK 5

##### *Tekst domaćeg zadatka*

Identifikujte tehniku (ili više njih, ako je moguće) utvrđivanja zahteva za koju mislite da bi najbolje funkcionalala na primeru sistema koji ste dobili da analizirate za DZ01. Objasnite kako biste primenili tu tehniku. Sa kojim akterima zainteresovanih strana biste najverovatnije komunicirali u cilju dobijanja zahteva? Identifikujte sve prepreke na koje biste mogli naići tokom primene tehnike za koju ste se opredelili i razmislite o načinima za prevazilaženje tih prepreka.

Traženu analizu opisati sa minimum 500 reči.

✓ Poglavlje 10

## Projektni zadatak

### ZADATAK ZA RAD NA PROJEKTU

*Tekst zadatka za rad na projektu*

Dopunite **Dokument o viziji i okviru**, ako ste nešto propustili nakon treće i četvrte nedelje nastave, i poslati ga predmetnom asistentu na mejl.

## ✓ Poglavlje 11

### Zaključak

#### ZAKLJUČAK

1. Razvoj zahteva ima ciklični karakter sa fazama: izazivanje, analiza i specifikacija zahteva. Sesija izazivanja zahteva ima 7 aktivnosti, 3 u pripremnoj fazi, jedna u izvršnoj, a dve u završnoj.
2. Koriste se sledeće tehnike izazivanja zahteva: intervjui, radionice, fokus grupe, ankete sa upitnicima, posmatranja i analiza spoljnijih interfejsa.
3. Plan izazivanja uključuje tehnike koje koristite, kada planirate da ih koristite i sa kojom svrhom.
4. U pripremnoj fazi izazivanja zahteva, sprovode se sledeće aktivnosti: planiranje okvira sesije i definisanje agende, planiranje resursa i priprema pitanja za sagovornika i priprema modela sagovornika.
5. U fazi izvršavanja izazivanja zahteva radite sledeće: obrazujte aktere, dobro vodite beleške i iskoristite fizički prostor za sesiju.
6. Posle izvršenja aktivnosti izazivanja zahteva, slede dve aktivnosti: Organizovanje i deljenje beleške sa sesije i dokumentovati otvorena pitanja.
7. Analitičar mora da klasifikuje pregršt zahteva koje je čuo, u više kategorija zahteva, da ih dokumentuje i da ih koristi kako treba
8. Morate da spojite informacije u jednu jasno prikazanu i dobro organizovanu kolekciju zahteva.
9. Da bi znali kada ste završili posao sa izazivanjem zahteva, koristite određene indikatore.
10. Pri izazivanju zahteva, obratite pažnju na sledeće: uravnotežite uticaj aktera, definišite odgovarajući okvir, izbegnite sukob analize i projektovanja. Istražujte potrebe sa razlogom.
11. Neki zahtevi nisu definisani jer su prepostavljeni ili podrazumevani i to kasnije napravi problem.
12. Neki zahtevi nedostaju. Da bi ih otkrili koristite preporučene tehnike njihovog otkrivanja.

#### 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 7** 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,



SE322 - INŽENJERSTVO ZAHTEVA

## Primena slučajeva korišćenja

Lekcija 06

PRIRUČNIK ZA STUDENTE

# SE322 - INŽENJERSTVO ZAHTEVA

## Lekcija 06

### ***PRIMENA SLUČAJEVA KORIŠĆENJA***

- ✓ Primena slučajeva korišćenja
- ✓ Poglavlje 1: Slučajevi korišćenja i priče korisnika
- ✓ Poglavlje 2: Slučajevi korišćenja
- ✓ Poglavlje 3: Utvrđivanje slučajeva korišćenja i zahteva u CTS
- ✓ Poglavlje 4: Vežba
- ✓ Poglavlje 5: Domaći zadatak
- ✓ Poglavlje 6: Projektni zadatak
- ✓ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

## ▼ Uvod

# UVOD

### *Uvodne napomene*

U ovoj lekciji detaljnije radimo slučajeve korišćenja koji su osnov za razumevanje potreba korisnika, za utvrđivanje njegovih zahteva, za definisanje scenarija, tj. tokova akcija koje se moraju izvršiti u nekom slučaju korišćenja, za utvrđivanje funkcionalnih zahteva i definisanja testova prihvatanja sistema. Ako se projektuju i razvijaju objektno-orientisani (OO) sistemi, onda se lako mogu iz slučajeva korišćenja da definišu model objekata i sekvensijalni dijagrami, iz kojih se daljim detaljisanjem razvijaju klase, sa svojim atributima i metodima.

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

## ▼ Poglavlje 1

# Slučajevi korišćenja i priče korisnika

## VIDEO PREDAVANJE ZA OBJEKAT "SLUČAJEVI KORIŠĆENJA I PRIČE KORISNIKA"

*Trajanje video snimka: 19min 29sek*

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

## OBLASTI PRIMENE SLUČAJEVA KORIŠĆENJA I PRIČA KORISNIKA

*Slučajevi korišćenja i priče korišćenja služe da se utvrde šta korisnici žele da sistem rad. Korisne su za poslovne aplikacije, razvoj veb sajtova, i interakciju*

Preduslov za projektovanje softvera koji zadovoljava potrebe svojih korisnika je da se razume šta korisnik namerava da radi sa softverom. Neko razvija softver tako da definiše sam šta sve softver može da radi nadajući se da je to potrebno korsnicima ("product-centric approach"). Drugi polaze od utvrđenih potreba korisnika i na osnovu toga specificiraju zahteve korisnika i prema tome razviju softver ("user-centric approach"). Usmerenje ka korisniku i njegovom planu korišćenja softvera pomaže otkrivanju potrebne funkcionalnosti softvera, izostavljanju svojstava softvera koje se neće koristiti, i olakšava određivanje prioriteta u razvoju softvera.

Zahtevi korisnika, kao što je poznato, se nalaze između poslovnih ciljeva projekta i funkcionalnih zahteva koji opisuju šta inženjer razvoja treba da projektuje i izvrši. U ovoj lekciji izložićemo dve najčešće korišćene metode za istraživanje zahteva korisnika: slučajeve korisnika i priče korisnika.

Priče korisnika su počele da se koriste sa primenom agilnih metoda razvoja softvera. One na koncizan način opisuju šta je potreba korisnika i služi za početak razgovora radi utvrđivanja detalja.

Obe metode služe da se utvrdi šta korisnice žele da sistema radi. Opisuju šta korisnik treba da radi sa sistemom, opisuju njihovu interakciju. Svaki slučaj korisnika opisuje više scenarija korišćenja (jedan obezbeđuje potrebnu funkcionalnost, a ostali opisuju razne posebne situacije i izuzetke).

Biznis analitičar treba da na bazi slučajeva korišćenja i njihovih scenarija definiše funkcionalne zahteve i da testira da li sistem, na osnovu njih, izvršava opisane scenarije.

Slučajevi korišćenja i priče korisnika su vrlo korisne za istraživanje zahteva poslovnih aplikacija, vebsajtova, kioska, i sistema u kojima korisnik kontroliše aplikaciju. Međutim, ove dve metode nisu zgodne za određene aplikacije, kao što je paketna obrada, računarski intenzivne aplikacije, biznis analitiku, a skladištenje podataka (**data warehousing**) može da ima samo nekoliko slučajeva korišćenja.

Pored ovih računarski intenzivnih aplikacija, postoje i druge aplikacije u kojima slučajevi korišćenja i priče korisnika nisu odgovarajuće. To su sistemi sa ugrađenim softverom i drugi sistemi u realnom vremenu. Zahtevi za ovakve sisteme su liste koje povezuju događaje i odgovarajuće odgovore sistema.

## UPOREĐENJE SLUČAJEVA KORIŠĆENJA I PRIČA KORISNIKA

*Slučaj korišćenja opisuje sekvencu interakcija između sistema i spoljnih aktera. Priča korisnika (user story) je kratak opis svojstva ispričan iz perspektive osobe koja želi novu sposobnost*

Slučaj korišćenja ili upotrebe (**use case**) opisuje sekvencu interakcija između sistema i spoljnih aktera koja dovodi akterima ostvarenje rezultata od značaja za njih. Nazivi slučajeva korišćenja se uvek pišu u formatu "glagol + objekat". Birajte snažne opisne nazine koje pokazuju vrednost koju obezbeđuju nekom korisniku. Na slici 1 prikazani su nazivi pojedinih slučajeva korišćenja za slučaj nekoliko aplikacija.

Aplikacija	Neki slučajevi korišćenja
Chemical Tracking System (Sistem za praćenje hemikalija)	Zahtevaj himikaliju Štampaj list sa podatkima o bezbednom materijalu Promeni zahtev ya hemikalijom Promeni status narušbenice Generiše kvartalni izveštaj o upotrebi hemikalije
Sistem prijavljivanja na aerodromu	Prijava za let Štampaj kartu za ukrcavanje Promeni sedište Proveri prtljag Kupi nadogradnju
Računovođstveni sistem	Kreiraj fakturu Uskladi izvod iz računa Unesi transakciju sa kreditnom karticom Odštampaj poreski formular prodavca Traži specifičnu transakciju
Onlajn prodavnica	Dopuni profil kupca Traži jednu stavku Kupi jednu stavku Prati aket sa isporukom Poništi neposlat nalog

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

Slika 1.1 Neki slučajevi upotrebe ili korišćenja nekoliko aplikacija

Priča korisnika (**user story**) je kratak opis svojstva ispričan iz perspektive osobe koja želi novu sposobnost, koja je obični ili korisnik ili kupac. Priča korisnika se obično piše u sledećem formatu:

*Kao <tip korisnika> želim <neki cilj> tako da <neki razlog>*

Upotreba ovog formata obezbeđuje prednost u odnosu i na kraći naziv slučaja korišćenja zbog toga što priča korisnika, pored cilja korisnika, daje informaciju i o klasi korisnika i o razlozima zahteva. Klasa korisnika odgovara primarno akteru u slučaju korišćenja. Razlog se mora navesti u kratkom opisu slučaja korišćenja. Na slici 2 se prikazuje se pojedini slučajevi korišćenja sa slike 1 mogu da predstave kratkim pričama

Aplikacija	Slučaj korišćenja	Priča korisnika
Chemical Tracking System	Zahtevaj hemikaliju	Kao hemičar, želim da yahtevam hemikaliju da bih izvršio eksperiment
Sistem prijavljivanja na aerodromu	Prijava za let	Kao putnik, želim da seprijavim ya let kako bi leteo u moju destinaciju
Računovođstveni sistem	Kreiraj fakturu	Kao mali biznismen, želim da kreiram fakturu, kako bi ispostavio račun kupcu.
Onlajn prodavnica	Dopuni profil kupca	Kao kupac, želim da dopunim moj profil kupca, tako da buduće kupovine idu na račun kartice sa novim brojem

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

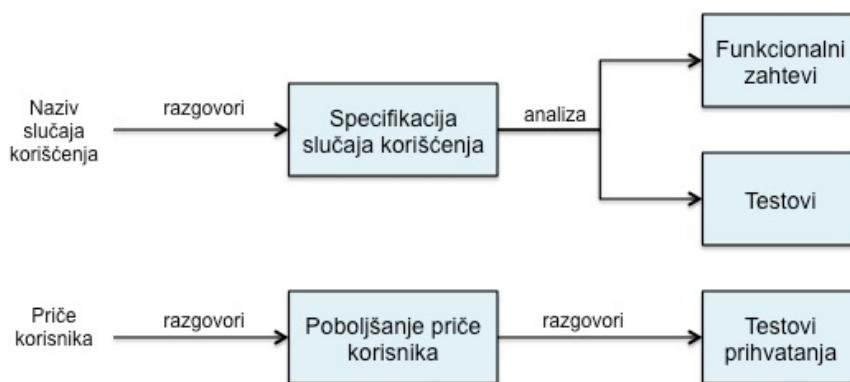
Slika 1.2 Slučajevi korišćenja (upotrebe) i odgovarajuće priče korisnika

## RAZLIKE IZMEĐU SLUČAJEVA KORIŠĆENJA I PRIČA KORISNIKA.

*Na osnovu specifikacije slučaja korišćenja, analitičar onda dobija funkcionalne zahteve. Priče korisnika se detaljišu i dele na manje, i vode do*

Na prvi pogled, slučajevi korišćenja i priče korisnika izgledaju slično. Obe metode pokušavaju da opišu šta različiti tipovi korisnika žele da urade interakcijama sa sistemom. Međutim, njihovi procesi nadalje idu različitim pravcima, kao što se vidi na slici 3. pristupa proizvode rezultate, kao što su vizuelni modeli analize.

U slučaju primene slučajeva korišćenja, analitičar mora u razgovoru sa korisnikom da razume kako on zamišlja dijalog kupca i sistema pri izvršenju slučaja korišćenja. Analitičar daje strukturu dobijenim informacijama u skladu sa formularom za definisanje slučaja korišćenja. On sadrži prostor za upisivanje razumevanja slučaja korišćenja.



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

Slika 1.3 Kako zahtevi korisnika vode do funkcionalnih zahteva i testova, preko slučajeva korišćenja i priča korisnika

Slučaj korišćenja pomaže analitičaru i korisniku da otkriju sve neophodne informacije vezane za slučaj korišćenja. Na osnovu specifikacije slučaja korišćenja, analitičar onda dobija funkcionalne zahteve koje inženjer razvoja treba da primeni, a tester treba da pripremi testove radi provere da li je slučaj korišćenja pravilno применjen. Inženjer razvoja može da примени u celini slučaj korišćenja i jednom izdanju ili iteraciji, ili da samo deo slučaja korišćenja, a ostale delova - u narednim izdanjima ili iteracijama.

Kod agilnih projekata, priča korisnika služi za unos budućih razgovora koji će se voditi u budućnosti, kada to zatreba, između inženjera razvoja, predstavnika kupca, i biznis analitičara. Ovi razgovori otkrivaju dodatne informacije koje inženjeri razvoja moraju da znaju da bi primenili priču. Na taj način oni poboljšavaju priče korisnika, što dovodi do kolekcije manjih pojedinačnih komada nove funkcionalnosti. Male priče korisnika su veličine koje se mogu realizovati u jednoj iteraciji.

## I JOŠ NEŠTO O RAZLIKAMA

*Priče korisnika nude prednost u konciznosti i jednostavnosti, a slučajevi korišćenja daju učesnicima strukturu i kontekst, što pričama korisnika nedost*

Umesto funkcionalnih zahteva, timovi agilnih projekta sa detaljnijim pričama korisnika, dobijaju "zadovoljavajuće uslove", koji su osnov za definisanje testova prihvatanja. Razmišljanje o testovima pomaže vam da utvrdite promene osnovne priče korisnika (ili slučaja korišćenja), uslove izuzetaka, i nefunkcionalne zahteve, kao što su performanse i mere bezbednosti. Ako programer primeni potreban kod da bi zadovoljio testove prihvatanja, time će zadovoljiti i uslove zadovoljenja, a to onda znači da je priča korisnika korektno применена (implementirana).

Priče korisnika daju koncizan iskaz o potrebama korisnika. Slučajevi korišćenja idu dublje u opisivanje kako korisnik zamišlja interakciju sa sistemom da bi zadovoljio svoje ciljeve. Slučajevi korišćenja ne bi trebalo da ulaze u aspekte projektovanja. Priče korisnika nude prednost u konciznosti i jednostavnosti, a slučajevi korišćenja daju učesnicima strukturu i kontekst, što pričama korisnika nedostaje. Slučajevi korišćenja omogućavaju analitičaru da na organizovani način vodi izazivanje zahteva, što je više nego samo prikupljanje onoga što korisnici žele i šta im je potrebno, kao kod priča korisnika.

Postoje mišljenja da priče korisnika ne nude rešenje za slučaj velikih i zahtevnijih projekata. One ne nude strukturu informacija o potrebama korisnika i može se desiti da se neki test prihvatanje ne definiše. Zato, analitičar treba da ima iskustva sa pričama korisnika da ne bi ispušto neku funkcionalnost.

## ▼ Poglavlje 2

### Slučajevi korišćenja

#### VIDEO PREDAVANJE ZA OBJEKAT "SLUČAJEVI KORIŠĆENJA"

*Trajanje video snimka: 32min 9sek*

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

#### PRIMER ODREĐIVANJA AKTERA U CHEMICAL TRACKING SYSTEM

*Slučajevi koršćenja obezbeđuju apstraktnu vizualno predstavljanje zahteva korisnika.*

Slučaj koršćenja opisuje redosled interakcija između sistema i spoljnog aktera dovodi do rezultata koji obezbeđuje akteru određenu vrednost. Slučajevi koršćenja obezbeđuju apstraktno vizualno predstavljanje zahteva korisnika.

Akter je osoba (ali može biti i drugi sistem, ili uređaj) koji je u interakciji sa sistemom i izvršava slučaj koršćenja. U slučaju koji razmatramo na predmetu, projekta Chemical Tracking System (CTS), ulogu aktera obavlja **Podnositelj zahteva (Requestor)** koji je predstavnika klasi korisnika **Requestor**. Članovi ove klase mogu biti i hemičari i zaposleni u skladištu hemikalija, jer i jedni i drugi mogu da zahtevaju isporuku hemikalija.

Da bi lakše utvrdili ko je akter sistema, možete predstavniku korisnika sistema postaviti sledeća pitanja:

- Ko (ili šta) se obaveštava kada se nešto desi u sistemu?
- Ko (ili šta) obezbeđuje informaciju ili servise sistemu?
- Ko (ili šta) pomaže da sistem odgovori i da završi zadatku?

Strelica od aktera ka sistemu ukazuje na primarnog aktera. Strelica koja od slučaja koršćenja vodi ka akteru, pokazuje ko je sekundarni akter, koji ne inicira interakciju sa slučajem koršćenja (kao primarni akter), ali u njoj učestvuje. Ostali softverski sistema obično imaju ulogu sekundarnog aktera, jer pomažu izvršenju slučaja koršćenja (na primer, baza za obuku)

Sistem se aktivira kada Podnositelj zahteva opasnu hemikaliju koja zahteva da je podnositelj zahteva obučen da je koristi. Strelice u dijagramu koršćenja pokazuju interakcije između

aktera i slučajeva korišćenja. Za razliku od dijagrama konteksta, one ne predstavljaju tok podataka, kontrolnih signala ili materijala.



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

Slika 2.1 Deo slučaja korišćenja projekta Chemical Tracking System

## SLUČAJEVI KORIŠĆENJA I SCENRIJI KORIŠĆENJA

*Scenario je opis jednog primera upotrebe sistema. Jedan slučaj korišćenja ima kolekciju scenarija.*

Slučaj korišćenja predstavlja jednu diskretnu, samostalnu aktivnost koju jedan akter izvršava da bi dobio rezultat koji predstavlja dodatnu vrednost. Sam slučaj korišćenja (aktiviran jednom akcijom aktera), u sebi sadrži niz aktivnosti koji imaju isti cilj. **Scenario je opis jednog primera upotrebe sistema. Jedan slučaj korišćenja je predstavlja kolekciju scenarija upotrebe sistema, a scenario je jedan primer (instanca) slučaja korišćenja.** Kada analizirate zahteve korisnika, možete početi od nekog uopštenog slučaja korišćenja, pa da razvijete više specijalizovane slučajeve korišćenja. Možete da radite i obrnuto, možete generalizovati prethodne razvijene specijalizovane scenarije.

Slika 2 prikazuje primer jednog slučaj korišćenja, "Zahtev hemikalije", u projektu Chemical Tracking sistem koji radimo u svim lekcijama ovog predmeta. Ne mora uvek da se popuni ceo formular. To zavisi od specifičnosti slučaja korišćenja. Ako neka informacija iz formulara negde postoji u dokumentaciji, dovoljno je sam dati referencu na tu informaciju

Bitni elementi jednog slučaja korišćenja su:

- jedinstveni identifikacioni broj slučaja korišćenja i njegov naziv.
- Kratki tekstualni opis svrhe slučaja korišćenja
- uslov početka koji pokreće izvršenje slučaja korišćenja
- nula ili više preduslova za početak slučaja korišćenja
- jedan ili više posluslova koji opisuju stanje sistema posle uspešnog završetka slučaja korišćenja.

UC-4 Zahtev hemikalije		
Miloš Ilić	Datum kreiranja	22.3.2013
Podnositelj zahteva	Sekundarni akter:	Kupac, Sladište hemikalija, Baza za obuku
Podnositelj zahteva specifira željene hemikaliju u zahtev unošenjem njenog imena ili ID broja ili unošenjem njene strukture iz alata za crtanje hemijske strukture. Sistem ili nudi podnositocu zahteva kontejner sa hemikalijom iz skladišta hemikalija ili dozvojava zahtevaocu hemikalija da naruči kontejner od prodavca.		
Podnositelj zahteva nayačuje da želi da yahteva hemikaliju.		
PRE-1. Identitet korisnika je utvrđen.		
PRE-2. Korisnik je ovlašćen da yahteva hemikaliju.		
PR-3 Baza hemikalija je onlajn.		
POST-1. Zahtev je uskladišćen in CST.		
POST-2. Zahtev je poslat u Sladište hemikalija ili Kupcu		
<b>4.0 Zahtev za hemikalijom iz Skladišta hemikalija.</b>		
1. Podnositelj zahteva specifira željenu hemikaliju.		
2. Sistem izlistava kontejnere sa željenom hemikalijom		
3. Sistem daje Podnositocu zahteva opciju na Pogled na istoriju kontejnera za bilo koji kontejner.		
4. Podnositelj zahteva bira određeni kontejner ili pita da postavi zahtev isporučiocu hemikalija (vidi 4.1).		
5. Podnositelj zahteva unosi druge informacije da bi kompletirao zahtev.		
6. Sistem skladišti zahtev i obaveštava Stovarište hemikalija.		
<b>4.1 Zahtev za hemikalijom od Isporučiova</b>		
1. Podnositelj zahteva traži katalog isporučiova za hemikaliju (vidi 4.1E)		
2. Sistem prikazuje listu prodavaca hemikalije sa raspoloživim veličinama kontejnera, kvalitetom i cenama.		
3. Podnositelj zahteva bira isporučioča hemikalije, veličinu kontejnera, njegov kvalitet i broj kontejnera.		
4. Podnositelj yahteva unosi druge informacije radi kompletiranja yahteva.		
5. Sistem skladišti zahtev i obaveštava Kupca.		
<b>4.1.E1 Hemikalija nije komercijalno na raspolaganju</b>		
1. Sistem prikazuje poruku «nema isporučioča za tu hemikaliju».		
2. Sistem podnositelja yahteva da li želi podnese yahtev ya neju drugu hemikaliju (3a) ili da iyače i sistema (4a).		
3a. Podnositelj yahteva traži d apodnese yahtev ya drugu hemikaliju		
3b. Sistem počinje sa novim normalnim tokom.		
4a. Podnositelj zahteva traži izlaz.		
4b. Sistem završava slučaj korišćenja.		
<b>Visoka</b>		
Otprilike 5 puta nedeljno po hemičaru, 200 puta nedeljno od zaposlenih u Sladištu hemikalija		
BR-28. BR-31		
Sistem mora da omogući unos hemijske strukture u standarnom obliku od bilo koga podržanog paketa sa crtežima hemijskih struktura.		
Prepostavlja se da su proverene unete strukture hemikalije		

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

Slika 2.2 Delimična specifikacija slučaja korišćenja "Zahtev za hemikaliju" za CTS

## PREDUSLOVI I POSTUSLOVI

*Preduslovi definišu šta mora da bude ispunjeno da bi sistem počeo da izvršava slučaj korišćenja. Postuslovi opisuju stanje sistema posle uspešnog izvršenja slučaja korišćenja.*

**Preduslovi** (*preconditions*) definišu šta mora da bude ispunjeno da bi sistem počeo da izvršava slučaj korišćenja. Sistem mora da ima testove za utvrđivanje ispunjenosti preduslova. Preduslovi mogu da opišu stanja sistema ("ATM mora da u sebi ima novac, da bi mogao da pokrenem slučaj povlačenje novca iz automata"), ali ne opisuju nameru korisnika ("korisniku je potreban novac").

Kada sistem dobije događaj koji ga pokreće na izvršenje nekog uslova korišćenja, on onda ide na proveru preduslova, i ako su zadovoljeni, počinje sa izvršavanjem slučaja korišćenja. Pokretač nije preduslov.

**Postuslovi** (*postconditions*) opisuju stanje sistema posle uspešnog izvršenja slučaja korišćenja. Postuslovi mogu da obuhvate:

- nešto što korisnik može da opazi (na primer, balans računa u banci).
- fizički rezultat (ATM je isporučio novac i izveštaj),

- promenjeno je unutrašnje stanje sistema (odbijena sa bilnas stanja računa povećena količina novca i taksa za transakciju).

Svi izlazi iz slučaja korišćenja bi trebalo da obezbede određenu vrednost korisniku, te su svi oni opažljivi korisniku. Ono što je neophodno da bude postuslov, a što korisnik ne mora da opazi, to mora biznis analitičar da stavi u postuslove, ako je to uslov da sistem radi ispravo,

Koristite konvenciju o označavanju malih i malih slova

Specifikacije slučajeva upotrebe sastoje se od brojnih malih paketa informacija: normalni i alternativni tokovi, izuzeci, preduslovi i postuslovi, i tako dalje. Primer na slici2 pokazuje jednostavnu konvenciju o označavanju koja vam može pomoći da ovi elementi budu ravni. Svaki slučaj upotrebe ima redni broj i smisleno ime koje odražava korisnikov cilj: UC-4 Zahtevaj hemikaliju. Identifikator normalnog protoka za ovaj slučaj upotrebe je 4.0. Alternativni tokovi se identifikuju povećanjem broja desno od decimalnog značaja, tako da je prvi alternativni tok 4.1, drugi bi bio 4.2, i tako dalje. I normalni i alternativni protok mogu imati svoje izuzetke. Prvi izuzetak za slučaj normalnog protoka upotrebe broj 4 bio bi označen sa 4.0.E1. Drugi izuzetak za prvi alternativni tok za ovaj slučaj upotrebe bio bi 4.1.E2.

## NORMALNI I ALTERNATIVNI TOKOVI (SCENARIJI) I IZUZECI

*Jedan scenario predstavlja jedan normalan tok događaja za slučaj korišćenja. Može imati i alternativne tokove (scenarije).*

Jedan scenario predstavlja jedan normalan tok događaja za slučaj korišćenja. On se takođe naziva i glavnim tokom događaja, ili osnovnim tokom, ili primarnim scenarijem, glavnim uspešnim scenarijem, i dr. Na slici 2 je bio opisan glavni tok događaja, tj. primarni scenario za slučaj korišćenja "Zahtev za hemikaliju". Kao što se vidi, dat je redosled koraka koje aktor i sistem treba da izvrše. Za svaki korak se navodi ko izvršava taj korak.

Drugi uspešni tokovi su alternativni tokovi ili sekundarni scenariji. Alternativni tokovi isporučuju rezultat kao i normalni tok, ali predstavljaju manje uobičajen rezultat ili neki specifičan rezultat. Normalni tok može imati grananje u neki alternativni tok u nekoj tački odlučivanja u sekvenci dijaloga. Kasnije se može spojiti sa glavnim tokom.

Priče korisnika najčešće pokrivaju jedan scenario ili alternativni tok.

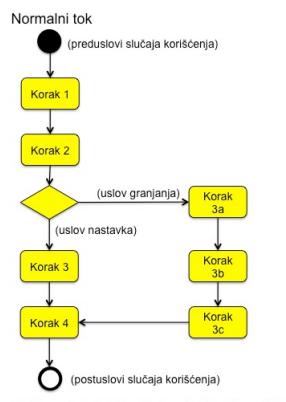
Uslovi koji mogu da spreče uspeh slučaja korisnika su izuzeci. Izuzeci (exceptions) opisuju uslove pri kojima se javljaju greške prilikom izršenja slučajeva korišćenja, U nekim slučajevima, slučaj korišćenja se može opraviti od greške i uspešno završiti, međutim, u nekim, ne može, i prekida izvršenje. Ako ne definišete akciju opravka sistema u uslovima kada dođe da izuzetka, onda sistem prekida rad.

Neke greške mogu da izazovu posledice i po više slučaja korišćenja, ili u više koraka normalnog toka događaja u jednom slučaju korišćenja. Na primer, gubitak veze u mreži, greška u bazi podataka, i dr. U ovakvim slučajevima ovo treba uzeti kao dodatne funkcionalne

zahteve, a ne kao izuzetke i ponavljati ih u svim slučajevima korišćenja u kojim može da dođe do ovakve greške

Morate izvršiti sve izuzetke ako želite da sprečita da vam tokovi stanu. Kodiranje u dobrom delu se odnosi na izvršenje izuzetaka. Međutim to je nužno da bi dobili robustan softver.

Pored tekstualnog opisa tokova, mogu se koristiti i grafičke metode, kao što su karte tokova ili UML dijagrami aktivnosti (slika 3).



Slika 2.3 UML dijagram aktivnosti sa glavnim i alternativnim tokom (scenarijom)

## EXTEND I INCLUDE

*Veza extend može povremeno da proširi osnovni slučaj korišćenja.  
Veza include omogućuje da se jedan slučaj korišćenja koristi u više slučajeva korišćenja.*

Postoje dve veze između slučajeva korišćenja:

- **extend** - slučaj korišćenja koji u nekim slučajevima može da proširi drugi, osnovni slučaj korišćenja
- **include** - slučaj korišćenja koji je obavezan deo obično više drugih slučajeva korišćenja.

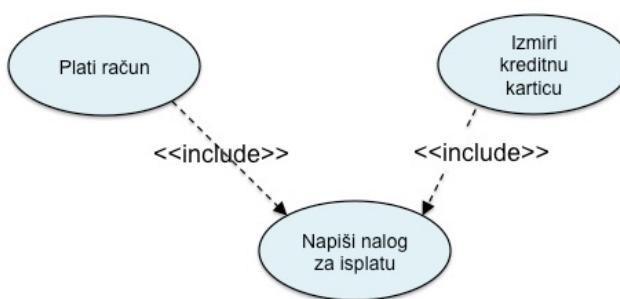
Navećemo prvo primer veze extend u dijagramu slučajeva korišćenja projekta Chemical Tracking Request koji je dat ranije na slici 1. Na tom dijagramu se vidi da akter **Kupac** koristi slučaj korišćenja "**Traži katalog proizvođača**". Ako se pogleda specifikacija slučaja korišćenja "**Zahtev za hemikalijom**" na slici 2, vidi se da sistem daje **Podnosiocu zahteva za hemikalijom** izveštaj o kontejnerima sa tom hemikalijom (ako je poseduje). Sistem daje i opciju Podnosiocu zahteva da koristi slučaj korišćenja "**Traži katalog proizvođača**", Kada Podnositelj zahteva bira ovu opciju? Kada je **Skladište hemikalija** nema, ili nema dovoljno tražene hemikalije. To se povremeno dešava. Tada tok događaja prelazi na alternativni tok događaja (alternativni scenario) i ide se u izvršenje slučaja korišćenja "**Traži katalog proizvođača**" koje je u vezi sa **Kupcem**, tj. agentom koji predstavlja nabavnu službu organizacije. Kako se slučaj korišćenja "**Traži katalog proizvođača**" ponekad koristi, onda se može on povezati sa vezom **extend** (slika 4).



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

Slika 2.4 Primer korišćenja veze extend između dva slučaja korišćenja u CTS.

Za vezu include dajemo sledeći primer. Zamislimo da softver za računovodstvo koristi dva slučaja korišćenja "**Plati račun**" i "**Izmiri kreditnu karticu**". Kako oba slučaja korišćenja podrazumevaju da korisnik treba da nalog za isplatu, onda se može uvesti treći slučaj korišćenja "**Daj nalog za isplatu**", koji bi koristila oba navedene slučaja korišćenja (slika 5).



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

Slika 2.5 Primer korišćenja veze include u softveru za računovodstvo

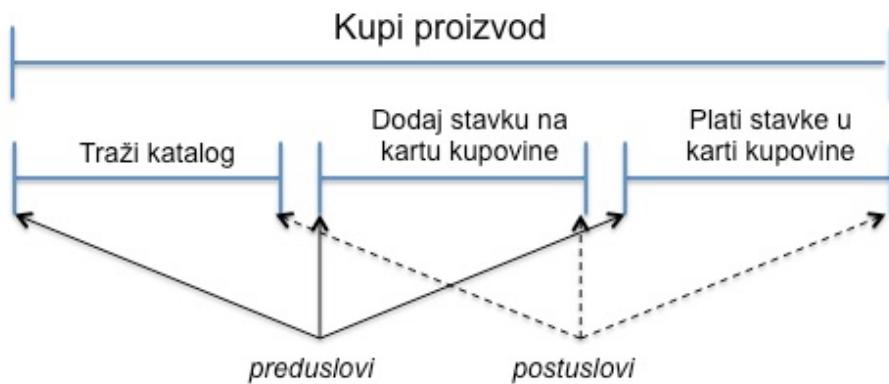
## POVEZIVANJE PREDUSLOVA I POSTUSLOVA

*Izjednačavanje postuslova sa prethodnim i preduslovima narednog slučaja korišćenja, mogu se povezati više sekvenčnih slučajeva korišćenja u jedan, makro slučaj korišćenja.*

Često je poželjno povezati nekoliko slučajeva korišćenja u jedan "makro" slučaj korišćenja koji sadrži više drugih slučajeva korišćenja. Time se jednom transakcijom pokreće izvršenje više drugih slučajeva korišćenja koje obuhvata "makro" slučaj korišćenja.

Navešćemo jedan primer, Veb sajt e-trgovine koji može koristiti sledeće slučajeve korišćenja: "**Traži katalog**", "**Dodaj stavku u kartu kupovine**" i "**Plati stavke u karti kupovine**". Ova tri slučaja korišćenja se uvek sekvenčno izvršavaju. Zato je jednostavnije da se oni povežu u jedan "makro" slučaj korišćenja "**Kupi proizvod**", pa kada se kreće u njegovo izvršavanja, onda se kreće u sekvenčno izvršavanje sva tri slučaja korišćenja, u okviru samo jedne transakcije.

Kako ovo specificirati? Da bi opisani proces radio, svaki od tri navedena slučaja korišćenja treba po svom izvršenju da napuste sistem koji treba da bude u stanju koje je neophodno za izvršenje sledećeg slučaja korišćenja, bez odlaganja. Da bi se to ostvarilo, potrebno je da postuslovi prethodnog slučaja korišćenja budu identični sa preduslovima narednog slučaja korišćenja. Na slici 6 je prikazano ovo povezivanje postuslova i preduslova navedena tri slučaja korišćenja.



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

Slika 2.6 Primer izjednačavanja postsulova i preduslova radi sekvencijalnog povezivanja tri slučaja korišćenja u jedan slučaj korišćenja "Kupi proizvod".

## SLUČAJEVI KORIŠĆENJA I POSLOVNA PRAVILA

*Poslovna pravila mogu da utiču na naredne korake u normalnom scenariju definišući ispravne ulazne vrednosti ili određujući kako će se izvršavati obračun.*

Slučajevi korišćenja i poslovna pravila su isprepletani. Neka poslovna pravila ograničavaju uloge svih ili pojedinih delova nekog slučaja korišćenja. Na primer, pravilo da samo korisnici sa posebnom dozvolom mogu da izvršavaju alternativne scenarije. To znači da ovo pravilo zahteva da se postavi dati preduslov i da se on testira pre nego što se korisniku dozvoli dalje izvršavanje. **Poslovna pravila mogu da utiču na naredne korake u normalnom scenariju definišući ispravne ulazne vrednosti ili određujući kako će se izvršavati obračuni.**

Prepostavimo da jedan avio prevoznik naplaćuje dodatak na kartu putnika ako on želi da rezerviše posebno sedište u avionu. Kad putnik izvrši slučaj korišćenja u kome on rezerviše sedište u avionu, odgovarajuća poslovna pravila se aktiviraju i menjaju cenu karte putnika, ako je on izabrao određena sedišta sa višom tarifom.

Kada specificirate neko poslovno pravilo, zapišite identifikator onoga ko to poslovno pravilo koje utiče na slučaj korišćenja postavio i zapišite koji deo slučaja korišćenja svako poslovno pravilo zahvata.

Kada istražujete slučajeve korišćenja, možete otkriti značajna poslovna pravila. Kada hemičar koji učestvuje u izazivanju zahteva za Chemist Tracking System, bilo je zahteva da jedan korisnik ne bi trebalo da vidi pravila drugog, i obrnuto. Tako su došli do novog pravila: "Korisnik može da vidi samo pravila za zahteve koje je on postavio".

Ponekad, vi otkrijete pravila za vreme izazivanja zahteva i njihove analize, ponekad unapredite postojeća pravila, a ponekad vi već znate postojeća pravila koja će sistem morati da poštuje.

## ▼ Poglavlje 3

# Utvrđivanje slučajeva korišćenja i zahteva u CTS

## UTVRĐIVANJE SLUČAJEVA KORIŠĆENJA

*Daje se preporuka postupka za utvrđivanje slučajeva korišćenja.*

Možete utvrditi slučajeve korišćenja na nekoliko načina:

1. Prvo utvrdite aktere , onda postavite poslovne procese koje sistem treba da podrži, i onda definišite slučajeve korišćenja za aktivnosti u kojima raspodelu dolazi do interakcija sistema i aktera.
2. Kreirajte specifične scenarije za svaki poslovni proces, onda uopštite scenario za svaki slučaj korišćenja, i utvrdite aktere koji učestvuju u njima.
3. Upotrebom opisa poslovnih procesa, postavite sebi pitanje "Koje zadatke sistem treba da izvrši da bi ostvario do kraja ovaj poslovni proces, ili da konvertuje ulaze u izlaze. Ovi zadaci mogu da postanu slučajevi korišćenja.
4. Utvrdite spoljnje događaje na koje sistem mora da odgovori, a onda povežite te događaje sa odgovarajućim akterima, i specifičnim slučajevima korišćenja.
5. Upotrebite CRUD analizu da bi pronašli unose podataka koje zahtevaju slučajevi korišćenja da bi kreirali, učitali, promenili, obrisali ili nešto drugo radili sa njima.
6. Ispitajte dijagram konteksta i pita se: "Koje ciljeve imaju svi spoljni entiteti, koji žele da ih ostvare, uz pomoć sistema.

CTS je primenio prvi pristup, upotrebljavajući poslovni proces koji je prethodni definisao. Tri biznis analitičara su održali su seriju dvočasovnih radionica, koje su se održavale dva puta nedeljno radi izazivanja zahteva.

Paralelno su održavane radionice sa drugim analitičarima, za različite grupe korisnika, To je moglo da se realizuje jer samo mali slučajeva korisnika zahtevalo učešće korisnika iz različitih grupa (klasa) korisnika.

Pre početka radionice, analitičari su tražili od učesnika da razmisle o zadacima koje bi trebalo da realizuju sa sistemom. Svaki zadatak je postao kandidat za jedan slučaj korišćenja. Dobijena lista mogućih slučajeva korišćenja sprečava da se neki od slučajeva korišćenja previdi.

Neki slučajevi korišćenja su izbačeni jer su van okvira, a neki su spojeni u jedan veći i uopšteniji slučaj korišćenja. Grupe su takođe nalazile i nove slučajeve korišćenja koje prethodni nisu utvrdile iz zadataka.

Nazivi slučajeva korišćenja ukazuju na ciljeve koje imaju slučajevi korišćenja, te zato počinju sa glagolom. Ponekad analitičar mora pitanjem korisnika da utvrdi njihove ciljeve sa pojedinim slučajevima korišćenja, da bi im dao odgovarajući naziv.

Pre ulaženja u detaljnu analizu svakog slučaja korišćenja, prvo ih utvrdite da bi imali celinu pred očima. To vam olakšava određivanje prioriteta i početnu raspodelu slučajeva korišćenja po budućim izdanjima sistem, odn. iteracijama. Onda počnite da analizirate slučajeve korišćenja sa najvećim prioritetom.

## ISTRAŽIVANJE SLUČAJEVA KORIŠĆENJA

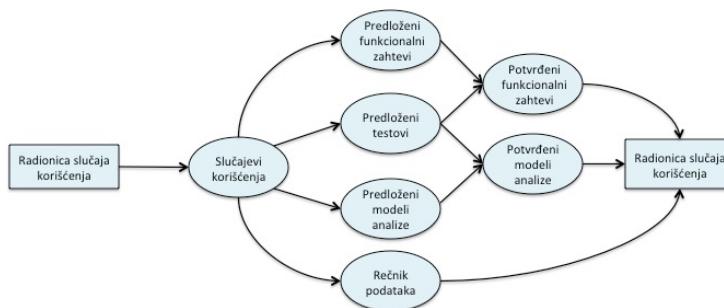
*Na radionicama se utvrđuju svi elementi slučajeva korišćenja. Ili se koriste lepljive beleške, ili se projektuje uzorak specifikacije slučajeva korišćenja*

Učesnici CTS radionica počinjali su rad na svakom slučaju korišćenja diskusijom s ciljem da utvrde aktera koji će imati korist od izvršenja slučaja korišćenja. Procenom frekvencije korišćenja slučaja korišćenja (SK), dobili su prvi indikator istovremenog korišćenja i zahteva kapaciteta.

Zatim su prešli na definisanje preduslova i postuslova, na granicama slučajeva korišćenja. Svi koraci u slučaju korišćenja se nalaze između ovih granica. Tokom diskusije vršile su se popravke ovih uslova. Zatim, analitičari su pitali učesnike kako zamišljaju interakciju sa sistemom za vreme izvršenja zadatka. Dobijeni redosled akcija aktera odredio je normalni scenario slučaja korišćenja. Analitičari su beležili predloge akcija aktera na lepljivim beleškama koje su lepili na tablu ili papir. Njihovim premeštanjem lako se vrši njihovo grupisanje ili zamena. Drugi način je da se projektuje formular za specifikaciju slučaja korišćenja i onda da ga grupa kolektivno popunjava. Sličan postupak je primenjen i za alternativne scenarije i izuzetke. Mnogi izuzeci su otkriveni kada je analitičar pitao "Šta bi se desilo ako bi u jednom trenutku prestala veza sa bazom podataka?" ili " Šta bi se desilo ako hemikalija nije komercijalno raspoloživa". Rad u grupi omogućava i da se govori i o očekivanom kvalitetu sistema, u vidu vremena odziva, raspoloživosti sistema, zahteva bezbednosti, i projektnih ograničenja korisničkog interfejsa.

Radionice su prvo radili sa SK sa većim prioritetom, i u više sesija.

Slika 1 prikazuje redosled dobijenih rezultata rada za vreme procesa utvrđivanja slučajeva korišćenja za slučaj CTS. Posle radionice, analitičar je na osnovu rezultata radionice, popunio specifikaciju slučajeva korišćenja, procenjujući potreban nivo detaljnosti specifikacije SK. Proces na slici 1 pokazuje da su nakon svake radionice BA iz sistema za praćenje hemikalija proizveli funkcionalne zahteve softvera iz slučajeva upotrebe. BA su takođe izradili neke modele analize, kao što je dijagram prelaska stanja koji je pokazao sve moguće statuse hemijskih zahteva i dozvoljene promene statusa. Slučajevi višestruke upotrebe mogu manipulirati hemijskim zahtevom, pa dijagram objedinjuje informacije i operacije koje obuhvataju nekoliko slučajeva upotrebe



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

Slika 3.1 Rezultati utvrđivanja slučajeva korišćenja na radionicama

## POTVRĐIVANJE SLUČAJEVA KORIŠĆENJA

*Umesto jednog načina predstavljanja zahteva dobijenih iz slučaja korišćenja, treba koristiti više načina predstavljanja, jer se tada otkrivaju više grešaka.*

Dan ili dva nakon svake radionice, BA je dao slučajeve upotrebe i funkcionalne zahteve učesnicima radionice, koji su ih pregledali pre sledeće radionice. Ovi neformalni pregledi otkrili su mnogo grešaka: prethodno neotkriveni alternativni tokovi, novi izuzeci, pogrešni funkcionalni zahtevi i nedostajući koraci dijaloga. Tim je brzo naučio da dozvoli bar jedan dan između uzastopnih radionica. Mentalno opuštanje koje dolazi nakon dan ili dva omogućava ljudima da pregledaju svoj raniji rad iz sveže perspektive. Jedan BA koji je održavao svakodnevne radionice utvrdio je da su učesnici imali poteškoća da uoče greške u pregledanim materijalima, jer su im informacije bile previše sveže. Mentalno su recitirali nedavnu diskusiju i nisu videli greške. Početkom razvoja potreba, testni sistem sistema za praćenje hemijskih proizvoda počeo je kreirati konceptualne testove - nezavisno od specifičnosti primene i korisničkog interfejsa - od slučajeva primene. Ovi testovi su pomogli tim da postigne zajedničko razumevanje kako se sistem treba ponašati u određenim scenarijima. Testovi omogućavaju BA-ima da provere jesu li izvukli funkcionalnost koja je potrebna kako bi korisnici izvršili svaki slučaj upotrebe. Tokom završne radionice o izlasku, učesnici su zajedno prolazili testove kako bi bili sigurni da su se dogovorili o tome kako treba da rade slučajevi upotrebe.

Rano konceptualno probno razmišljanje poput ovog mnogo je jeftinije i brže od pisanja koda, izgradnje dela sistema, izvođenja testova i tek tada otkrivanja problema sa zahtevima. To je analogno agilnom pristupu izmišljanja korisničkih priča sa testovima prihvatanja, ali CTS tim je napisao i funkcionalne zahteve i testove. Izvršeno je upoređivanje dve otkrivene greške pre nego što je bilo koji kod napisan. CTS tim stvorio je višestruko predstavljanje zahteva koje su identifikovali: spisak funkcionalnih zahteva, skup odgovarajućih testova i modele analize, a sve zasnovano na slučajevima korišćenja. Upoređivanje ovih alternativnih stavova sa zahtevima je moćna tehnika kvaliteta. Tim je koristio testove za verifikaciju funkcionalnih zahteva, tražeći testove koji se ne mogu „izvršiti“ sa setom zahteva i zahtevima koji nisu obuhvaćeni testovima.

Ako stvorite samo jedno predstavljanje ili jedan pogled na zahteve, morate joj verovati. Nemate s čim da ga uporedite radi traženja grešaka, praznina i različitih interpretacija.

Agilni projektni timovi obično ne dokumentuju funkcionalne zahteve, radije stvaraju testove prihvatanja. Iako je razmišljanje o testiranju tokom istraživanja zahteva odlična ideja za svaki projekat, ipak vam ostavlja samo jedan prikaz zahteva za koje morate verovati da su tačni. Slično tome, tradicionalni projektni timovi koji stvaraju samo niz funkcionalnih zahteva i testiranje ostave za kasnije u projektu imaju samo jedno predstavljanje. Dobićete najbolje rezultate razumnom kombinacijom pismenih zahteva, testova, modela analize i prototipa.

## SLUČAJI KORIŠĆENJA I FUNKCIONALNI ZAHTEVI

*Slučajevi upotrebe opisuju perspektivu korisnika, sagledavajući spoljno vidljivo ponašanje sistema. Oni ne sadrže sve informacije potrebne programeru.*

Programeri softvera ne primenjuju poslovne zahteve ili zahteve korisnika. Oni implementiraju funkcionalne zahteve, specifične bitove ponašanja sistema. Neki praktičari smatraju da su slučajevi upotrebe funkcionalni zahtevi. Međutim, videli smo da mnoge organizacije upadaju u probleme kada jednostavno prenesu svoje primene namenjene programerima na implementaciju. Slučajevi upotrebe opisuju perspektivu korisnika, sagledavajući spoljno vidljivo ponašanje sistema. Oni ne sadrže sve informacije potrebne programeru da bi napisao softver. Korisnik bankomata ne zna za bilo kakvu pozadinsku obradu, poput komunikacije sa računаром banke. Ovaj detalj je korisniku nevidljiv, ali programer mora znati za njega. Programeri koji primaju čak i potpuno opisane slučajeve upotrebe često imaju mnogo pitanja. Da biste smanjili ovu nesigurnost, razmislite o tome da BA ima izričito specificiranje funkcionalnih zahteva neophodnih za implementaciju svakog slučaja upotrebe.

Mnogi funkcionalni zahtevi ispadaju iz dijaloških koraka između aktera i sistema. Neke su očigledni, kao što je „Sistem će svakom zahtevu dodeliti jedinstveni redni broj.“ Nema smisla da ih duplirate negde drugde ako su to jasni zahtevi iz slučaja upotrebe. Ostali funkcionalni zahtevi se ne prikazuju u opisu slučaja upotrebe. Na primer, način na koji se dokumentiraju slučajevi upotrebe obično ne precizira šta sistem treba da radi ako nije ispunjen preduslov.

Ovo je primer kako slučajevi upotrebe često ne pružaju sve potrebne informacije programeru da zna šta da gradi. BA mora izvući te nedostajuće zahteve i dostaviti ih programerima i ispitivačima. Analiza dobijanja pogleda programera na zahteve na osnovi pogleda korisnika na zahteve, treba da izvrši BA i time da doda vrednost projektu

Sistem praćenja hemikalija koristio je slučajeve upotrebe prvenstveno kao alat za otkrivanje potrebnih funkcionalnih potreba. Analitičari su napisali samo povremene opise manje složenih slučajeva upotrebe. Zatim su izvukli sve funkcionalne zahteve koji bi, kada se primene, omogućili akteru da izvrši slučaj upotrebe, uključujući alternativne tokove i rukovaće izuzetaka. Analitičari su ove funkcionalne zahteve dokumentovali u SRS-u, koji je organizovan kao svojstvo proizvoda.

Funkciju povezану са slučajем коришћења можете документовати на више начина. Ниједна од следећих метода nije савршена, па одаберите приступ који најбоље одговара начину на који ћете да документujete и управљате softverskim zahtevима ваšeg projekta.

# NAČINI DOKUMENTOVANJA ZAHTEVA NA OSNOVU SLUČAJEVA UPOTREBE

*Postoji nekoliko načina dokumentovanja funkcionalnih zahteva koji su proizišli iz slučajeva korišćenja.*

## **Koristite samo slučajeve upotrebe**

Ovo je jedna od mogućnosti je da se uključe funkcionalni zahtevi uz svaku specifikaciju slučaja upotrebe, ako već nisu očigledni. I dalje ćete morati da dokumentujete nefunkcionalne zahteve i bilo koju funkciju koja nije povezana sa slučajem korišćenja. Pored toga, može d se desi da nekoliko slučajeva upotrebe koristi isti funkcionalni zahtev. Ako pet slučajeva upotrebe zahteva da se identitet korisnika potvrdi, ne želite da pišete pet različitih blokova koda u tu svrhu. Umesto da ih duplira, funkcionalni zahtevi koji se pojavljuju u višestrukim slučajevima koriste se unakrsnim referencama. Slučajevi upotrebe mogu se prikupiti u dokumentu o potrebama korisnika.

## **Koristite slučajeve upotrebe i funkcionalne zahteve**

Druga opcija je pisanje prilično jednostavnih slučajeva korišćenja i dokumentovanje funkcionalnih zahteva proisteklih iz svakog u SRS-u ili skladištu zahteva. U ovom pristupu trebalo bi da uspostavite sledljivost između slučajeva upotrebe i njihovih povezanih funkcionalnih potreba. Na taj način, ako se slučaj upotrebe promeni, brzo možete pronaći pogodjene funkcionalne zahteve. Najbolji način za upravljanje sledljivošću je korišćenje alata za upravljanje zahtevima.

## **Samo funkcionalni zahtevi**

Još jedna opcija je da organizujete svoje funkcionalne zahteve u slučaju upotrebe ili po karakteristikama i da uključite i slučajeve upotrebe i funkcionalne zahteve u SRS ili skladište zahteva. Ovo je pristup koji je koristio CTS tim, a isto smo uradili i na nekoliko projekata za razvoj veb lokacija. Pisali smo većinu naših slučajeva u vrlo sažetom obliku, ne dovršavajući ceo obrazac sa slike 2 u poglavlju 2. Detalji su zatim specificirani kroz skup funkcionalnih zahteva. Ovaj pristup ne rezultira posebnim dokumentom o potrebama korisnika.

## **Koristite slučajeve upotrebe i testove**

Ako napišete detaljne specifikacije slučajeva upotrebe i funkcionalne zahteve, možda ćete primetiti neko dupliranje, posebno oko normalnog protoka. Mala je vrednost pisanja istog zahteva dva puta. Dakle, druga strategija je pisati prilično kompletne specifikacije slučaja upotrebe, ali zatim napisati testove prihvatanja kako biste utvrdili da li sistem pravilno postupa sa osnovnim ponašanjem slučaja upotrebe, alternativnim putevima uspeha i različitim stvarima koje mogu poći po zlu.

## IZBEGNITE NEKE ZAMKE PRI RADU SA SLUČAJEVIMA KORIŠĆENJA

*Daje se pet preporuka za izbegavanje zamki pri kreiranju slučajeva korišćenja.*

- **Koristite isuviše mnogo slučajeva korišćenja:** Ne pišite poseban slučaj korišćenja za svaki scenario. Imaćete mnogo više slučajeva korišćenja nego poslovnih zahteva i svojstava, ali i mnogo više funkcionalnih zahteva nego slučajeva korišćenja.
- **Upotreba vrlo složenih slučajeva korišćenja:** **Vrlo** složeni slučajevi korišćenja nisu razumljivi. Ako ne možete da pojednostavite poslovne zadatke, možete da pojednostavite njihovo predstavljanje u slučajeva korišćenja. Izaberite jedan uspešan put slučaja korišćenja i programirajte ga normalnim tokom, tj. scenarijom. Koristite alternativne tokove (scenarije) i izuzetke za opis grananja u tokovima. Možete imati puno alternativnih tokova, ali će oni biti kratki i laki za razumevanje. Ako tok ima više od 10 do 15 koraka (akcija), vidite da li on stvarno predstavlja samo jedan scenario. Međutim nemojte ga slobodno preseći, samo zato što je dugačak.
- **Uključivanje projektovanje u slučajeve korišćenja:** Slučajevi korišćenja bi trebalo da se usmere ka opisivanju ŠTA korisnik očekuje od sistema, a ne KAKO sistem to treba da izvrši, ili kako će izgledati prikaz na monitoru. Koristite koncepcione interakcije između aktera i sistema. Možete koristiti skice izgleda prikaza na monitoru, ali samo radi vizualizacije interakcije aktera i sistema, a ne kao specifikaciju projektnog rešenja.
- **Uključivanje definisanje podataka u slučajeve korišćenja:** Neki autori uključuju definisanje značajnih podataka u slučajevima korišćenja. To otežava njihovo nalaženje, jer nije jasno koji slučaj korišćenja koristi koji podatak. Takođe, može voditi i duploj specifikaciji istog podatka. Definicije podataka treba posebno skladištiti u rečniku podataka i u modelu podataka.
- **Slučajevi korišćenja koje korisnici ne razumeju:** Nejasne i efektivne komunikacije. Problem je ako korisnici ne mogu da povežu neki slučaj korišćenja sa njihovim poslovnim procesom i ciljevima. Pišite slučajeve korišćenja iz perspektive korisnika, ne iz perspektive sistema, i pitajte korisnike da ih recenziraju. Koristite jednostavne slučajeve korišćenja, a da mogu da ostvare postavljene ciljeve.

## PREDNOSTI KORIŠĆENJA ZAHTEVA KONCENTRISANIH NA KORISNIKA

*Slučajevi korisnika pomažu da se razjasne razne nejasnoće i protivurečnosti u ranim fazama razvoja sistema, a i mogu da generišu testove na osnovu slučajeva korišćenja.*

Snaga slučajeva korišćenja i priča korisnika je u njihovoj perspektivi klasa korisnika, koncentrisanoj ka korisniku. Korisniku će biti jasnije šta može da očekuje od sistema, nego da je primenjen koncept slučajeva korišćenja koji je usmeren ka svojstvima (feature-based).

Slučajevi korišćenja pomažu analitičarima i inženjerima razvoja da razumeju posao korisnika. Slučajevi korisnika im pomažu da razjasne razne nejasnoće i protivurečnosti u ranim fazama razvoja sistema, a i mogu da generišu testove na osnovu slučajeva korišćenja.

Specificiranje prevelikog broja dostižnih funkcionalnih zahteva unapred, dovodi do primene i neposrednih zahteva. Fokus na korisnika pre kreiranju slučajeva korišćenja, dovodi do sistema koji dozvoljava korisniku da izvrši određene zadatke, koji su verifikovani.

Razvoj zahteva korisnika pomaže u određivanju prioriteta zahteva. Najveći prioritet dobijaju funkcionalni zahtevi koji proizilaze iz korisničkih zahteva koji imaju najviši prioritet, kao što su:

- oni koji opisuju ključne poslovne procese koje sistem treba da podrži,
- oni koje koriste često veći broj korisnika,
- oni koje zahteva glavna klasa korisnika
- oni koji treba da zadovolje propise.

Slučajevi korišćenja donose i tehničku korist. Oni otkrivaju važne objekte domena i njihove međusobne odnose. Inženjeri razvoja objektno-orientisanih sistema mogu da mapiraju slučajeve korišćenja u modela objekata i sekvenčalne dijagrame.

Kako se poslovni procesi vremenom menjaju, i zadaci koji se vezuju za pojedine zahteve korisnika se vremenom menjaju. Ako povežete funkcionalne zahteve, projektna rešenja, kod, i testove, sa njihovim zahtevima korisnika od kojih su potekli, onda možete onda možete da kaskadno povežete ove promene zahteva korisnika kroz ceo sistem. To znači jači "glas kupca".

## VIDEO 11 - WHAT IS A USE CASE - WIEGERS (VIDEO)

*Trajanje: 6:56 minuta*

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

## VIDEO 12 - USE CASES, SCENARIOS, AND STORIES - WIEGERS (VIDEO)

*Trajanje: 4:57 minuta*

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

## VIDEO 13 - USE CASES AND FUNCTIONAL REQUIREMENTS - WIEGERS (VIDEO)

*Trajanje: 4:53 minuta*

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

## ▼ Poglavlje 4

### Vežba

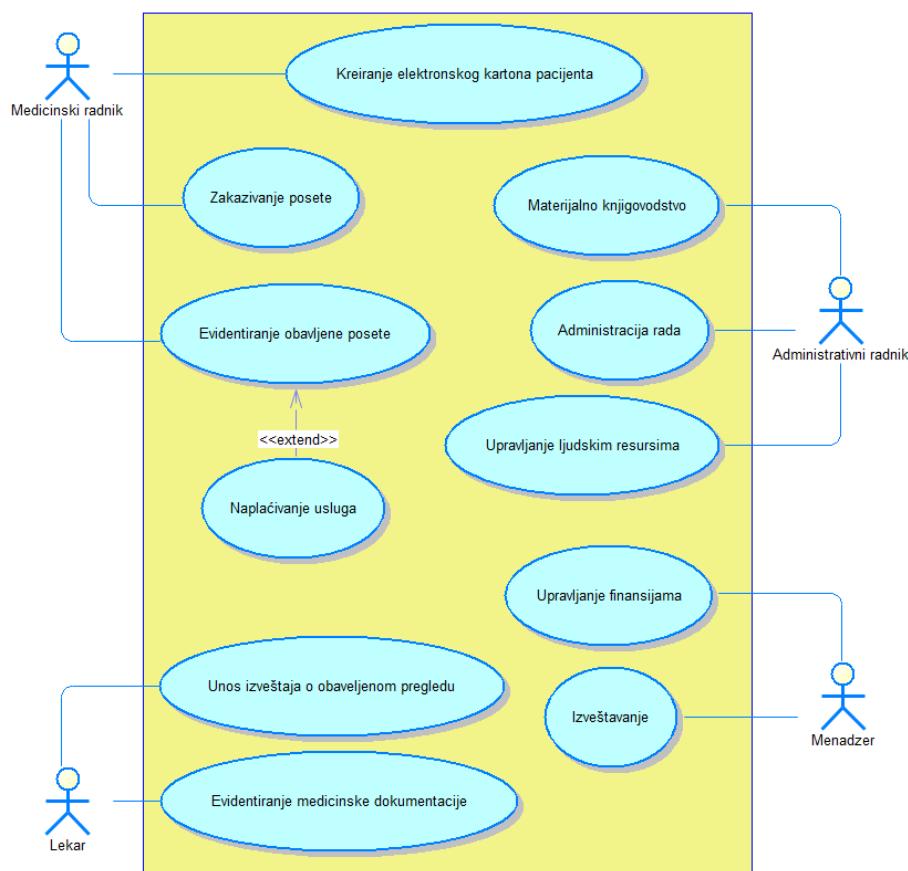
#### DIJAGRAM SLUČAJEVA KORIŠĆENJA

##### *Dijagram slučajeva korišćenja za poslovni sistem privatne klinike*

U ovoj vežbi je izrađen dijagram slučajeva korišćenja (slika 1), koji predstavlja osnov za razumevanje potreba korisnika i za utvrđivanje njegovih zahteva. Skiciranje slučajeva upotrebe na ovaj način biće ispraćeno i definisanjem scenarija, tj. tokova akcija, koje se moraju izvršiti u svakom pojedinačnom slučaju korišćenja, radi utvrđivanja funkcionalnih zahteva u vežbama koje slede.

Slučaj korišćenja prikuplja sve scenarije, pokazujući sve načine na koje se cilj može uspešno postići ili neuspešno završiti. Ukoliko je neki korak u slučaju korišćenja složen i može uspeti ili ne uspeti, treba ga predstaviti kao pod slučaj korišćenja. Da biste slučajevе korišćenja pravilno identifikovali, o njima treba razmišljati kao o priči ili igri koja se neprekidno odvija. Ako se ta priča ilustruje, a zatim prateći ilustraciju, napiše u tačnim koracima, umnogome će pomoći svim učesnicima na razvoju bilo kog softverskog proizvoda.

Opisi slučajeva korišćenja za aktera Medicinski radnik, uslediće na slikama 2-9.



Slika 4.1 Dijagram slučajeva korišćenja za poslovni sistem privatne klinike

## OPISI SLUČAJA KORIŠĆENJA UC-1 ZA POSLOVNI SISTEM PRIVATNE KLINIKE

*UC-1: Kreiranje elektronskog kartona pacijenta*

ID i naziv:	<b>UC-1 Kreiranje elektronskog kartona pacijenta</b>		
Kreator:	Marina Damnjanović	Datum kreiranja:	08/09/19
Primarni akter:	Medicinski radnik	Sekundarni akteri:	Lekar
Pokretač:	Medicinski radnik treba da kreira elektronski karton za pacijenta koji nikada nije lečen u klinici		
Opis:	Medicinskom radniku treba omogućiti unos podataka koji se traže za formiranje elektronskog kartona pacijenta i čuvanje kartona za dalju upotrebu.		
Preduslovi:	PRE-1. Postoji konekcija na internet. PRE-2. Baza podataka je onlajn. PRE-3. Medicinski radnik je autentifikovan pomoću svojih podataka za pristup sistemu.		
Postuslovi:	POST-1. Elektronski karton novog pacijenta je uspešno sačuvan. POST-2. Pacijentu se može zakazati pregled.		
Normalni tok:	<b>1.0 Kreiranje elektronskog kartona novog pacijenta</b> <ol style="list-style-type: none"> <li>Medicinski radnik zahteva kreiranje novog pacijenta</li> <li>Sistem daje formu za evidentiranje ličnih podataka, prebivališta, kontakt podataka, podataka o staratelju i dodavanje komentara.</li> <li>Sistem daje mogućnost unosa svih podatke ručno ili preko čitača lične karte (pogledati 1.1)</li> <li>Medicinski radnik unosi potrebne podatke i potvrđuje unos klikom na dugme za čuvanje</li> <li>Medicinski radnik traži napuštanje slučaja korišćenja ili traži unos dodatnih podataka (pogledati 1.2)</li> <li>Sistem čuva elektronski karton novog pacijenta.</li> </ol>		

Slika 4.2 UC-1 Kreiranje elektronskog kartona pacijenta [Izvor: Marina Damnjanović]

Alternativni tokovi:	<b>1.1 Unos ličnih podataka preko čitača lične karte</b> <ol style="list-style-type: none"> <li>Medicinski radnik bira učitavanje ličnih podataka direktno iz lične karte.</li> <li>Sistem daje signal da je spreman za očitavanje lične karte. (pogledati 1.1.E1)</li> <li>Medicinski radnik unosi ličnu kartu u čitač i pokreće očitavanje podataka.</li> <li>Sistem obrađuje podatke i daje signal kada su podaci očitani i lična karta se može skloniti sa čitača.</li> <li>Medicinski radnik odstranjuje ličnu kartu iz čitača.</li> <li>Medicinski radnik menja podatke ako je potrebno i potvrđuje unos klikom na dugme za čuvanje.</li> <li>Sistem čuva elektronski karton novog pacijenta.</li> </ol> <b>1.2 Unos dodatnih podataka o pacijentu</b> <ol style="list-style-type: none"> <li>Medicinski radnik traži unos podataka o privatnom zdravstvenom osiguranju.</li> <li>Sistem daje pretragu postojećih osiguravajućih kuća, tipa osiguranja i mogućnost unosa osiguravajućeg koda.</li> <li>Medicinski radnik unosi podatke dobijene od pacijenta i zahteva čuvanje.</li> <li>Sistem ažurira elektronski karton novog pacijenta.</li> <li>Medicinski radnik traži unos podataka o rizicima i alergijama.</li> <li>Sistem daje forme za unos podataka o tipovima rizika ili alergija i vremenskom trajanju.</li> <li>Medicinski radnik unosi podatke dobijene od pacijenta i zahteva čuvanje.</li> <li>Sistem ažurira elektronski karton novog pacijenta.</li> </ol>
Izuzeci:	<b>1.1.E1 Aplikacija za očitavanje lične karte nije aktivna</b> <ol style="list-style-type: none"> <li>Sistem prikazuje poruku da aplikacija za očitavanje lične karte nije trenutno aktivna.</li> <li>Sistem vraća korisnika na formu za ručni unos ličnih podataka.</li> <li>Medicinski radnik unosi potrebne podatke i zahteva čuvanje.</li> <li>Sistem čuva elektronski karton novog pacijenta.</li> </ol>
Prioritet:	Visok
Frekvencija upotrebe:	Minimum 1 dnevno, maksimum 10 dnevno. Na nedeljnjenom nivou, minimum 6 puta, maksimum 60 puta.
Poslovna pravila:	...
Druge informacije:	Treba da postoji mogućnost učitavanja ličnih podataka iz čipovane lične karte. Čitač lične karte mora da bude integrisan sa sistemom i treba obezbediti aplikaciju koja omogućava čitanje podataka.
Prepostavke:	Prepostavlja se da je lična karta važeća. Prepostavlja se da je čitač lične karte ispunjen.

Slika 4.3 UC-1 Kreiranje elektronskog kartona pacijenta [Izvor: Marina Damnjanović]

# OPISI SLUČAJA KORIŠĆENJA UC-2 ZA POSLOVNI SISTEM PRIVATNE KLINIKE

## *UC-2: Zakazivanje posete*

ID i naziv:	<b>UC-2 Zakazivanje posete</b>		
Kreator:	Marina Damnjanović	Datum kreiranja:	08/09/19
Primarni akter:	Medicinski radnik	Sekundarni akteri:	Lekar
Pokretač:	Medicinski radnik treba da zakaže posetu pacijentu koji ima kreiran elektronski zdravstveni karton u sistemu privatne klinike.		
Opis:	Medicinskom radniku treba omogućiti zakazivanje posete lekaru u terminu u kome je lekar dostupan za rad, u skladu sa važećim rasporedom.		
Preduslovi:	PRE-1. Postoji konekcija na internet. PRE-2. Baza podataka je onlajn. PRE-3. Medicinski radnik je autentifikovan pomoću svojih podataka za pristup sistemu. PRE-4. Pacijent ima kreiran elektronski zdravstveni karton u sistemu privatne klinike.		
Postuslovi:	POST-1. Pacijentu je uspešno zakazana poseta adekvatnom lekaru u slobodnom terminu.		
Normalni tok:	<b>2.0 Zakazivanje posete lekaru</b> <ol style="list-style-type: none"> <li>Medicinski radnik zahteva prikaz rasporeda rada za predstojeći period.</li> <li>Sistem daje prikaz opšteg rasporeda rada u formi kalendarja.</li> <li>Medicinski radnik bira dan kada pacijent želi da zakaže posetu</li> <li>Medicinski radnik proverava dostupnost lekara za odabrani dan i slobodne termine.</li> <li>Medicinski radnik zahteva da zakaže posetu za odabranog lekara i odabrani termin. (pogledati 2.0.E1 i 2.0.E2)</li> <li>Sistem traži od medicinskog radnika da unese JMBG pacijenta kome želi da zakaže posetu.</li> <li>Sistem nalazi registrovanog pacijenta ili zahteva od medicinskog radnika da prvo kreira elektronski karton za novog pacijenta. (pogledati 2.1)</li> <li>Sistem uspešno čuva rezervisanu posetu.</li> </ol>		

Slika 4.4 UC-2 Zakazivanje posete [Izvor: Marina Damnjanović]

Alternativni tokovi:	<b>2.1 Zakazivanje posete lekaru pacijentu koji nema otvoren karton u sistemu privatne klinike</b> <ol style="list-style-type: none"> <li>1. Sistem prikazuje poruku da pacijentu nema otvoren karton u sistemu privatne klinike.</li> <li>2. Sistem daje mogućnost medicinskom radniku da kreira elektronski karton novom pacijentu (3a) ili da prekine proces zakazivanja posete (4b).</li> <li>3.a. Medicinski radnik bira da kreira elektronski karton novom pacijentu</li> <li>3.b. Sistem pokreće izvršenje slučaja korišćenja UC-1.</li> <li>3.c. Medicinski radnik kreira elektronski karton novom pacijentu.</li> <li>3.d. Sistem preusmerava medicinskog radnika da nastavi sa izvršenjem tekućeg slučaja korišćenja.</li> <li>4.a. Medicinski radnik bira da prekine proces zakazivanja.</li> <li>4.b. Sistem prekida izvršenje slučaja korišćenja.</li> </ol>
Izuzeći:	<b>2.0.E1 Željeni lekar nije u rasporedu rada za odabrani dan</b> <ol style="list-style-type: none"> <li>1. Sistem ispisuje poruku da traženi lekar nije u rasporedu za odabrani dan.</li> <li>2. Medicinski radnik ponovo izvršava normalan tok tekućeg slučaja korišćenja.</li> <li>3. Medicinski radnik vrši pretragu rasporeda rada prema željenom lekaru kako bi pronašao prvi slobodan termin kod istog lekara.</li> </ol> <b>2.0.E2 Traženi lekar nema više dostupnih termina za odabrani dan</b> <ol style="list-style-type: none"> <li>1. Sistem ispisuje poruku da traženi lekar nema više dostupnih termina za odabrani dan.</li> <li>2. Medicinski radnik ponovo izvršava normalan tok tekućeg slučaja korišćenja.</li> <li>3. Medicinski radnik vrši pretragu rasporeda rada prema željenom lekaru kako bi pronašao prvi sledeći slobodan termin kod istog lekara.</li> </ol>
Prioritet:	Visok
Frekvencija upotrebe:	U proseku 20 puta na dan, a maksimalno 200 puta na nedeljnom nivou.
Poslovna pravila:	...
Druge informacije:	/
Prepostavke:	Podrazumeva se da su dnevni rasporedi unapred kreirani. Dnevni rasporedi su kreirani na osnovu rasporeda rada angažovanog osoblja, rasporeda radnih/heradnih dana i radnog vremena klinike. Podrazumeva se da su rasporedi usaglašeni sa regularnim godišnjim kalendarima.

Slika 4.5 UC-2 Zakazivanje posete [Izvor: Marina Damnjanović]

## OPISI SLUČAJA KORIŠĆENJA UC-3 ZA POSLOVNI SISTEM PRIVATNE KLINIKE

*UC-3: Evidentiranje realizovane posete*

ID i naziv:	<b>UC-3 Evidenciranje realizovane posete</b>		
Kreator:	Marina Damnjanović	Datum kreiranja:	08/09/19
Primarni akter:	Medicinski radnik	Sekundarni akteri:	Lekar, Menadžer
Pokretač:	Medicinski radnik treba da evidentira da je poseta lekaru realizovana.		
Opis:	Medicinskom radniku treba omogućiti unos podataka o realizaciji posete lekaru koja je prethodno bila zakazana.		
Preduslovi:	PRE-1. Postoji konekcija na internet. PRE-2. Baza podataka je onlajn. PRE-3. Medicinski radnik je autentifikovan pomoću svojih podataka za pristup sistemu. PRE-4. Poseta lekaru je prethodno bila zakazana.		
Postuslovi:	POST-1. Medicinski radnik je uneo potvrdu o realizaciji zakazane posete.		
Normalni tok:	<p><b>3.0 Evidenciranje realizovane posete lekaru</b></p> <ol style="list-style-type: none"> <li>Medicinski radnik zahteva prikaz rasporeda poseta za dan.</li> <li>Sistem daje dnevni raspored poseta za dan.</li> <li>Medicinski radnik bira posetu za koju treba potvrditi realizaciju. (pogledati 3.0.E1 i 3.0.E2)</li> <li>Medicinski radnik unosi podatke o trajanju posete.</li> <li>Medicinski radnik proverava da li ima uputa za druge lekare nakon obavljenog pregleda i zakazuje nove preglede, ako ih ima. (pogledati 3.1)</li> <li>Medicinski radnik proverava da li ima usluga za naplatu i izvršava naplaćivanje usluga, ako ih ima (pogledati UC-4)</li> <li>Medicinski radnik traži da sačuva evidenciju o obavljenoj poseti klikom na dugme.</li> <li>Sistem ažurira istoriju dnevnih poseta za taj dan.</li> <li>Sistem ažurira istoriju poseta pacijenta u okviru njegovog elektronskog kartona.</li> </ol>		

Slika 4.6 UC-3 Evidenciranje realizovane posete [Izvor: Marina Damnjanović]

Alternativni tokovi:	<p><b>3.1 Zakazivanje dodatnih poseta za pacijenta koji je obavio pregled kod lekara</b></p> <ol style="list-style-type: none"> <li>Medicinski radnik zahteva izvršenje slučaja korišćenja UC-2.</li> <li>Sistem pokreće normalan tok slučaja korišćenja UC-2.</li> <li>Medicinski radnik izvršava slučaj korišćenja UC-2.</li> <li>Sistem preusmerava medicinskog radnika da nastavi sa izvršenjem tekućeg slučaja korišćenja.</li> </ol>
Izuzeci:	<p><b>3.0.E1 Pacijent je hitan slučaj i nije imao kreiranu rezervaciju posete</b></p> <ol style="list-style-type: none"> <li>Medicinski radnik zahteva unos hitne posete.</li> <li>Sistem daje formu za evidentiranje hitne posete.</li> <li>Medicinski radnik unosi podatke o trajanju hitne posete i beleži koji je lekar preuzeo pacijenta.</li> <li>Sistem traži od medicinskog radnika da unese JMBG pacijenta koji je imao hitnu posetu.</li> <li>Sistem nalazi registrovanog pacijenta (5.a) ili zahteva od medicinskog radnika da prvo kreira elektronski karton za novog pacijenta (6.a).</li> <li>a. Medicinski radnik proverava ostale podatke o obavljenom pregledu i vrši naplatu usluga.</li> <li>b. Sistem pokreće izvršenje slučaja korišćenja UC-4.</li> <li>a. Sistem pokreće izvršenje slučaja korišćenja UC-1.</li> <li>b. Medicinski radnik kreira elektronski karton novom pacijentu.</li> <li>c. Sistem preusmerava medicinskog radnika da nastavi sa izvršenjem tekućeg slučaja korišćenja.</li> </ol> <p><b>3.0.E1 Pacijent se nije pojavio ili je prethodno otkazao pregled</b></p> <ol style="list-style-type: none"> <li>Medicinski radnik označava termin kao ponovo slobodan i čuva promenu.</li> <li>Sistem prikazuje termin prikazuje kao slobodan i dozvoljava novu rezervaciju posete u tom terminu.</li> </ol>
Prioritet:	Srednji
Frekvencija upotrebe:	U proseku 30 puta na dan, a maksimalno 200 puta na nedeljnou nivou.
Poslovna pravila:	...
Druge informacije:	/
Prepostavke:	/

Slika 4.7 UC-3 Evidentiranje realizovane posete [Izvor: Marina Damnjanović]

## OPISI SLUČAJA KORIŠĆENJA UC-4 ZA POSLOVNI SISTEM PRIVATNE KLINIKE

### *UC-4: Naplaćivanje usluga*

ID i naziv:	<b>UC-4 Naplaćivanje usluga</b>		
Kreator:	Marina Damnjanović	Datum kreiranja:	08/09/19
Primarni akter:	Medicinski radnik	Sekundarni akteri:	Lekar, Menadžer
Pokretač:	Medicinski radnik treba da naplati usluge koje je evidentirao u sistemu.		
Opis:	Medicinskom radniku treba omogućiti da može pacijentu da naplati korišćene medicinske usluge, prema važećem cenovniku klinike, i da mu izda dokaz o plaćanju u vidu fiskalnog računa.		
Preduslovi:	PRE-1. Postoji konekcija na internet. PRE-2. Baza podataka je onlajn. PRE-3. Fiskalna kasa je u funkciji. PRE-4. Fiskalna kasa je uključena.		
Postuslovi:	POST-1. Naplata usluga je uspešno izvršena i evidentirana u sistemu. POST-2. Pacijent je dobio račun kao potvrdu plaćanja.		
Normalni tok:	<b>4.0 Naplaćivanje pruženih medicinskih usluga pacijentu</b> <ol style="list-style-type: none"> <li>Medicinski radnik zahteva da naplati usluge koje su evidentirane u okviru posete pacijenta.</li> <li>Sistem daje medicinskom radniku pregled stavki koje će biti naplaćene prema važećem cenovniku.</li> <li>Sistem daje medicinskom radniku da odabere način na koji će pacijentu naplatiti usluge: u gotovini ili karticom (videti 4.1)</li> <li>Medicinski radnik bira način plaćanja i traži izdavanje računa.</li> <li>Sistem obrađuje naplatu i komunicira sa fiskalnom kasom, na kojoj se štampa račun.</li> <li>Sistem prikazuje poruku o uspešnom naplaćivanju usluga.</li> <li>Sistem ažurira podatke o plaćanjima.</li> </ol>		

Slika 4.8 UC-4 Naplaćivanje usluga [Izvor: Marina Damnjanović]

Alternativni tokovi:	<b>4.1 Plaćanje se vrši karticom</b> <ol style="list-style-type: none"> <li>1. Medicinski radnik bira naplaćivanje usluga sa kartice.</li> <li>2. Sistem traži unos iznosa za naplatu na POS terminalu.</li> <li>3. POS terminal traži od medicinskog radnika da prisloni platnu karticu na POS terminal radi izvršenja transakcije. (videti 4.1.E1 i 4.1.E2)</li> <li>4. POS terminal obrađuje transakciju.</li> <li>5. Transakcija uspešno prolazi, na fiskalnoj kasi se štampa račun.</li> <li>6. Sistem prikazuje poruku o uspešnom naplaćivanju usluga.</li> <li>7. Sistem ažurira podatke o plaćanjima.</li> </ol>
Izuzeci:	<b>4.1.E1 Transakcija je neuspešna zbog prekida signala na POS terminalu</b> <ol style="list-style-type: none"> <li>1. Sistem prikazuje poruku da je došlo do greške u komunikaciji sa POS terminalom.</li> <li>2. Sistem daje mogućnost medicinskom radniku da pokuša ponovo da obavi transakciju (3.a) ili da naplati u gotovini (4.a)</li> <li>3.a. Medicinski radnik ponovo izvršava alternativni tok 4.1</li> <li>4.a. Medicinski radnik bira naplatu u gotovini i direktno štampanje računa.</li> </ol> <b>4.1.E1 Transakcija je neuspešna zbog nedovoljno sredstava na računu pacijenta</b> <ol style="list-style-type: none"> <li>3. Sistem prikazuje poruku da je došlo do greške zbog nedovoljnog stanja na računu pacijenta.</li> <li>4. Sistem daje mogućnost medicinskom radniku da ga preusmeri na naplatu usluga u gotovini.</li> </ol>
Prioritet:	Visok
Frekvencija upotrebe:	U proseku 30 puta na dan, a maksimalno 200 puta na nedeljnom nivou.
Poslovna pravila:	...
Druge informacije:	Mora da postoji integracija sistema sa fiskalnom kasom. Mora da postoji integracija POS terminala i sistema.
Prepostavke:	Fiskalna kasa je u ispravnom stanju. POS terminal je u ispravnom stanju. <i>Cenovnik usluga je ispravno unet.</i>

Slika 4.9 UC-4 Naplaćivanje usluga [Izvor: Marina Damnjanović]

## ZADACI ZA VEŽBU

### Tekst zadataka za vežbu

#### ZADATAK 1.

Pregledajte prethodno opisane slučajeve upotrebe kako biste bili sigurni da su koraci tačni. Proverite da li su razmotrene sve varijacije u normalnom toku i da li su svi izuzeci predviđeni. Dajte kritičko mišljenje o tome li su slučajevi upotrebe opisani na način na koji bi kupci smatrali da je razuman. (5 min)

#### ZADATAK 2.

Napišite samostalno slučajeve korišćenja gde je primarni akter Lekar. (20 min)

#### ZADATAK 3.

Pokušajte da napišete prethodno opisane slučajeve upotrebe kao korisničku priču ili skup korisničkih priča da biste procenili razlike između tradicionalnog i agilnog pristupa u opisivanju korisničkih zahteva. (10 min)

#### ZADATAK 4.

Nacrtajte dijagram slučajeva korišćenja za sistem za upravljanje radom taksi udruženja koji je opisan u prvoj vežbi. (15 min)

**ZADATAK 5.**

Nacrtajte dijagram slučajeva korišćenja za ISUM-ov modul E-student. (15 min)

## ▼ Poglavlje 5

### Domaći zadatak

#### DOMAĆI ZADATAK 6

##### *Tekst domaćeg zadatka*

Nacrtajte dijagram slučajeva korišćenja za sistem koji ste dobili za DZ01. Opišite 1 slučaj korišćenja od navedenih, pomoću šablona datog u predavanju. Obavezno uključite alternativne tokove i izuzetke.

*Napomene:*

Zadatak se rešava opisno i šalje kao .docx fajl. Dijagram pošaljite kao .oom fajl iz PowerDesigner alata.

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

## ✓ Poglavlje 6

### Projektni zadatak

#### ZADATAK ZA RAD NA PROJEKTU

##### *Tekst zadatka za rad na projektu*

Najpre, nacrtajte **dijagram slučajeva korišćenja** za vaš sistem.

Zatim, pomoću šablona datog u uzorku dokumenta Slučajevi korišćenja, kreirajte jedan takav dokument i napravite listu slučajeva korišćenja za vaš trenutni projekat. Opišite sve identifikovane slučajeve korišćenja, takođe prema datom šablonu. Kad god je moguće, uključite alternativne tokove i izuzetke. Polje koje se odnosi na poslovne zahteve za sada ostavite prazno. Oformite **dokument Slučajevi korišćenja**.

## ▼ Poglavlje 7

### Zaključak

## ZAKLJUČAK

1. Slučajevi korišćenja i priče korišćenja služe da se utvrde šta korisnici žele da sistem rad. Korisne su za poslovne aplikacije, razvoj veb sajtova, i interaktivnih sistema.
2. Slučaj korišćenja opisuje sekvencu interakcija između sistema i spoljnih aktera. Priča korisnika (user story) je kratak opis svojstva ispričan iz perspektive osobe koja želi novu sposobnost.
3. Na osnovu specifikacije slučaja korišćenja, analitičar onda dobija funkcionalne zahteve. Priče korisnika se detaljišu i dele na manje, i vode dobijanju testova prihvatanja.
4. Priče korisnika nude prednost u konciznosti i jednostavnosti, a slučajevi korišćenja daju učesnicima strukturu i kontekst, što pričama korisnika nedostaje.
5. Slučajevi korišćenja obezbeđuju apstraktну vizualno predstavljanje zahteva korisnika. Scenario je opis jednog primera upotrebe sistema. Jedan slučaj korišćenja ima kolekciju scenarija.
6. Preduslovi definišu šta mora da bude ispunjeno da bi sistem počeo da izvršava slučaj korišćenja. Postuslovi opisuju stanje sistema posle uspešnog izvršenja slučaja korišćenja.
7. Jedan scenario predstavlja jedan normalan tok događaja za slučaj korišćenja. Može imati i alternativne tokove (scenarije). Izuzeci opisuju uslove za prekid tokova zbog grešaka.
8. Veza extend može povremeno da proširi osnovni slučaj korišćenja. Veza include omogućuje da se jedan slučaj korišćenja koristi u više slučajeva korišćenja.
9. Na radionicama se utvrđuju svi elementi slučajeva korišćenja. Ili se koriste lepljive beleške, ili se projektuje uzorak specifikacije slučajeva korišćenja i kolektivno popunjavaju.
10. Iz dobijenih slučajeva korišćenja, biznis analitičar definiše funkcionalne zahteve. Analitičar takođe generiše modele analize, kao što je dijagram stanja
11. Slučajevi korišćenja otkrivaju važne objekte domena i njihove međusobne odnose. Projektanti OO sistema mogu da mapiraju slučajeve korišćenja u modela objekata i sekvensijalne dijagrame

## REFERENCE

Nastavi 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 materijalom. 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 8** knjige: **Karl Wiegers, Joy Beatty, 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,



SE322 - INŽENJERSTVO ZAHTEVA

## Dokumentovanje zahteva

Lekcija 08

PRIRUČNIK ZA STUDENTE

# SE322 - INŽENJERSTVO ZAHTEVA

## Lekcija 08

### **DOKUMENTOVANJE ZAHTEVA**

- ✓ Dokumentovanje zahteva
- ✓ Poglavlje 1: Predstavljanje i označavanje softverskih zahteva
- ✓ Poglavlje 2: Uzorak specifikacije softverskih zahteva
- ✓ Poglavlje 3: Specifikacija zahteva za agilni razvoj softvera
- ✓ Poglavlje 4: Karakteristike odličnih zahteva
- ✓ Poglavlje 5: Preporuke za pisanje zahteva
- ✓ Poglavlje 6: Vežba
- ✓ Poglavlje 7: Domaći zadatak
- ✓ Poglavlje 8: Projektni zadatak
- ✓ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

## ❖ Uvod

# UVOD

### *Uvodne napomene*

Jasna i efektivna komunikacija je ključni princip razvoja zahteva. To je komunikacija između onih koji imaju potrebe za softverom i onih koji treba da ga razviju, primene i verifikuju. Vešt poslovni analitičar treba da izabere najefektniji način komunikacije informacije o svakom tipu zahteva korisnicima te informacije.

Rezultat razvoja zahteva je dokumentovan sporazum među akterima o proizvodu koji treba da se razvije i primeni. Kao što smo pokazali ranije, dokument o viziji i okviru sadrži poslovne zahteve, zahtevi korisnika se zahvataju primenom slučajeva korišćenja ili korisničkih priča. Funkcionalni i nefunkcionalni zahtevi se najčešće nalaze u specifikaciji softverskog zahteva (SRS - Software Requirement Specification). Ovaj dokument je namenjen onima koji treba da projektuju, konstruišu i verifikuju (potvrde) razvijeno softversko rešenje. Zapisivanje zahteva na jedan organizovan način na koji ključni akteri projekta mogu da izvrše njihovu recenziju, obezbeđuje da oni znaju o čemu su se dogovorili,

U ovoj lekciji izložićemo svrhu, strukturu i sadržaj Dokumenta sa specifikacijom softverskih zahteva (SRS)-Mada se izlaže kao dokument, SRS ne mora da bude u formi dokumenta. U formi dokumenta, SRS ima sledeća ograničenja:

- teško je uneti opise atributa zajedno sa zahtevima,
- upravljanje promenama je nezgrapno,
- teško je zadržati istoriju verzija zahteva,
- nije lako da se izdvoji deo zahteva koji su alocirani u određenoj iteraciji, ili da se zadrži trag sa onim koji su jednom dobine saglasnost, ali su kasnije povučene ili poništene,
- teško je povezati zahteve sa drugim artifikatima projektovanja sistema,
- dupliranje zahteva koji logički odgovaraju i više različitih delova sistema dovode do problema sa održavanjem.

Kao alternativa, SRS možete organizovati u Excel-u, kao Wiki, u bazi podataka, ili u softverskom alatu za upravljanje zahtevima (RM - Requirement Management). Međutim, bez obzira na formu skladištenja informacija u zahtevima sistema, vi imate potrebu za istim informacijama. Uzorak SRS dokumenta koji se ovde izlaže je zbog toga koristan, jer daje strukturu potrebnih informacija, što je podsetnik za njihovo prikupljanje i organizovanje

### UVODNI VIDEO

*Trajanje video snimka: 1min 9sek*

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

## ▼ Poglavlje 1

# Predstavljanje i označavanje softverskih zahteva

## VIDEO PREDAVANJE ZA OBJEKAT "PREDSTAVLJANJE I OZNAČAVANJE SOFTVERSkiH ZAHTEVA"

*Trajanje video snimka: 28min 10sek*

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

## PREDSTAVLJANJE SOFTVERSkiH ZAHTEVA

*Progresivno poboljšanje detalja je ključni princip efektivnog razvoja zahteva.*

Neki misle da ne treba gubiti vreme na dokumentovanju zahteva. Smatraju da kreiranje i održavanje dokumentacije o zahtevima ne daje mnogo vrednosti proizvodu. Međutim, trošak zapisivanja i uređivanja zahteva je mali u odnosu na trošak prikupljanja znanja ili reinženjeringu sistema u nekom trenutku u budućnosti. Tada dokument sa specifikacijom zahteva i njihovo modelovanje može da pomogne učesnicima u projektu da usmere svoja razmišljanja i da precizno postave stanje stvari, što obično same diskusije ostavljaju dvosmislenim. Takođe, u budućim sličnim projektima, preneseno znanje o zahtevima iz prethodnih projekata je vredno nasleđe koje daje veliku prednost firmama koje ga imaju. To smanjuje nove napore, jer ima dosta ponavljanje kod projekata, i sprečava da se neka teško stečena znanja zaborave.

Nikada nećete dobiti idealne zahteve. Kao što znate, pišete zahteve za određene korisnike. Učešće detalja, vrstu informacija koju dajete, i način na koji organizujete informacije - sve je podređeno potrebama korisnika za koje pišete dokumenta i projektnu dokumentaciju, jer je pišete u skladu sa potrebama tih korisnika. Analitičari obično pišu dokumenta iz svoje perspektive, međutim, dobro bi bilo da je pišu iz perspektive tih korisnika, tako da dokumentacija bude ono što njima treba. Zato je dobro da predstavnik korisnika dokumenta o specifikaciji zahteva vrši njegovu recenziju. Progresivno poboljšanje detalja je ključni princip efektivnog razvoja zahteva. Najčešće nije potrebno u ranim fazama projekta ući u detalje svakog zahteva.

Umesto toga, razmišljajte o nivoima zahteva. Na početku, treba da znate zahteve u meri koja je potrebna da napravite prioritete među njima i da ih rasporedite po izdanjima softvera ili po iteracijama (ako primenjujete agilne metode razvoja). Kasnije, kada treba da date

inženjerima razvoja i programerima vaše zahteve, vi ćete ih dopuniti neophodnim detaljima. Time izbegavate ponovni rad na njima kada kasnije imate više saznanja, u odnosu na rane faze projekta.

Ni najbolji dokument o zahtevima ne može da zameni diskusije o zahtevima tokom trajanja projekta. Držite otvorene komunikacione linije između biznis analitičara, razvojnog tima, predstavnika kupaca, i drugih aktera projekta, kako bi brzo razrešili mnogobrojne probleme koji će se javiti tokom rada na projektu.

Možete predstaviti softverske zahteve na nekoliko načina:

- dobro strukturisan i pažljivo napisan **dokument u prirodnom jeziku**
- **primena vizuelnih modela** koji ilustruju procese transformacije, stanja sistema, i promene između njih, bez među podacima, logičke tokove, i sl.
- **formalne specifikacije** koje definišu zahteve upotrebom preciznijih matematičkih jezika za specifikaciju.

Ovde ćemo govoriti o praktičnom načinu dokumentovanja zahteva, primenom kombinacije prva dva načina.

## SPECIFIKACIJA SOFTVERSKIH ZAHTEVA

*Specifikacija softverskih zahteva sadrži funkcije i sposobnosti koje softverski sistem mora da obezbedi, njegove karakteristike, i ograničenja koja mora da poštuje.*

Specifikacija softverskih zahteva (SRS -Software Requirement Specification) sadrži funkcije i sposobnosti koje softverski sistem mora da obezbedi, njegove karakteristike, i ograničenja koja mora da poštuje. On treba da opiše u potrebnoj meri, ponašanje sistema pod određenim uslovima, kao i poželjne kvalitete sistema, kao što su performanse, bezbednost i upotrebljivost. SRS je osnova za izradu drugih dokumenta, kao što je planiranje sistema, projektovanje, i kodiranje. On je takođe temelj za pripremu testiranja softvera, i za izradu korisničke dokumentacije. Međutim, SRS ne bi trebalo da sadrži detalje o projektnom rešenju softvera, njegovoj konstrukciji, testiranju i o upravljanju projektom, sem ograničenja koje projektovanje i implementacija može da zahteva. I kod primene agilnih metoda razvoja softvera, potrebne su informacije koje sadrži jedan dobar SRS dokument. Oni ne prikupljaju na jedan organizovan način informacije kao što se radi sa SRS dokumentom, ali SRS dokument ih podseća na vrstu znanja koje treba da istražuju.

Kako različiti akteri imaju potrebe za različitim informacijama, mi na predmetu predlažemo tri vrste dokumenta:

- Dokument o viziji i okviru projekta,
- Dokument o zahtevima korisnika, i
- Dokument o specifikaciji softverskih zahteva (SRS)

Korisnici SRS dokumenta su:

- kupci, marketing, i prodavci, koji treba da znaju kakav proizvod treba da dobiju, odn. da isporuče,
- Menadžeri projekta zasnivaju na zahtevima svoje procene rokova, rada, i resursa
- Razvojni tim softvera saznaće šta treba da razvije
- Testeri, koji ga koriste da bi razvili testove bazirane na zahtevima, planove testiranja i test procedure.
- Osoblje za održavanje i podršku ga koriste da razumevanje šta koji deo softvera radi.
- Pisci dokumenata baziraju korisnička uputstva na SRS dokumentu
- Zaduženi za obuke upotrebljavaju SRS i korisničku dokumentaciju da bi razvila materijal obuke
- Pravnici da zahtevi zadovoljavaju zakone i propise.
- Proizvođači baziraju svoj rad i pravno su vezana, na SRS

Ako se nešto ne nalazi u SRS, niko to ne treba da očekuje u svojstvima proizvoda.

## OBELEŽAVANJE ZAHTEVA

*Svaki zahtev treba da ima jedinstveni identifikator. On se koristi kod svakog pozivanja na odgovarajući zahtev.*

Važno je da pišete i organizujete dokument sa specifikacijom softverskih zahteva (SRS) tako da ga razumeju svi akteri projekta. Da bi to postigli, ovde se daju sledeći saveti:

- Koristiti odgovarajući uzorak (sadržaj dokumenta) da bi organizovali sve potrebne informacije.
- Obeležite i uredite konsistentno sekcije, podsekcije, i pojedinačne zahteve.
- Koristite vizuelna sredstva (boldiran, italic, podvučena i obojena slova, kao i odgovarajuće tipove slova - fontove) radi dobijanja uređenijeg i jasnijeg teksta.
- Kreirajte sadržaj dokumenta
- Dajte brojeve svim tabelama i slikama u dokumentu, upišite njihove naslove.
- Koristite svojstvo povezivanja referenci, koje nude savremeni sistemi za pripremu elektronskih dokumenata (kao MS Word).
- Koristite linove ka pojedinim delovima SRS dokumenta
- Uključite vizualno predstavljanje informacije uvek kada to možete.

### Obeležavanje zahteva

Svaki zahtev treba da ima jedinstveni identifikator. On se koristi kod svakog pozivanja na odgovarajući zahtev. To olakšava i kolaboraciju unutar tima.

### Broj redosleda

Najjednostavniji način obeležavanja je davanje broja po redosledu pojavljivanja. Na primer, UV-9 (za slučaje korišćenja) ili DR-29 /za funkcionalne zahteve), Prefiks označava tip zahteva. Softverski alati za uređivanje zahteva automatski dodeljuju brojne oznake, koje su neponovljive.

### Hijerarhijsko dodeljivanje brojeva

Ovo se najčešće koristi. Ako je broj nekog funkcionalnog zahteva FR3.2, onda će svi zahtevi koji iz njega proizilaze dobiti istu oznaku (3.2), ali će nadalje dodavati, iza tačke, nove brojeve (npr. 3.2.1, 3.2.2, 3.2.2.1, itd.).

Svi editori teksta omogućavaju automatsku hijerarhijsku numeraciju teksta. Međutim, nastaje problem kod brisanja nekog zahteva. Kada se obriše zahtev, obriše se i njegova oznaka, sve naredne brojčane oznake se smanjuju za 1, a to se kosi sa našim zahtevom da se brojevi zahteva, kada se jedanput definišu, više ne menjaju, da se ne bi poremetile reference na njega u drugim tekstovima i dokumentima.

Problem se ne rešava, ali se lakše radi, kada se vrši kombinacija oznaka i rednih brojeva, pri označavanju zahteva. Na primer, ED-1, Ed-2, itd..

## HIJERARHIJSKI TEKSTUALNI TAGOVI

*Hijerarhijski tekstualni tagovi su strukturisani, ukazuju na značenje i ne menjaju se pri dodavanju, brisanju ili pomeranju drugih zahteva*

### **Hijerarhijski tekstualni tagovi**

Hijerarhijski tekstualni tagovi su strukturisani, ukazuju na značenje , i ne menjaju se pri dodavanju, brisanju ili pomeranju drugih zahteva. Ovaj metod isto zgodan kod obeležavanje poslovnih pravila. Primer:

#### **Print.ConfirmeCompies**

Ova oznaka se odnosi na sledeći zahtev: "Sistem će zahtevati od korisnika da potvrdi bilo koji zahtev da štampa više od 10 kopija. " Tako zvana oznaka ukazuje da se radi o funkciji štampanja, i da je povezan sa brojem kopija koji treba štampati.

Na primer, oznaka:

#### **Product.Cart.Discount.Error.Shipping**

Ovo može da znači da se proizvod (**Product**) kupuje preko veb sajta koji koristi karte za kupovinu (**Cart**), sa određenim popustom (**Discount**), pri čemu sistem daje i oznaku greške (**Error**), ako kupac unese pogrešnu informaciju, a pri kupovini se dodaju troškovi isporuke proizvoda (**Shipping**). Ovakvim oznakama se eliminiše problem sa održavanjem, koje ima hijerarhijsko označavanje , ali su tagovi duži i treba im davati smislena imena. Ove oznake se mogu kombinovati i sa brojevima, na primer:

**Product.Card.01, Product.Crad.02, ....**

### **Rad sa nekompletnim informacijama**

Ponekad vi znate da vam privremeno nedostaje neka informacija. Onda je možete obeležiti sa TBD (**to be determined** - na engleskom), da bi označili ove rupe u znanju. Potrebno je planirati otklanjanje svih ovakvih "rupa" pre nego što dođe do primene skupa zahteva u kojima se pominju.

## ▼ Poglavlje 2

# Uzorak specifikacije softverskih zahateva

## SADRŽAJ SRS DOKUMENTA

*Postoje različiti uzorci SRS dokumenta, jer postoje i različite veličine projekata razvoj softvera. Na vama je da izaberete onaj koji najviše odgovara vašoj organizaciji i projektu.*

Postoje različiti uzorci SRS dokumenta, jer postoje i različite veličine projekata razvoja softvera. Na vama je da izaberete onaj koji najviše odgovara vašoj organizaciji i projektu. Na slici 1 prikazan je uzorak SRS dokumenta koji se pokazao dobrim u mnogim tipovima projekata.

Izbegnite duple informacije u dokumentu, tj. jednu informaciju upišite samo u jedan deo dokumenta. Ako je potrebna i u drugim delovima, dajte samo referencu ka informaciji u dokumentu. Koristite reference i linkove u dokumentu, da bi se lakše nalazile potrebne informacije.

Pri razvoju dokumenta, koristite verzije i vodite računa o njihovim oznakama i ažuriranju dokumenata. Posle internog objavljivanja verzije, svaka izmena u toj verziji mora da se sadrži u sledećoj verziji, tj. posle prve izmene, odmah promenite verziju dokumenta na kojoj radite, i koja će biti sledeća javna verzija dokumenta. Uključite i istoriju verzija u dokument.

Nadalje ćemo ukratko opisati svaku sekciju podsekciju dokumenta

<b>1. Uvod</b>
1.1. Svrha
1.2. Konvencije u dokumentu
1.3. Okvir projekta
1.4. Reference
<b>2. Opis celine</b>
2.1. Perspektiva proizvoda
2.2. Klase korisnika i karakteristike
2.3. Radno okruženje
2.4. Ograničenja projektovanja i primene
2.5. Prepostavke i zavisnosti
<b>3. Svojstva sistema</b>
3.1. Svojstvo sistema 1
3.1.1. Opis
3.1.2. Funkcionalni zahtevi
<b>4. Zahtevi podataka</b>
4.1. Logički model podataka
4.2. Rečnik podataka
4.3. Izveštaji
4.4. Prikupljanje podataka, integritet, zdržavanje i odlaganje
<b>5. Zahtevi spoljnih interfejsa</b>
5.1. Interfejsi korisnika
5.2. Interfejsi softvera
5.3. Interfejsi hardvera
5.4. Interfejsi komunikacija
<b>6. Atributi kvaliteta</b>
6.1. Upotrebljivost
6.2. Performanse
6.3. Sigurnost
6.4. Bezbednost
6.5... drugi
<b>7. Zahtevi internacionalizacije i lokalizacije</b>
<b>8. Ostali zahtevi</b>
Dodatak A: Rečnik pojmova
Dodatak B: Modeli analize

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

Slika 2.1 Sadržaj dokumenta sa specifikacijom softverskih zahteva (SRS)

# 1. UVOD

*Uvod predstavlja pregled koji će pomoći čitaocu da razume kako je SRS organizovan i kako da ga koristi*

## 1.1 UVOD

Uvod predstavlja pregled koji će pomoći čitaocu da razume kako je SRS organizovan i kako da ga koristi.

### 1.1 Svrha

Identifikujte proizvod čiji su zahtevi za softver navedeni u ovom dokumentu, uključujući broj revizije ili izdanja. Opišite različite vrste korisnika kojima je dokument namenjen, kao što su programeri, rukovodioci projekata, marketinško osoblje, korisnici, testeri i pisci dokumentacije.

### 1.2 Konvencije o dokumentima

Opišite sve korištene standarde ili tipografske konvencije, uključujući značenje određenih stilova teksta, isticanja ili notacija. Ako ručno označavate jedinstvene identifikatore zahteva, ovde možete odrediti format za svakoga ko ga treba kasnije dodati.

### 1.3 Obim projekta

Navedite kratak opis softvera koji se specificira i njegovu svrhu. Povežite softver sa korisničkim ili korporativnim ciljevima i sa poslovnim ciljevima i strategijama. Ako su na raspolaganju zasebna vizija i opseg ili sličan dokument, obratite se njemu, a ne da duplirate njegov sadržaj ovde. SRS koji određuje postepeno puštanje proizvoda koji se razvija treba da

sadrži sopstvenu izjavu o obimu kao podskup dugoročne strateške vizije proizvoda. Možete da date rezime visokih nivoa glavnih funkcija koje izdanje sadrži ili značajnih funkcija koje obavlja.

#### 1.4 Reference

Lista svih dokumenata ili drugih izvora na koje se odnosi ovaj SRS. Uključite hiperveze ako su na postojanoj lokaciji. Oni mogu uključivati vodiče za stil korisničkog interfejsa, ugovore, standarde, specifikacije sistemskih zahteva, specifikacije interfejsa ili SRS za srodnji proizvod. Dajte dovoljno informacija kako bi čitalac mogao pristupiti svakoj referenci, uključujući njen naslov, autora, broj verzije, datum i izvor, lokaciju za skladištenje ili URL.

## 2. OPŠTI OPIS

*Opšti opis predstavlja pregled proizvoda i okruženja na visokom nivou apstrakcije, očekivanih korisnika i poznata ograničenja, prepostavke i zavisnosti.*

### 2. OPŠTI OPIS

Ovaj odeljak predstavlja pregled proizvoda i okruženja na visokom nivou apstrakcije, očekivanih korisnika i poznata ograničenja, prepostavke i zavisnosti.

#### 2.1 Perspektiva proizvoda

Opišite kontekst i poreklo proizvoda. Da li je to sledeći član rastuće linije proizvoda, sledeća verzija zrelog sistema, zamena za postojeću aplikaciju ili potpuno novi proizvod? Ako ovaj SRS definiše komponentu većeg sistema, navedite kako se ovaj softver odnosi na celokupni sistem i identifikujte glavne interfejse između ova dva. Razmotrite uključivanje vizuelnih modela poput kontekstnog dijagrama ili mape ekosistema da biste pokazali odnos proizvoda prema drugim sistemima

#### 2.2 Klase i karakteristike korisnika

Identifikujte različite klase korisnika za koje očekujete da će koristiti ovaj proizvod i opišite njihove relevantne karakteristike. Neki zahtevi mogu se odnositi samo na određene klase korisnika. Identificirajte omiljene klase korisnika. Korisničke klase predstavljaju podskup zainteresovanih strana opisan u dokumentu o viziji i opsegu. Opisi korisničkih klasa su izvor koji se može ponovo koristiti. Ako su dostupni, možete uključiti opise korisničkih klasa tako što ćete ih jednostavno pokazati u katalogu matične klase umesto dupliranja podataka ovde.

#### 2.3 Operativno okruženje

Opišite okruženje u kojem će softver raditi, uključujući hardversku platformu; operativni sistemi i verzije; geografske lokacije korisnika, servera i baza podataka; i organizacije koje nude odgovarajuće baze podataka, servere i veb lokacije. Navedi sve ostale softverske komponente ili aplikacije sa kojima sistem mora mirno koegzistirati. Ako je potrebno razviti opsežne radove na tehničkoj infrastrukturi zajedno sa razvojem novog sistema, razmislite o stvaranju zasebne specifikacije za infrastrukturu da biste je detaljno opisali.

## 2. OPŠTI OPIS (NASTAVAK)

*Nastavak uputstva za pripremu poglavља Opšti opis*

## 2.4 Ograničenja u projektovanju i primeni

Opišite sve faktore koji će ograničiti mogućnosti dostupne programerima. Oni mogu uključivati: korporativne ili regulatorne politike; ograničenja hardvera (vreme i memorijski zahtevi); interfejs do drugih aplikacija; specifične tehnologije, alate i baze podataka koje se koriste; zahtevi ili ograničenja programskog jezika.

## 2.5 Pretpostavke i zavisnosti

Navedi sve prepostavljene faktore (za razliku od poznatih činjenica) koji bi mogli uticati na zahteve koji su navedeni u SRS. Oni mogu da uključuju treće ili komercijalne komponente koje planirate da koristite, ponovo koristite očekivanja, probleme oko razvojnog ili operativnog okruženja ili ograničenja. Na projekat bi moglo uticati ako su te prepostavke netačne, ne dele se ili se menjaju. Takođe identifikujte sve zavisnosti koje projekat ima od spoljnih faktora van njegove kontrole

# 3. FUNKCIJE SISTEMA

*Ovaj obrazac ilustruje organizovanje funkcionalnih zahteva za proizvod prema karakteristikama sistema, glavnim uslugama koje proizvod pruža.*

## 3. FUNKCIJE SISTEMA

Ovaj obrazac ilustruje organizovanje funkcionalnih zahteva za proizvod prema karakteristikama sistema, glavnim uslugama koje proizvod pruža. Ovaj odjeljak možete radije organizovati prema slučaju, načinu rada, korisničkoj klasi, klasi predmeta, funkcionalnoj hijerarhiji, stimulaciji, odgovoru ili kombinacijama istih, bez obzira što ima najlogičniji smisao za vaš proizvod.

### 3.1 Funkcija sistema X

Nemojte zaista reći „Funkcija sistema X.“ Navedite ime funkcije u samo nekoliko reči.

#### 3.1.1 Opis

Navedite kratak opis funkcije i navedite da li je ona visokog, srednjeg ili niskog prioriteta.

#### 3.1.2 Sekvence stimulusa / odgovora

Lista sekvenca korisničkih radnji i sistemskih odgovora koji podstiču ponašanje definisano za ovu značajku. Oni će odgovarati elementima dijaloga koji su povezani sa slučajevima upotrebe.

#### 3.1.3 Funkcionalni zahtevi

Podesite specifične funkcionalne zahteve povezane sa ovoim svojstvom. Ovo su softverske mogućnosti koje moraju biti implementirane da korisnik izvrši usluge ove funkcije ili da izvrši slučaj upotrebe. Opišite kako proizvod treba da reaguje na očekivane uslove greške. Koristite „TBD“ kao rezervirano mesto da naznačite kada potrebne informacije još uvek nisu dostupne

### 3.2 Funkcija sistema 2 (i tako dalje)

.....

## 4. ZAHTEVI ZA PODATKE

*Ovaj odeljak opisuje različite aspekte podataka koje će sistem koristiti kao ulaze, obrađivati na neki način ili kreirati kao izlaze.*

**4. ZAHTEVI ZA PODATKE** Ovaj odeljak opisuje različite aspekte podataka koje će sistem koristiti kao ulaze, obrađivati na neki način ili kreirati kao izlaze.

### 4.1 Logički model podataka

Model podataka je vizuelni prikaz objekata podataka i zbirki koje će sistem obraditi i odnosa između njih. Uključite model podataka za poslovne operacije kojima se bavi sistem ili logičku reprezentaciju za podatke kojima će sam sistem manipulisati. Modeli podataka najčešće se kreiraju kao dijagram odnosa entiteta.

### 4.2 Rečnik podataka

Rečnik podataka definiše sastav podaktovnih struktura i značenje, vrstu podataka, dužinu, format i dopuštene vrednosti za elemente podataka koji čine te strukture. U mnogim slučajevima je bolje da skladištite rečnik podataka kao poseban artefakt, umesto da ga ugrađujete u sredinu SRS-a. To takođe povećava njegov potencijal ponovne upotrebe u drugim projektima.

### 4.3 Izveštaji

Ako će vaša aplikacija generisati bilo koji izveštaj, identifikujte ih ovde i opišite njihove karakteristike. Ako izveštaj mora biti u skladu s određenim unapred definisanim izgledom, ovde možete to specificirati kao ograničenje, možda primerom. U suprotnom, usredsreditе se na logičke opise sadržaja izveštaja, redosled sortiranja, ukupnog nivoa i tako dalje, odlažući detaljan izgled izveštaja u fazi projektovanja.

### 4.4 Prikupljanje podataka, integritet, zadržavanje i odlaganje

Ako je relevantno, opišite kako se podaci prikupljaju i održavaju. Navedite sve zahteve koji se odnose na potrebu zaštite integriteta podataka sistema. Identifikujte bilo koje posebne tehnike koje su neophodne, kao što su rezervne kopije, kontrolne tačke, dupliranje (mirroring) ili verifikacija tačnosti podataka. Državne politike koje sistem mora primenjivati ili za čuvanje ili za uklanjanje podataka, uključujući privremene podatke, metapodatke, preostale podatke (kao što su izbrisani zapisi), keširane podatke, lokalne kopije, arhive i privremene sigurnosne kopije.

## 5. ZAHTEVI ZA SPOLJNE INTERFEJSE

*Ovaj odeljak sadrži informacije kojima se osigurava da će sistem pravilno komunicirati sa korisnicima i sa spoljnim hardverskim ili softverskim elementima.*

### 5. ZAHTEVI ZA SPOLJNE INTERFEJSE

Ovaj odeljak sadrži informacije kojima se osigurava da će sistem pravilno komunicirati sa korisnicima i sa spoljnim hardverskim ili softverskim elementima.

#### 5.1 Korisnički interfejsi

Opišite logičke karakteristike svakog interfejsa između softverskog proizvoda i korisnika. Ovo

može da uključuje uzorce slika na ekranu, bilo koje GUI standarde ili vodiče za porodični stil proizvoda koji se moraju pridržavati, ograničenja izgleda ekrana, standardne tastere i funkcije (npr. Pomoć) koje će se pojavljivati na svakom ekranu, prečice na tastaturi, standardi prikazivanja greške i uskoro. Definišite softverske komponente za koje je potreban korisnički interfejs. Pojedinosti o dizajnu korisničkog interfejsa treba da budu dokumentovane u posebnoj specifikaciji korisničkog interfejsa.

### 5.2 Softverski interfejsi

Opišite veze između ovog proizvoda i drugih softverskih komponenti (identifikovanih imenom i verzijom), uključujući ostale aplikacije, baze podataka, operativne sisteme, alate, biblioteke, veb lokacije i integrisane komercijalne komponente. Navedite svrhu, formate i sadržaj poruka, podataka i kontrolnih vrednosti koje se razmenjuju između softverskih komponenti. Navedite preslikavanja ulaznih i izlaznih podataka između sistema i sve prevode koji su potrebni da bi podaci prešli iz jednog sistema u drugi. Opišite usluge potrebne od ili od spoljnih softverskih komponenti i prirodu komunikacije interkomponenata. Identifikujte podatke koji će se razmenjivati između ili deliti između komponenti softvera. Navedite nefunkcionalne zahteve koji utiču na interfejs, kao što su nivoi usluga za vreme i frekvencije odgovora ili sigurnosne kontrole i ograničenja.

## 5. ZAHTEVI ZA SPOLJNE INTERFEJSE (NASTAVAK)

*Ovo je nastavak uputstva za popunjavanje odeljka 5. Zahtevi za spoljne interfejse*

### 5.3 Hardverski interfejsi

Opišite karakteristike svakog interfejsa između softverske i hardverske (ako postoje) komponente sistema. Ovaj opis može uključivati podržane tipove uređaja, podatke i kontrolne interakcije softvera i hardvera i komunikacijske protokole koji će se koristiti. Navedite ulaze i izlaze, njihove formate, njihove važeće vrednosti ili raspone i sve probleme sa vremenom koji programeri moraju biti svesni. Ako su ove informacije opsežne, razmislite o kreiranju posebnog dokumenta specifikacije interfejsa.

### 5.4 Komunikacijski interfejsi

Navedite zahteve za sve komunikacione funkcije koje će proizvod koristiti, uključujući e-poštu, veb pregledač, mrežne protokole i elektronske obrasce. Definišite bilo koje relevantno formatiranje poruke. Navedite probleme sigurnosti ili šifrovanja komunikacije, brzine prenosa podataka, rukovanje rukama i mehanizme sinhronizacije. Navedite bilo kakva ograničenja oko ovih interfejsa, kao što su da li su prilozi e-pošte prihvativi ili ne.

## 6. ATRIBUTI KVALITETA

*Ovde se definišu nefunkcionalni zahtevi*

### 6. ATRIBUTI KVALITETA

**6.1 Upotrebljivost** Navedite sve zahteve u vezi sa karakteristikama zbog kojih će softver izgledati kao "user-friendli". Upotrebljivost obuhvata jednostavnost korišćenja, jednostavnost

učenja; pamćenje; izbegavanje grešaka, rukovanje i oporavak; efikasnost interakcija; pristupačnost; i ergonomija. Ponekad se mogu sukobiti jedno sa drugim, kao i sa lakoćom korišćenja u odnosu na lakoću učenja. Navedite sve standarde ili smernice za dizajn korisničkog interfejsa kojima se aplikacija mora uskladiti.

### **6.2 Performanse**

Zahlevi performansi za različite operacije sistema. Ako različiti funkcionalni zahtevi ili karakteristike imaju različite zahteve za performansama, primereno je da se ciljevi performansi tačno odrede odgovarajućim funkcionalnim zahtevima, a ne da se prikupljaju u ovom odeljku.

### **6.3 Bezbednost**

Navedite sve zahteve u vezi sa pitanjima bezbednosti ili privatnosti koji ograničavaju pristup ili upotrebu proizvoda. Oni se mogu odnositi na fizičku, bezbednost podataka ili softver. Sigurnosni zahtevi često potiču iz poslovnih pravila, pa identifikujte sve sigurnosne ili privatne politike ili propise kojima se proizvod mora pridržavati. Ako su oni dokumentovani u skladištu poslovnih pravila, samo ih pogledajte.

### **6.4 Bezbednost**

Navedite zahteve koji se odnose na mogući gubitak, oštećenje ili štetu koja može proizaći iz upotrebe proizvoda. Definišite sve zaštitne mere ili radnje koje se moraju preduzeti, kao i potencijalno opasne radnje koje se moraju sprečiti. Identifikujte sve sigurnosne sertifikate, politike ili propise kojima se proizvod mora uskladiti.

### **6.5 [Ostali po potrebi]**

Kreirajte poseban odeljak u SRS-u za svaki dodatni atribut kvaliteta proizvoda da biste opisali karakteristike koje će biti važne bilo kupcima ili programerima. Mogućnosti uključuju dostupnost, efikasnost, instalabilnost, integritet, interoperabilnost, izmenljivost, prenosivost, pouzdanost, upotrebljivost, robusnost, skalabilnost i proverljivost. Napišite ove da budu specifične, kvantitativne i proverljive. Razjasnite relativne prioritete za različite atribute, kao što je sigurnost nad performansama.

## **7. USLOVI INTERNACIONALIZACIJE I LOKALIZACIJE**

*Odeljak daje uputstvo za primenu internacionalizaciju i lokalizaciju*

### **7. USLOVI INTERNACIONALIZACIJE I LOKALIZACIJE**

Zahlevi internacionalizacije i lokalizacije osiguravaju da će proizvod biti pogodan za upotrebu u nacijama, kulturama i geografskim lokacijama koje nisu one u kojima je stvoren. Takvi zahtevi mogu da reše razlike u: valuti; formatiranje datuma, brojeva, adresa i telefonskih brojeva; jezik, uključujući nacionalne pravopisne konvencije na istom jeziku (poput američkog naspram britanskog engleskog), korišćene simbole i skupove znakova; određeno ime i prezime; vremenske zone; međunarodni propisi i zakoni; kulturna i politička pitanja; korišćene veličine papira; utezi i mere; električni naponi i oblici utikača; i mnogi drugi

## 8. OSTALI USLOVI

*Ovaj odeljak sadrži informacije o zakonskim, regulatornim ili finansijskim uslovima za rad softvera, kao i informacije o primjenjenim industrijskim stan*

### 8. OSTALI USLOVI

Primeri su: zakonski, regulatorni ili finansijski uslovi i zahtevi za standardi; zahtevi za instalaciju proizvoda, konfiguraciju, pokretanje i gašenje; i zahteve za evidentiranje, nadgledanje i reviziju. Umesto da sve to kombinujete pod „Ostalo“, dodajte sve nove odeljke u predložak koji se odnose na vaš projekat. Propustite ovaj odeljak ako su svi vaši zahtevi smešteni u drugim delovima.

#### DODACI

##### Dodatak A: Rečnik

Definišite sve specijalizovane pojmove koje čitalac mora da zna da bi shvatio SRS, uključujući skraćenice i skraćenice. Prepišite svaki akronim i navedite njegovu definiciju. Razmislite o stvaranju višestrukog upotrebnog pojma na nivou preduzeća koji obuhvata više projekata i koji sadrži reference po bilo kojim uslovima koji se odnose na ovaj projekat.

##### Dodatak B: Modeli analize

Ovaj opcioni odeljak uključuje ili ukazuje na relevantne modele analize kao što su dijagrami protoka podataka, stabla funkcija, dijagrami stanja i dijagrami odnosa entiteta. Možda biste radile da umetnete određene modele u relevantne odeljke specifikacije umesto da ih prikupljate na kraju.

## ▼ Poglavlje 3

# Specifikacija zahteva za agilni razvoj softvera

## VIDEO PREDAVANJE ZA OBJEKAT "SPECIFIKACIJA ZAHTEVA ZA AGILNI RAZVOJ SOFTVERA"

*Trajanje video snimka: 40min 3sek*

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

## SPECIFIKACIJA ZAHTEVA PRI AGILNOM RAZVOJU SOFTEVERA

*Uместо funkcionalnih zahteva, najčešće se razvoj upravlja prema postavljenim testovima prihvatanja nastalim iz korisničkih priča.*

Pri agilnim razvoju softvera najviše se koriste priče korisnika radi utvrđivanja zahteva. Svaka priča je jedan iskaz o potrebama korisnik ili o funkciji koju celi sistem treba da ima. Kada se prikupi dovoljan skup priča, pravi se njihov prioritet i raspoređivanja po iteracijama. U slučaju većih korisničkih priča koje se ne mogu realizovati u jednoj iteraciji, dele se na više manjih priča i raspoređuju po iteracijama, zapisivanje priča korisnika se mogu koristiti kartice ili softverski alati za rad sa pričama korisnika.

Kod svake iteracije akteri projekta diskutuju i definišu nove detalje priča korisnika. Znači, zahtevi se ažuriraju i detaljniju u vreme kada se primenjuju. Testiranje se vrši i u budućim iteracijama, radi provere.

U tom slučaju umesto funkcionalnih zahteva, najčešće se razvoj upravlja prema postavljenim testovima prihvatanja nastalim iz korisničkih priča. Testovi se rade u istoj iteraciji u kojoj se implementira priča korisnika. Treba da pokriju i očekivano ponašanje i izuzetke, tj. reakciju sistema u izuzetnim slučajevima, pokrenuti nekim događajima. Ako tim odstupi od početnih priča korisnika, onda od dokumentacije ostaju samo testovi prihvatanja.

Testovi se mogu pisati na kartama, ali se mogu unositi i preko alata za testiranje. Testovi se automatski izvode radi primene regresionog testiranja.

Nefunkcionalni zahtevi se ne unose kroz priče korisnika, već kao ograničenja. Alternativno, mogu se zapisati kao nefunkcionalni zahtevi koji se povezuju sa odgovarajućom pričom korisnika u vidu testa prihvatanja i kriterijuma za ocenjivanje testa. Razvojni tim može da

koristi i modele analize za predstavljanje znanja o radu sistema. Samo od razvojnog tima zavisi koji će metod izabrati za specifikaciju zahteva. Važno je da je izabran metod dovoljno dobar za konstrukciju sledećeg dela softvera pri čemu se radi sa prihvatljivom dozom rizika. Formalni nivo i nivo detaljisanja koji će dokument o zahtevima imati zavisi od sledećeg:

- od proširenja neformalne, just-in-time verbalne i vizuelne komunikacije između kupaca i razvojnog tima, koji treba da omogući dovoljno informacija za razvoj proizvoda u novoj iteraciji.
- od proširenje sinhronizovane neformalne komunikacije među članovima tima
- zavisno od potrebe da se za buduće projekte zadrži stečeno znanje, radi održavanja, reinženjeringu aplikacija, verifikaciju, ili radi ugovornih obaveza i provera ispunjenja ugovora.
- od mere kojom testovi prihvatanja mogu da budu efektivna zamena opisa očekivanog ponašanja sistema
- od stepena do kog ljudska memorija može da zameni pisano predstavljanje.

## ▼ Poglavlje 4

### Karakteristike odličnih zahteva

#### KARAKTERISTIKE ISKAZA O ZAHTEVIMA

*Kompletност, tačnost, ostvarljivost, potreba, usklađenost sa prioritetima, nedvosmislenost, proverljivost - to su karakteristike dobro definisanih zahteva.*

**Kompletnost** Svaki zahtev treba da sadrži dovoljno informacija potrebnih da ih korisnik razume. U slučaju funkcionalnih zahteva, to znači da informacija mora da zadovolji potrebe programera kako bi pravilo primenio zahtev. Nedostajuće informacije se moraju uneti pre početka konstrukcije softvera.

**Tačnost** Svake mogućnosti kroz koje zahtev može da opiše sposobnosti sistema da tačno zadovolje neke potrebe aktera uz jasan opis funkcionalnosti koja se mora ostvariti. Tačnost proverava onaj koji je i zahtevao određeni zahtev. To može biti i slučaj korišćenja, poslovno pravilo, inicijalni dokument, lista zahteva sa visokim nivoom apstrakcije ili neki drugi dokument. Predstavnici korisnika bi trebalo da recenziraju zahteve.

**Ostvarljivost** Zahtev treba da bude ostvarljiv sa poznatim sposobnostima i ograničenjima sistema i njegovog radnog okruženja, kao i u okviru ograničenja projekta: rokovi, budžet i ljudi. Inženjer razvoja koji učestvuje u utvrđivanju zahteva treba da vrši proveru regularnosti zahteva. Ostvarljivost zahteva se vrši na dva načina: primenom prototipova za proveru koncepta i primenom inkrementalnog razvoja. Ako se zahtev odbaci kao neostvarljiv, onda treba proveriti posledicu na viziju i okvir projekta.

**Potreba** Svaki zahtev treba da opisuje sposobnost sistema koje donosi poslovnu vrednost akterima projekta, da se proizvod razlikuje od drugih na tržištu, ili ostvaruje usaglašenost sa standardima i drugom regulativom. Svaki zahtev treba da potiče iz izvora koji je ovlašćen da postavlja i definiše zahteve. Povežite sve funkcionalne i nefunkcionalne zahteve sa njihovim izvorima, da bi ovo proverili. Mora se povezati svaki zahtev sa poslovnim ciljevima kako bi se pokazalo da je on potreban.

**Usklađenost sa prioritetima** Dajte prioritet zahtevima koji su najvažniji za ostvarenje željene vrednosti. Dodelite prioritet svakom funkcionalnom zahtevu, zahtevu korisnika, scenariju slučaja korišćenja, ili svojstvu, kako bi se pokazalo koliko je bitan za određeno izdanje softvera. Određivanje prioriteta se kolektivno radi.

**Nedvosmislenost** Svi moraju da razumeju značenje zahteva na isti način, to se proverava inspekcijom.

**Proverljivost** Zahtevi moraju biti proverljivi i to se utvrđuje testiranjem. Zato treba testere uključiti u proces utvrđivanja zahteva.

## KARAKTERISTIKA KOLEKCIJA ZAHTEVA

*Kompletност, konzistentност, promenljivost i sledljivost su karakteristike skupa zahteva izdanja.*

Nije dovoljno imati odlične iskaze o zahtevima. Važno je i da skup svih zahteva koji su planirani za određeno izdanje softvera ili iteraciju ima dobre karakteristike, koje se ovde navode:

**Kompletnost (Complete)** Ne bi trebalo da nedostaje nijedan zahtev ili važna informacija. Njih je teško otkriti, jer se ne pominju, a često se implicitno podrazumevaju. Treba otkriti da neki zahtev nije eksplisitno naveden, i treba ga onda dodati.

**Konzistentnost (Consistent)** Konsistentni zahtevi su oni koji nisu u sukobu sa drugima zahtevima istog tipa, ili sa zahtevima poslovanja u širem smislu, zahtevima korisnika, ili sa sistemskim zahtevima. Ako ne otklonite takve konfliktne situacije, programeri će kasnije te probleme morati da rešavaju. Zato je važno da znate izvor svakog zahteva da bi sa izvorom otklonili problem.

**Promenljivost (Modifiable)** Neophodno je da zapisujete i održavate istoriju promena svakog zahteva. Morate znati veze i zavisnosti među zahtevima da bi mogli da ih sve promenite, kada menjate jedan od njih. Promenljivost znači da je svaki zahtev jedinstveno obeležen i nezavisno iskazan, tako da možete nedvosmisleno tražiti i naći. Zato, izbegnite redundantnost podataka, tj. da se isti zahtev navodi na različitim mestima. Na drugim mestima se može koristiti samo referenca na zahtev koji se definiše samo na jednom mestu.

**Sledljivost (Traceable)** Sledljiv zahtev može da se poveže sa svojim izvorima, ali i sa zahtevima koji su iz njega izvedeni, sa elementima projektnog rešenja, sa kodom koji ga primenjuje, i sa testom koji potvrđuje njegovu primenu. Ne morate da definišete sve te veze za praćenje da bi omogućili da zahtev bude sledljiv. Sledljiv zahtev se jedinstveno obeležava sa trajno memorisanim identifikatorom. Oni se pišu na strukturisan način, sa potrebnim nivoom detaljnosti, a ne u vidu dugačkih paragrafa. Izbegnite navođenje više zahteva zajedno u okviru jednog iskaza, jer pojedini iskazi mogu da vide do različitih komponenti razvoja, a njihovo grupisanje to sprečava (jer su u "istom košu").

U praksi, nikada nećete imati slučaj da svi zahtevi imaju sve ove karakteristike. Ako o njima vodite računa, imaćete bolje definisane zahteve, te samim tim, i bolji softver.

## ▼ Poglavlje 5

# Preporuke za pisanje zahteva

## PISANJE ZAHTEVA IZ PERSPEKTIVE SISTEMA ILI KORISNIKA

*Pri pisanju zahteva, možete ga pisati tako da kažete šta sistem treba da uradi (perspektiva sistema) ili šta korisnik treba da uradi (perspektiva korisnika).*

Ne postoji formalan način pisanja odličnih zahteva. Do toga se dolazi iskustvom i praćenjem reakcija korisnika vaših specifikacija zahteva. Zato su recenzije bitne. Razmenjujte specifikacije zahteva sa vašim kolegama, analitičarima. Učićete jedni od drugih. Poboljšaćete vašu kolektivnu sposobnost pisanja dobrih zahteva i otkrićete greške i mogućnosti za njihovo otklanjanje u ranim fazama projekta. Pri pisanju zahteva, imate dva cilja:

- Svako ko čita zahtev, treba da ga razume ga na isti način,
- Razumevanje svakog čitaoca nekog zahteva je isto sa autorovim razumevanjem i namerom u vezi zahteva.

### **Perspektiva sistema ili korisnika**

Pri pisanju zahteva, možete ga pisati tako da kažete šta sistem treba da uradi (perspektiva sistema) ili šta korisnik treba da uradi (perspektiva korisnika). Izaberite onu koja najjasnije definiše određeni zahtev. Pišite zahteve na konsistentan način: "Sistem treba..." ili "Korisnik treba...". Specificirajte akciju pokretača i uslove za to. Ovde se daje opšti uzorak za pisanje zahteva:

**[ opcioni uslov] [opcioni događaj pokretanja] [očekivan odgovor sistema]  
sistem treba [odgovor sistema ]**

Evo primera jednog funkcionalnog zahteva:

*Ako je zahtevana hemikalija pronađena u skladištu hemikalija, sistem treba da prikaže listu svih kontejnera sa hemikalijom koji se trenutno nalaze u skladištu.*

Ovaj zahtev sadrži preduslov, ali nemaju pokretač akcije (događaj). Da bi se izbeglo ponavljanje, može se i izostaviti fraza, "sistem treba...". Trebalo bi da navedete akciju opkretača ili uslova koji dovodi do reakcije sistema na opisani način.

Ponekad, jasnije je ako se zahtev napiše iz perspektive korisnika, uz korišćenje aktivnih glagola, jer čini traženu akciju jasnijom.

**[klasa korisnika ili ime aktera ] treba da [uradi nešto ] [nekom objektu ] [uslovi prihvatanja, vreme odgovora, ili iskaz kvaliteta ]**

Evo primera:

*Hemičar treba da ponovo zahteva hemikalije koje ja zahtevao u prošlosti pregledajući menjajući detalje u ranijim zahtevima.*

Umesto "korisnik", ovde se navodi konkretna klasa korisnika "Hemičar". To je dobro, jer treba da zahtev uvek kada je moguće, bude što konkretniji.

## STIL PISANJA

*Prvo navedite iskaz o potrebi ili o funkcionalnosti, a onda detalje podrške, tj. razloge, izvor, prioritet, i druge atributie zahteva. Važno je pisati jednostavno, jasno i koncizno.*

Prvo navedite iskaz o potrebi ili o funkcionalnosti, a onda detalje podrške \*razloge, izvor, prioritet, i druge atribute zahteva.

Ne mešajte pasivno i aktivno pisanje. Konsistentno primenjujte jedno od njih. Ne koristite više izraza za isti koncept. Važno je pisati jednostavno i razumljivo. Ovde se daju preporuke za pisanje jasnih zahteva:

**Jasno i koncizno:** Pišite zahteve sa kompletном rečenicom, sa odgovarajućom gramatikom. Neka rečenice budu kratke i direktne. Izbegnite žargone i pišite jezikom domena. Dajte objašnjenje termina u rečniku termina. Pored jasnoće, važna je (mada manje) i konciznost. Uvek se pitajte šta će korisnik raditi sa informacijom koju mu dajete. Ako to nije jasno, onda je izbacite. Budite precizni i konkretni.

**Ključna reč "treba":** Ona označava sposobnost sistema koju želite. Ne mešajte termine u zahtevima. Ponavljajte istu ključnu reč.

**Aktivni govor:** Pišite aktivnim rečenicama tako da bude jasno šta entitet treba da uradi. Umesto da pišete šta se nekom treba da uradi (pasivni način), pišite ko i šta treba da uradi (aktivni način pisanja). Evo primera:

Kada odeljenje isporuke potvrdi da je pošiljka sa poboljšanim proizvodom послата, sistem će promeniti serijski broj proizvoda u ugovoru sa novim serijskim brojem.

**Pojedinačni zahtevi:** Umesto jednog složenog sa više zahteva u sebi, pišite jednostavne, posebne i pojedinačne zahteve. U takvim slučajevima, napišite više pojedinačnih zahteva.

## NIVO DETALJA

*Zahteve treba pisati na onom nivou detalja koliko je minimalno potrebno za njihovu primenu*

Zahteve treba pisati na onom nivou detalja koliko je minimalno potrebno inženjerima razvoja da na osnovu toga, ih pravilno primene.

**Odgovarajući detalji:** Uopštene zahteve treba razbiti na više jednostavnih na potrebnom nivou detaljnosti informacija. Koliko detaljno? Onoliko koliko je minimalno potrebno da

minimizirate rizik od različitih razumevanja zahteva. Treba da uključite više detalja u sledećim slučajevima:

- Radi se posao za odličnog kupca.
- Razvoj ili testiranje biće dato u outsourcing, tj, dato kooperantu.
- Članovi projektnog tima su na različitim lokacijama.
- Testiranje sistema biće zasnovano na zahtevima.
- Potrebne su tačne procene.
- Potrebno je trasirati zahteve, tj. imati veze ka njihovim izvorima i ka izvedenim zahtevima.

#### **Može se dati manje detalja kada:**

- Kada radite interni projekt za vašu kompaniju.
- Kupci su vrlo uključeni u projekat.
- Programeri i inženjeri imaju puno iskustva.
- Dozvoljeni su presedani, kao u slučaju zamene prethodne aplikacije.
- Koristi se paketno rešenje (softver)

**Konsistentna granularnost:** Pisci zahteva se stalno pitaju sa kojom granularnošću da pišu funkcionalne zahteve? Ne morate sve zahteve pisati na istom nivou detalja. U delovima koji nose veće rizike možete više u dubinu (veći novo detalja). Ipak, zgodno je ceo skup funkcionalnih zahteva pisati sa konzistentnim nivoom granularnosti. Postoji preporuka da granularnost bude takva da zahtev može da se verifikuje jednim testom.

## TEHNIKE PRESTAVLJANJA ZAHTEVA

*Izaberite najefektniji način predstavljanja zahteva. Izbegavajte dvosmislenosti.*

#### **Predstavljanje zahteva:**

Koristite najefektniji oblik predstavljanja svakog zahteva u skladu sa odgovarajućim auditorijumom. Mogu se koristiti liste, tabele, vizuelni modeli analize, dijagrami, matematičke formule, zvučni i video zapisi. Sve ovo ne može biti zamena i za primenu tekstualnog opisa zahteva. ali služi kao odličan dodatak koji povećava čitaocu razumljivost zahteva.

Kada imamo grupu sličnih zahteva, koji se razlikuju po nekoj činjenici ili podatku, onda je najbolje koristiti tabelarni prikaz takvih zahteva. Ako neka kombinacija ne dovodi do nekog funkcionalnog zahteva, onda je bolje je zadržati u tabeli, ali označiti je "nije primenljivo", umesto da je izostavite. To bi moglo da izazove nedoumicu kod čitaoca, i da se pitaju da li je nekom greškom ta kombinacija izostavljena.

#### **Izbegavanje dvosmislenosti**

Recenzija zahteva može najbolje da ukaže na nejasne ili na dvosmislene zahteve. Ovde se navode nekoliko izvora dvosmislenosti u zahtevima.

**Nejasne reči:** Koristite konzistentno pojmove i definišite ih u vašem rečniku projekta. Ne mešajte sinonime. Usvojite samo jedan naziv nekog pojma (koncepta) za pisanje zahteva. Izbegnite korišćenje dvosmislenih ili nepreciznih termina i dvosmisleni jezik.

**Oblik A/B:** Mnogi zahtevi sadrže razlomke tipa A/B u kojima se nalaze sinonimi illi neki drugi termini. Često su oni dvosmisleni. Autor možda nije bio siguran koji od dva termina da koristi, pa je na ovaj način naveo oba. Čitalac može svaki od njih da razume različito, i to onda dovodi do dvosmislenosti,

**Granične vrednosti:** Ako granične vrednosti nisu precizno i eksplisitno navedene, onda one unose dvosmislenost u zahteve i prave nejasnoće. Na primer, "Student dobija 8, ako ima 80 do 90 poena, a 10, ako ima od 90 do 100": Nije jasno koju ocenu student dobija ako ima 90 poena.

**Negativni zahtevi:** Zahtev govori šta sistem ne treba da radi. To može da unese nejasnoće. Napišite takve zahteve u pozitivnom smislu, koji bi jasno ukazao na željeno ponašanje sistema,

## IZBEGAVANJE NEKOMPLETNOSTI

*Postoje nekoliko tehnika da pronađete nedostajuće zahteve:  
Simetrija, složena logika i nedostajući izuzeci.*

Postoje nekoliko tehnika da pronađete nedostajuće zahteve. Usmerite se više na zadatke korisnika umesto na svojstva sistem da bi pronašli nezahtevanu, a potrebnu funkcionalnost.

**Simetrija:** Simetrične operacije su česti izvor nedostajućih zahteva. Na primer: "Korisnik mora da memoriše ugovor u bilo kom trenutku tokom manuelnog postavljanja ugovora". Ne govori se o tome da korisnik mora da ima mogućnost da izvadi nekompletan zahtev koji je memorisan, da bi nastavio rad na njemu. Programer mora to da zna.

**Složena logika:** Obično zahtevi u kojima se koriste složeni logički iskazi ostavljaju neke kombinacije bez odgovora, tj. nisu kompletne. Programer kada analizira takve zahteve dođe u nedoumicu kako da ih tumači.

**Nedostajući izuzeci:** Svaki zahtev ima deo koji govori šta sistem treba da radi, ali bi trebalo da ima i deo šta treba da radi u slučaju izuzetnih situacija (tzv. izuzeci). Ako oni nedostaju u opisu zahteva, može u praksi da dođe do nepredviđene situacije, i sistem se blokira. Programer to može da uoči, ali mora da stvar raspravi sa autorom zahteva ili sa analitičarem.

## VIDEO 24 - CHARACTERISTICS OF EXCELLENT REQUIREMENTS - WIEGERS (VIDEO)

*Trajanje: 5:14 minuta*

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

## VIDEO 25 - TIPS FOR WRITING CLEAR REQUIREMENTS - WIEGERS (VIDEO)

*Trajanje: 4:02 minuta*

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

## VIDEO 26 - TIPS FOR WRITING CLEAR REQUIREMENTS - 2 - WIEGERS (VIDEO)

*Trajanje: 5 minuta*

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

## VIDEO 27 - WORDS TO AVOID IN REQUIREMENTS - WIEGERS (VIDEO)

*Trajanje: 4:31 minuta*

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

## ✓ Poglavlje 6

### Vežba

#### PISANJE I IZRAŽAVANJE ZAHTEVA

##### *Primeri izražavanja zahteva iz perspektive korisnika ili perspektive sistema*

Postoji više tehnika i stilova pisanja zahteva. Zahtevi se, takođe, mogu izražavati na različitim nivoima detaljnosti. Specifični po tom pitanju su projekti razvoja softvera na kojima se primenjuju agilne metode razvoja.

Dve perspektive koje se obično primenjuju u pisanju zahteva su **perspektiva korisnika** ili **perspektiva sistema**. Tradicionalno, zahtevi se uglavnom specificiraju tako da izraze ono što sistem treba da radi. To ih čini jasnim i nedvosmislenim. Noviji pristupi inženjeringu zahteva preporučuju izražavanje zahteva iz perspektive korisnika, posebno što se od korisnika očekuje da budu uključeni u proces razvoja. Da bi mogli da pomognu, oni najpre moraju da razumeju zahteve, a najjednostavniji način za to je postaviti zahteve iz njihove perspektive.

U nastavku vežbe dati su primeri pisanja zahteva, primenom pomenutih perspektiva, kao i primenom dveju različitih notacija.

##### **Perspektiva sistema**

Sistem treba da omogući izmenu podataka o zaposlenju registrovanog zaposlenog osoblja klinike.

Ako pacijent sa unetim JMBG brojem postoji u sistemu, sistem treba da prikaže sadržaj njegovog elektronskog kartona.

##### **Perspektiva korisnika**

Administrativni radnik treba da ima mogućnost izmene podataka o zaposlenju registrovanog zaposlenog osoblja klinike.

Medicinskom radniku treba da bude prikazan sadržaj elektronskog kartona pacijenta, ukoliko pacijent sa unetim JMBG brojem postoji u sistemu.

#### PISANJE ZAHTEVA

##### *Primeri pisanja zahteva primenom konvencije rednih brojeva i hijerarhijskih tekstualnih oznaka*

##### **Primena rednih brojeva**

REQ-80: Za sve izveštaje koji se generišu iz sistema, treba da postoji mogućnost generisanja u formi Word, PDF ili Excel fajla.

REQ-85: Sistem treba da omogući menadžeru generisanje izveštaja o rashodima napravljenim u određenom periodu.

REQ-86: Izveštaj o rashodima u određenom periodu treba da ima kao obavezne ulazne parametre početni datum za period i krajnji datum za period. Opcioni ulazni parametar je odeljenje klinike za koje se generiše izveštaj.

REQ-87: Ako se izveštaj o rashodima u određenom periodu generiše za sva odeljenja klinike, spisak rashoda treba da bude sortiran prema odeljenjima. Ako se izveštaj o rashodima u određenom periodu generiše za odabранo odeljenje, on treba da sadrži samo spisak rashoda za odabranu odeljenje. Za svaki rashod na spisku treba da bude naznačeno da li je rashod izmiren ili nije.

### **Primena hijerarhijskih tekstualnih oznaka**

*Izveštaji:* Generisanje izveštaja iz sistema

*Izveštaji.Format:* Za sve izveštaje koji se generišu iz sistema, treba da postoji mogućnost generisanja u formi Word, PDF ili Excel fajla.

*Izveštaji.RashodiZaPeriod:* Sistem treba da omogući menadžeru generisanje izveštaja o rashodima napravljenim u određenom periodu.

*Izveštaji.RashodiZaPeriod.ObavezniParametri:* Izveštaj o rashodima za period treba da ima kao obavezne ulazne parametre početni datum za period i krajnji datum za period.

*Izveštaji.RashodiZaPeriod.OpcioniParametri:* Izveštaj o rashodima za period treba da ima kao opcioni ulazni parametar odeljenje klinike za koje se generiše izveštaj.

*Izveštaji.RashodiZaPeriod.Soritanje:* Ako se izveštaj generiše za sva odeljenja klinike, spisak rashoda treba da bude sortiran prema odeljenjima. Ako se izveštaj generiše za odabranu odeljenje, treba da sadrži samo spisak rashoda za odabranu odeljenje. Za svaki rashod na spisku treba da bude naznačeno da li je izmiren ili nije.

## **PITANJA ZA DISKUSIJU**

### **Tekst pitanja za diskusiju**

#### **PITANJE 1.**

Pogledajte sadržaj templeta SRS dokumenta. Njegova namena nije u tome da dobijete na broju stranica i popunite baš svako poglavlje, već da osigurava da ćete sakupiti potrebne informacije za uspešan projekat. Taj šablon je koristan kao podsetnik i primer je dobre prakse dokumentovanja zahteva. Koja poglavlja nikada ne ispuštate iz vida kada utvrđujete zahteve? S druge strane, koja su to poglavlja koja zanemarujete i smatrate da bi trebalo da im posvetite više pažnje da biste potpunije dokumentovali zahteve? (10 min)

#### **PITANJE 2.**

Koju ćete konvenciju za označavanje pojedinačnih zahteva datih u tekstu predavanja odlučiti da primenjujete? Zbog čega? (5 min)

## ZADACI ZA VEŽBU

*Tekst zadataka za vežbu*

### **ZADATAK 1.**

Dopuniti zahteve za sistem privatne klinike, poštujući pravila data u primerima. Napišite zahteve koji se odnose na slučajeve korišćenja u kojima je akter Lekar. (15 min)

### **ZADATAK 2.**

Primenom odabrane konvencije za označavanje zahteva, popišite zahteve za ISUM-ov model E-student za svaki pojedinačan slučaj korišćenja koji ste identifikovali. Povedite računa o tome da nemate sukobljene, nekonzistentne ili nedovoljno jasne zahteve. (20 min)

### **ZADATAK 3.**

Napisane zahteve za E-studenta, preformulisati tako da budu izraženi potpuno korišćenjem samo jedne od ponuđenih perspektiva (korisnik ili sistem). (20 min)

## ✓ Poglavlje 7

### Domaći zadatak

#### DOMAĆI ZADATAK 8

##### *Tekst domaćeg zadatka*

Prođite kroz opisan slučaj upotrebe, koje ste napisali u šestoj nedelji nastave za sistem dodeljen za DZ01.

Na početku rada navedite koju ćete konvenciju označavanja zahteva primenjivati u radu i zbog čega. Opredelite se za izražavanje zahteva iz perspektive sistema ili perspektive korisnika. Takođe na početku obrazložite zbog čega ste napravili takav izbor.

Identifikujte potrebne funkcionalne zahteve iz prethodno opisanih glavnih tokova, iz preduslova, postuslova, poslovnih pravila i poslovnih zahteva. Specificirajte identifikovane funkcionalne zahteve. Vodite računa da pravilno označite sve funkcionalne zahteve, tako da ih je moguće lako pratiti. Proverite da li svaki od Vaših zahteva ima karakteristike odličnih zahteva.

*Napomene:*

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

#### ZADATAK ZA RAD NA PROJEKTU

##### *Tekst zadatka za rad na projektu*

Prođite kroz sve slučajeve upotrebe, koje ste za potrebe projekta napisali nakon šeste nedelji nastave. Pokušajte da dobijete potrebne funkcionalne zahteve na svakom koraku, iz preduslova, postuslova, poslovnih pravila i drugih zahteva. Specificirajte identifikovane funkcionalne zahteve (**Poglavlje 3 - Funkcije sistema**). Preporučeno je da pratite formu slučajeva korišćenja ili eventualno neku sličnu koja je predložena u uzorku SRS dokumenta. Vodite računa da pravilno označite sve funkcionalne zahteve, tako da ih je moguće lako pratiti. Proverite da li svaki od Vaših zahteva ima karakteristike odličnih zahteva.

Popunite i druge odeljke SRS dokumenta koje ste u mogućnosti nakon do sada pređenih materijala, na primer **Poglavlje 1 - Uvod** i **Poglavlje 2 - Opšti opis**. Dokument poslovnih zahteva i predlog projekta koji ste slali asistentu vam umnogome može pomoći da uradite ovaj deo. Proverite pažljivo svoj rad.

## ✓ Poglavlje 9

### Zaključak

## ZAKLJUČAK

1. Progresivno poboljšanje detalja je ključni princip efektivnog razvoja zahteva.
2. Specifikacija softverskih zahteva sadrži funkcije i sposobnosti koje softverski sistem mora da obezbedi, njegove karakteristike, i ograničenja koja mora da poštuje.
3. Svaki zahtev treba da ima jedinstveni identifikator. On se koristi kod svakog pozivanja na odgovarajući zahtev.
4. Hijerarhijski tekstualni tagovi su strukturisani, ukazuju na značenje i ne menjaju se pri dodavanju, brisanju ili pomeranju drugih zahteva
5. Postoje različiti uzorci SRS dokumenta, jer postoje i različite veličine projekata razvoj softvera. Na vama je da izaberete onaj koji najviše odgovara vašoj organizaciji i projektu.
6. Pri agilnom razvoju softvera, umesto funkcionalnih zahteva, najčešće se razvoj upravlja prema postavljenim testovima prihvatanja nastalim iz korisničkih priča.
7. Kompletност, tačnost, ostvarljivost, potreba, usklađenost sa prioritetima, nedvosmislenost, proverljivost - to su karakteristike dobro definisanih zahteva.
8. Kompletност, konzistentnost, promenljivost i sledljivost su karakteristike skupa zahteva izdanja.
9. Pri pisanju zahteva, možete ga pisati tako da kažete šta sistem treba da uradi (perspektiva sistema) ili šta korisnik treba da uradi (perspektiva korisnika).
10. Stil pisanja - prvo navedite iskaz o potrebi ili o funkcionalnosti, a onda detalje podrške \*razloge, izvor, prioritet, i druge atribute zahteva. Važno je pisati jednostavno i razumljivo.
11. Nivo detalja: Zahteve treba pisati na onom nivou detalja koliko je minimalno potrebno za njihovu primenu
12. Izaberite najefektniji način predstavljanja zahteva. Izbegavajte dvosmislenosti.
13. Postoje nekoliko tehnika da pronađete nedostajuće zahteve: Simetrija, složena logika i nedostajući izuzeci.

## REFERENCE

Nastavi 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 poglavljima 10 i 11** knjige: **Karl Wiegers, Joy Beatty, 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,



## SE322 - INŽENJERSTVO ZAHTEVA

Specifikacija zahteva za  
podacima i softverski alati za  
razvoj zahtevai upravljanje njima

Lekcija 10

PRIRUČNIK ZA STUDENTE

# SE322 - INŽENJERSTVO ZAHTEVA

## Lekcija 10

### ***SPECIFIKACIJA ZAHTEVA ZA PODACIMA I SOFTVERSKI ALATI ZA RAZVOJ ZAHTEVAI UPRAVLJANJE NJIMA***

- ✓ Specifikacija zahteva za podacima i softverski alati za razvoj zahtevai upravljanje njima
  - ✓ Poglavlje 1: Modeliranje podataka
  - ✓ Poglavlje 2: Rečnik podataka
  - ✓ Poglavlje 3: Analiza podataka
  - ✓ Poglavlje 4: Priprema izveštaja
  - ✓ Poglavlje 5: Alati za razvoj zahteva
  - ✓ Poglavlje 6: Alati za upravljanje zahtevima
  - ✓ Poglavlje 7: Izbor i primena alata
  - ✓ Poglavlje 8: Vežba
  - ✓ Poglavlje 9: Domaći zadatak
  - ✓ Poglavlje 10: Projektni zadatak
  - ✓ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

## ❖ Uvod

# UVOD

### *Uvodne napomene*

Računarski sistemi manipulišu podacima na način koji kupcima pruža vrednost. Gde god postoji funkcija, postoje podaci. Bez obzira da li podaci predstavljaju piksele u video igrama, paketi u pozivu na mobilni telefon, podaci o kvartalnoj prodaji kompanije, aktivnosti vašeg bankovnog računa ili bilo šta drugo, programska funkcionalnost je određena za kreiranje, modifikovanje, prikazivanje, brisanje, obradu i upotrebu podataka . Poslovni analitičar bi trebalo da počne sa prikupljanjem definicija podataka pošto se pojavljuju tokom iznošenja zahteva.

Dobro mesto za početak je ulazni i izlazni tokovi u kontekstnom dijagramu sistema. Ovi tokovi predstavljaju glavne elemente podataka na visokom nivou apstrakcije, koje BA može precizirati u detalje kako napreduje izazivanje zahteva. Imenice koje korisnici spominju tokom iznošenja zahteva često označavaju važne podatke: hemijski zahtev, zahtevalac, hemikalija, status, izveštaj o upotrebi. U ovoj lekciji su opisani načini za istraživanje i predstavljanje podataka koji su važni korisnicima vaše aplikacije, kao i načini za određivanje svih izveštaja ili nadzornih ploča koje aplikacija mora da generiše.

## UVODNI VIDEO

*Trajanje video snimka: 3min 44sek*

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

## ▼ Poglavlje 1

### Modeliranje podataka

#### VIDEO PREDAVANJE ZA OBJEKAT "MODELIRANJE PODATAKA"

*Trajanje video snimka: 13min 54sek*

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

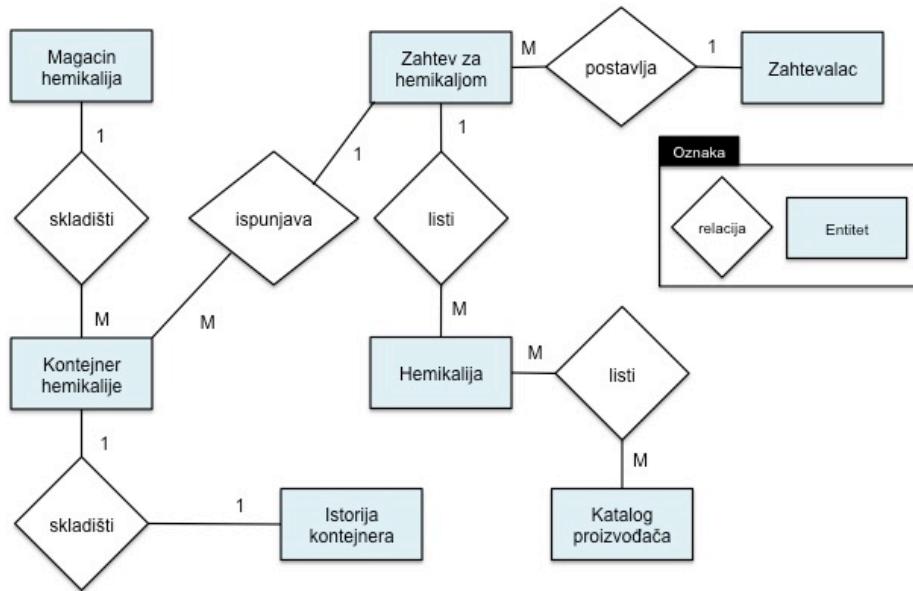
#### MODEL ENTITETA I RELACIJA

*Model entiteta i relacija povezuje entitete sa relacijama koje na krajevima daju kardinalnost, tj. broj primeraka entiteta koji se može povezati relacijom.*

**Model entiteta i relacija** (entity-relationship diagram - ERD) se često koristi za modeliranje podataka. Ovaj metod može da posluži kao alat za analizu zahteva o podacima. ERD analiza vam pomaže da razumete podatke i da koristite komponente podataka poslovanja ili sistema. Pomoću ERD vi u fazi projektovanja sistema kreirate logičku ili fizičku (primenjenu) strukturu podataka baze podataka sistema.

**Entiteti** (**entities**) predstavljaju fizičke stvari (uključujući ljude) ili skup podataka koji su važni za poslovanje koje analizirate ili za sistem koji nameravate da razvijete. Entiteti su označeni imenicom upisanom u pravougaonoj ERD dijagrama. Na slici 1 prikazan je deo modela entiteta i relacija (ERD) za slučaj Chemical Tracking System. Navedeni entiteti su isti kao i na dijagramu toka podataka prikazanog u lekciji 8. Ostali entiteti predstavljaju aktere koji su u interakciji sa sistemom. Svaki entitet ima više atributa, a instance entiteta imaju konkretnе vrednosti tih atributa.

**Relacije** (**relationships**) - predstavljene izokrenutim rombom - predstavljaju logičke veze između para entiteta. Relacije imaju ime koje pokazuje prirodu veze. Na slici 1 prikazan je model entiteta i relacija za slučaj zahtevanja hemikalije u sistemu Chemical Tracking System. Primena ERD pomaže u utvrđivanju zahteva da se bolje razumete sa kupcima ili korisnikom sistema oko zahteva koji se odnose na podatke.



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

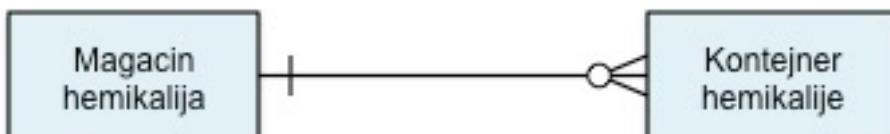
Slika 1.1 Model entiteta i relacija (ERD) u slučaju Chemical Tracking System

Kardinalnost (*cardinality*) označava mogući broj primeraka (instanci) jednog entiteta u vezi. Na slici 1 kardinalnost je obeležena ili brojem 1 ili M, gde je M neki broj veći od 1

## UML MODEL KLASA

*UML dijagram klasa na visokom nivou apstrakcije se koristi pri utvrđivanju zahteva podataka.*

Alternativno označavanje kardinalosti u relacijama je pokazano na slici 2. Relacija nije obeležena obrnutim rombom, već linijom iznad koje je upisan naziv relacije. Umesto brojeva ili slova koja označavaju kardinalnost koristi se grafička oznaka. Ako je kardinalnost 1, stavlja se samo jedna vertikalna crta, a ako je kardinalnost višestruka (M), stavlja se krug iz koga izlaze tri linije do entiteta.



Slika 1.2 Alternativno označavanje relacije i kardinalosti u ERD [Wiegers]

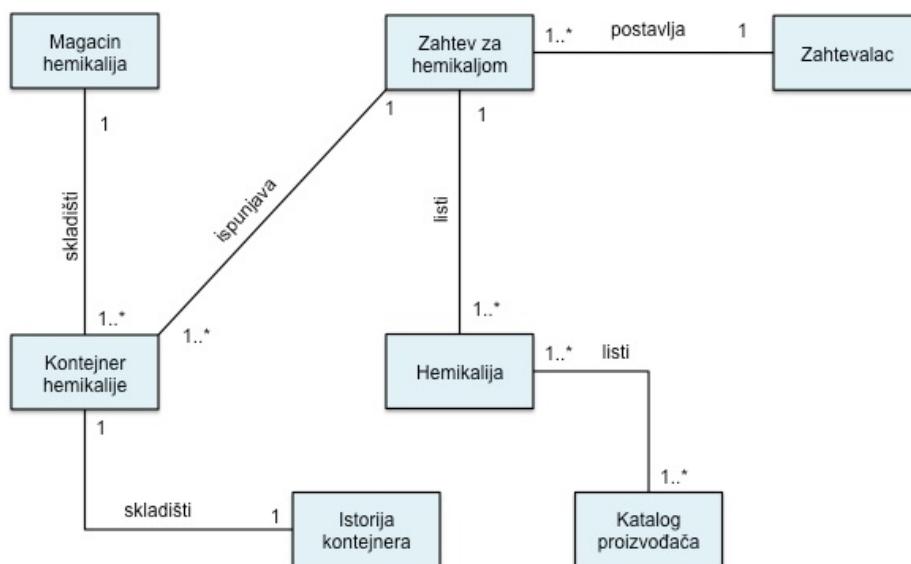
Ako razvijate objektno-orientisani softver, prirodnije je da umesto ERD dijagrama koristite UML dijagram klasa (slika 3)



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

Slika 1.3 Veza dve klase u Unified Modelling Language (UML)

Na slici 4 je prikazan alternativni model podataka sa slike 3, ali sada kao UML dijagram klasa.



Slika 1.4 UML dijagram klasa u slučaju Chemical Tracking System [Wieggers]

Pri otkrivanju zahteva, koriste se UML dijagrami klasa sa visokim nivoom apstrakcije. U fazi projektovanja sistema, one se detaljnije definiju. Dijagram klasa ili ERD, kada opisuju neku funkcionalnost, se obično povezuju sa odgovarajućim slučajem korišćenja ili sa pričom korisnika

## ✓ Poglavlje 2

### Rečnik podataka

#### CTS: REČNIK PODATAKA

*Rečnik podataka je kolekcija detaljnih informacija o entitetima podataka koji se upotrebljavaju u nekoj aplikaciji.*

Rečnik podataka (data dictionary) je kolekcija detaljnih informacija o entitetima podataka koji se upotrebljavaju u nekoj aplikaciji. To su informacije o kompoziciji entiteta, tipovima podataka, dozvoljenim vrednostima, podaci korisni sa potvrđivanje (validaciju) podataka, koji omogućavaju programerima da napišu tačan kod, i koji minimiziraju probleme integracije. Rečnih podataka je komplementaran sa spiskom termina projekta (**project glossary**) koji opisuje poslovne termine, skraćenice, i akronime upotrebljavane u nekom projektu. Rečnik podataka i spisak termina su dve odvojene stvari i tako treba i da ih koristite.

Za vreme analize, informacije u rečniku podataka predstavljaju elemente podataka i strukture domena aplikacije. Ove informacije ulaze u fazu projektovanja softvera kao šema baze podataka, tabele i atributi koje na kraju postaju imena promenljivih u programu. Ulog vremena u pripremu rečnika podataka nadoknadićete izbegavanjem grešaka u sistemu zbog različitog tumačenja podataka od strane učesnika u projektu. Ako redovno ažurirate rečnik, onda je to koristan alat koji će koristiti tokom životnog veka sistema. Održavanje rečnika podataka je značajna investicija u kvalitet softvera. Definicije podataka se često šire po komponentama sistema,. Upotrebom konsistentnih definicija podataka na nivou preduzeća, smanjuje greške i pri integraciji i kod interfejsa u sistemu. Kada je moguće, koristite standardne definicije podataka.

Ako u jedan rečnik podataka skupite i prikažete sve definicije podataka , onda će to znatno olakšati njihovo pretraživanje. Time se izbegavaju redundantnosti (ponavljanja) i nedoslednosti. Nije lako održavati integritet podataka koji se ponavljaju. Ako postoji samo jedan tačan i referentni primerak jedne definicije, onda je problem znatno olakšan. Slika 1 pokazuje deo rečnika podataka u slučaju Chemical Tracking System.

Element podataka	Opis	Kompozicija ili tip podataka	Dužina	Vrednosti
Zahtev za hemikalijom	Zahtev a novom hemikalijom iz magacina ili od proizvođača	ID zahteva: + Zahtevalac + Datum zahteva + Broj računa sa kog se plaća + 1:10 (zahtevana hemikalija)		
Lokacija isporuke	Mesto isporuke zahtevane hemikalije	Zgrada: + Broj laboratorije + Deo laboratorije		
Broj kontejnera	Broj kontejnera sa zahtevanom hemikalijom i zahtevanom veličinom	Pozitivan ceo broj	3	
Kvantitet	Količina hemikalije u kontejneru	broj	6	
Jedinica kvaliteta	Jedinica pridružena količini zahtevane hemikalije	slova	10	gram, kilogram, miligrami, svaki
ID zahteva	Jedinstveni identifikator zahteva	ceo broj	8	Sistemski generisan redni ceo broj, počevši od 0.
Zahtevana hemikalija	Opis zahtevane hemikalije	ID hemikalije + broj kontejnera + kvalitet + kvantitet + jedinica kvantiteta + (proizvođač)		
Zahtevalac	Informacija o pojedincu koji je postavio zahtev za hemikalijom.	Ime zahtevaoca + broj zaposlenog + odjeljenje + lokacija isporuke		
Ime zahtevaoca	Ime zaposlenog koji je podne zahtev za hemikalijom	slova	40	Može da sadrži prazna polja, zapete, apostrofe

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

Slika 2.1 Deo rečnika podataka u slučaju Chemical Tracking System

## TIPOVI PODATAKA

*Rečnik podataka sadrži najčešće sledeće tipove podataka: primitive, strukture i ponavljamajuće grupe.*

Rečnik podataka sadrži najčešće sledeće tipove podataka:

**Primitivi:** Primitivni element podataka je onaj ko se ne može ili ne treba dalje dekomponovati. U tabeli na slici 1 korišćeni su primitivni topovi podataka za sledeće atrbute: Kvantitet, Jedinica kvaliteta, ID zahteva i Ime zahtevaoca.

**Struktura:** Struktura podataka je sačinjeno od više elemenata podataka. Strukture podataka na slici 1 su: **Zahtev za hemikalijom**, **Lokacija isporuke**, **Zahtevana hemikalija** i **Zahtevalac**. Strukture mogu da sadrže i druge strukture. Struktura **Zahtevalac** sadrži strukturu: **Lokacija isporuke**. Svaki element podataka u strukturi mora biti definisan u rečniku podataka. Ako neki element strukture nije obavezan, onda se stavlja između zagrade.

Relink podataka može da koristi i hiperlinkove da bi povezao javljanje nekog termina sa njegovom definicijom u rečniku. Na slici 1 to je uređeno sa elementom podataka **količina**, koja se nalazi u okviru strukture: **Opis zahtevane hemikalije**.

**Ponavljamajuća grupa:** Ako se neki element podataka ponavlja više puta u nekoj strukturi, onda se to može skraćeno pisati tako, što se on stavi između dve velike zgrade, a ispred se stavi minimalni broj pojavlivanja : maksimalni broj pojavlivanja. Na slici 1 to je primenjeno na primeru elementa podatka **Zahtevana hemikalija**. Ona se pojavljuje u strukturu Zahtev za hemikalijom i naznačeno je 1:10{zahtevana hemikalija} što znači da može da ima od 1 do 10 ponavljanja u strukturi. Ako ne želimo da ograničimo broj ponavljanja, onda na mesto maksimalnog broj upišimo slovo n.

Nije jednostavno definisati podatke, čak i one jednostavne. Kako tumačiti velika, a kako mala slova, kako specijalne znake, kako koristiti razne alfabete raznih jezika i dr. Zato je potrebno sve to jasno definisati, kako bi programer znao kako da ih tumači.

# ✓ Poglavlje 3

## Analiza podataka

### CRUD MATRICA

*CRUD matrica je efikasan način za utvrđivanje zahteva koji nedostaju.*

Za vreme analize podataka vi upoređujete podatke iz različitih modela podataka i dijagrama, radi utvrđivanja grešaka, neslaganja, nekonzistentnosti i dr. To vam pomaže da dalje i dublje usmerite utvrđivanje zahteva podataka.

CRUD matrica je analiza kojom se utvrđuju nedostajući zahtevi. CRUD je skraćenica od Create, Read, Update, i Delete (kreiranje, učitavanje, promena i brisanje). CRUD matrica povezuje akcije sistema sa entitetima podataka da bi se pokazalo gde je i kako svaki značajniji entitet podataka kreiran, učitan, menjan i obrisan. Mogu se analizirati sledeće veze:

- Entiteti podataka i sistemski događaji,
- Entiteti podataka i zadaci korisnika ili slučajevi korisnika,
- Klase objekata i slučajevi korišćenja,

Slika 1 prikazuje CRUD matricu koja povezuje entitet podatka i slučaj korišćenja u slučaju Chemical Tracking System. Redove matrice čine slučajevi korišćenja i kolone - entiteti podataka. U poljima matrice se upisuju početna slova slova operacija kreiranja (C-create), čitanja (R-read), menjanja (U-update) i brisanja (D-delete). Posle kreiranja matrice pogledajte da li se neko od ovih četiri slova ne pojavljuje ni u jednoj ćeliji kolone. Na primer, ako se pokazuje da se podatak menja, a da nije prethodno kreiran, šta to znači?

entitet Slučaj koririšćenja	Zahtev	Hemikalija	Zahtevalac	Katalog prodavca
Postavljanje zahteva	C	R	R	R
Zahtev za promenu	U,D		R	R
Uređivanje zaliha hemikalija		C,U,D		
Izvešta po zahtevima	R	R	R	
Promena Zahtevaoca			C,U	

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

Slika 3.1 CRUD matrica za slučaj Chemical Tracking System

Primećujete da nijedna ćelija u koloni označena sa Zahtevalac ne sadrži slovo D (operacija brisanja). Postoji tri moguća razloga:

1. Brisanje Zahtevaoca nije funkcija koja se očekuje u CTS

2. Nedostaje nam slučaj korišćenja u kome se briše Zahtevalac.
3. Slučaj korišćenja Promena Zahtevaoca nije kompletan. Trebalo bi da dozvoli korisniku da ga obriše, ali to nije predviđeno u postojećem slučaju korišćenja.

## ▼ Poglavlje 4

### Priprema izveštaja

## VIDEO PREDAVANJE ZA OBJEKAT "PRIPREMA IZVEŠTAJA"

*Trajanje video snimka: 45min 6sek*

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

## PRIKUPLJANJE ZAHTEVA ZA IZVEŠTAJIMA

*Analitičar mora da utvrdi koje izveštaje korisnici očekuju od novog sistema.*

Mnoge aplikacije generišu izveštaje iz baza podataka, datoteka, i drugih informacionih resursa. Izveštaji mogu da budu formi tabela, grafikona, različitih dijagrama, ili sa bilo kakvim kombinacijama sa ovim formama izveštaja. Istraživanje sadržaja i formata potrebnih izveštaja je jedan važan aspekt razvoja zahteva. Specifikacija izveštaja povezuje zahteve (koja informacija ide u izveštaj i kako je organizovana) i projektovanje (kako bi izveštaj trebalo da izgleda).

### **Prikupljanje zahteva za izveštajima**

Uobičajena pitanja koje analitičar treba da postavi su sledeća:

- *Koje izveštaje sada koristite?* Moguće je da neke izveštaje preuzmete i za novi sistem.
- *Koji postojeći izveštaji treba da se izmene?* Novi sistem bi trebalo da takve izveštaje ostvari.
- *Koji se izveštaji sada generišu, ali se ne koriste?* Onda te izveštaje ne stavljajte u vaš novi sistem.
- Da li možete da mi kažete koji organizacijski, industrijski ili državni standardi i propisi zahtevaju određene izveštaje, bilo po sadržaju ili po izgledu. Prikupite te standarde i propise.

Daju se sledeće preporuke za pitanja radi utvrđivanja zahteva za izveštajima;

- Koji je naziv izveštaja?
- Koja je svrha izveštaja ili poslovna namera izveštaja? Kako primaoci izveštaja koriste isti? Koje se odluke donose na bazi ovih izveštaja i ko ih donosi?
- Da li je izveštaj rađen ručno? Ako jeste, koliko često i koja klasa korisnika ga radi?

- Koja je tipična a koja maksimalna veličina izveštaja?
- Da li postoji potreba da se na jednoj tabli prikaže nekoliko izveštaja i/ili grafova. Ako je potrebno, da li korisnici imaju potrebe da idu u dubinu izveštaja ili da dođu do nekog njegovog elementa?
- Kakva je raspoloživost izveštaja posle generisanja? Da li se prikazuje na monitoru, šalje nekome, prebacuje na proračunsku tabelu (npr. Excell), ili se automatski štampa? Da li se skladišta ili arhivira negde radi budućeg korišćenja?
- Da li ima bezbednosnih, privatnih, i poslovnih ograničenja koja ograničavaju pristup izveštaju pojedincima ili pojedinim klasama korisnika? Navedite relevantna poslovna pravila koja se bave sigurnošću.

## SPECIFIKACIJA IZVEŠTAJA

*Razmatrajte i druge mogućnosti , ispitajte raspoloživost podataka, uzmite u obzir i rast kompanije, mogućnost primene i dinamičkih, pored statičkih izveštaja.*

Daju se sledeće preporuke analitičaru koji ispituje potrebu za izveštajima:

**Razmatranje drugih mogućnosti:** Analitičar može korisniku da predloži da razmotre i druge mogućnosti pripreme izveštaja od one koje korisnik navodi kako bi povećali poslovnu vrednost izveštaja. Na primer, mogućnog drugačijeg redosleda podata u kolonama tabela, ili primena alata koji dozvoljava da korisnik menja taj redosled, ili izbor dobijanja sumarnih izveštaja koji sadrže koncizne rezultate više različitih izveštaja na jednom mestu, ili dubinski izveštaj, koji daje detalje o izvorima koji utiču na dobijene krajnje rezultate.

**Nalaženje podataka:** Proverite da li postoje podaci na osnovu koji treba sačiniti određeni izveštaj. Ova analiza može otkriti neke do sada nepoznate zahteve za pristup nekim podacima.

**Naslutite porast:** Ako se očekuje rast kompanije, možda će neki izveštaji morati da pretrpe izmene, jer su se bazirali na manjem broju podataka. Na primer, zbog većeg broja podataka, neke tabele postaju veće, te mora da se promeni format papira, sa "portret" na "landscape".

**Gledajte na sličnosti:** Različiti korisnici, pa i isti korisnik, često traže slične izveštaje. Pokušajte da ih objedinite u jedan koji bi zadovolji sve navedene potrebe. Mogu se koristiti parametri koji izdvajaju specifične zahteve.

**Razlika statičkih i dinamičkih izveštaja:** Statički izveštaj prikazuje podatke koji su dobijeni u određenom vremenskom trenutku. Dinamički izveštaj prikazuje podatke u realnom vremenu, te se mogu stalno menjati.

**Prototipovi izveštaja:** Izradom prototipa izveštaja možete korisniku pokazati njegov izgled radi dobijanja komentara. Korisnik može tada izneti i neka ograničenja, tj. proširiti listu zahteva podataka.

# UZORAK FORMULARA ZA SPECIFICIRANJE IZVEŠTAJA

*Daje se preporuka za specifikaciju izveštaja.*

Element izveštaja	Opis elementa
ID izveštaja	Broj, kod, ili oznaka koja označava ili klasificuje izveštaj
Naslov izveštaja	Naziv, pozicija u izveštaju, parametri upita
Svrha izveštaja	Poslovna potreba izveštaja, ili kratki opis projekta
Odluke koje se donose na osnovu izveštaja	
Prioritet	Prioritet izrade ovog izveštaja u odnosu na druge
Korisnici izveštaja	Klase korisnika koji generišu ili koriste izveštaj
Izvori podataka	Aplikacije, baze podataka, datoteke,
Frekvencija i raspolaganje	Statički ili dinamički izveštaj? Frekvencije izrade izveštaja. Koliko podataka ili transakcija je upotrebljeno? Koji su preduslovi izrade izveštaja? Automatska ili ručna izrada? Ko dobija izveštaj? Na koji način se isporučuje?
Vreme kašnjenja	Koliko će brzo izradi izveštaj po dobijanju zahteva? Koliko sveži podaci moraju da budu za izveštaj?
Vizualni raspored	"Landscap" ili "portret"? Veličina papira? Tipovi grafova, njihovi parametri...

Slika 4.1 Prvi deo uzorka za specifikaciju izveštaja [Wiegerts]

Element izveštaja	Opis elementa
Zagлавље и текст подноžја	Naziv izveštaja. Oznaka strane. Napomena izveštaja. Vreme korišćenja izveštaja. Ime osobe koja je pripremila izveštaj. Izvor podataka. Datum početka i završetka izrade izveštaja. Naziv organizacije. Nivo poverljivosti izveštaja.
Telo izveštaja	Kriterijumi za izbor podataka. Uključena polja tabela. Specifični nazivi polja, po zahtevu korisnika. Zaglavljiva kolona i redova, formati. Format prikazivanja (tip i veličina slova i dr.). Način rešavanja probijanja prostora za tekst i brojeve. Izvršeni obraćuni i transformacije. Kriterijumi sortiranja u tabeli. Kriterijumi filtriranja upita izveštaja pre njegovog izvršenja. Grupisanje i sumiranje brojeva. Formatiranje zbirova u redovima na prelomima. Kriterijumi određivanja stranica.
Indikator kraja izveštaja	Izgled i lokacija indikator kraja izveštaja.
Interaktivnost	Ako je dinamički izveštaj ili ako je kreiran interaktivno, koje opcije ima korisnik za njegovu promenu ili za promenu izgleda inicijalnog izveštaja. Kako se trajno smešta izveštaj između sesija njegovog korišćenja?
Ograničenja pristupa iz razloga bezbednosti	Navesti ograničenja za pojedince, grupe i organizacije u vezi pristupa ili generisanja izveštaja, ili izdvajaju pojedinih podataka iz izveštaja.

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

Slika 4.2 Drugi deo uzorka za specifikaciju izvestaja

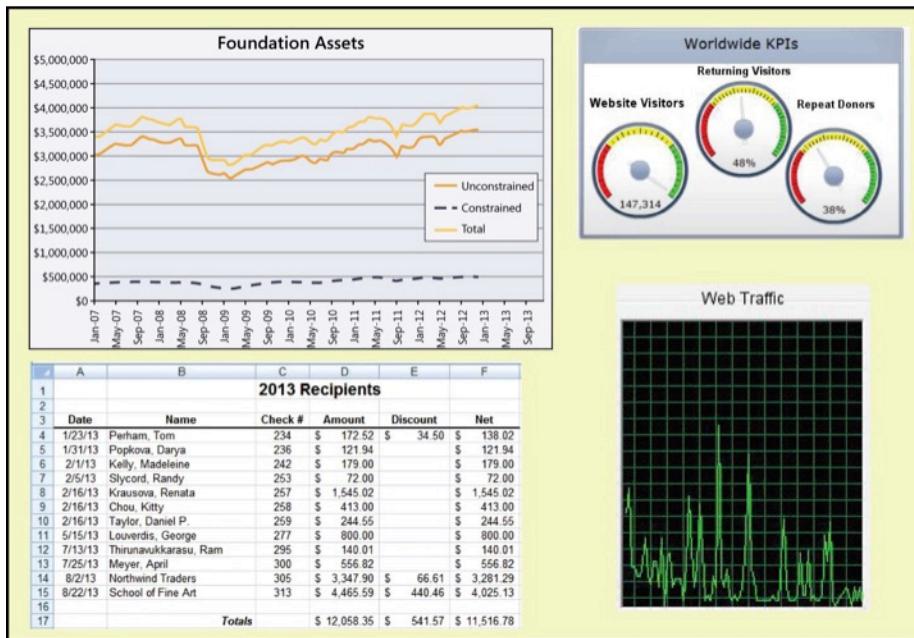
## TABLA ZA IZVEŠTAVANJE (DASHBOARD)

*Table za prikazivanje izveštaja daje jedan konsolidovan višedimenzionalni pogled na ono što se dešava u organizaciji ili u nekom procesu.*

**Tabla za prikazivanje izveštaja** (dashboard) je prikaz na monitoru ili odštampan izveštaj koji upotrebljava više tekstualnih i grafičkih predstavljanja podataka, da bi obezbedio jedan konsolidovan, višedimenzionalni pogled na ono što se dešava u organizaciji ili u nekom procesu. Kompanije na ovaj način prave koncizne uglavnom grafičke izveštaje koji na jednoj stranici ili monitoru istovremeno prikazuju o prodaji, troškovima, ključnim indikatorima performansi i sl. Mogu se prikazivati i dinamički izveštaji čiji se sadržaj menja u realnom vremenu. Pri kreiranju table sa izveštajima, koristite sledeće korake:

- Odredite informacije koje tabla za izveštavanje treba da sadrži. To zavisi od odluka koje prouzrokuje. Razumi te način korišćenja ovih podataka da bi primenio odgovarajući izgled.
- Utvrdite izvor podataka, njihovu pristupačnost i da li su statički ili dinamički
- Izaberite odgovarajući tip prikaza (tabela, grafikon, dijagram,...)

- Utvrdite optimalni raspored i relativnu veličinu različitih prikazanih izveštaja pogodna da korisnik lakše prihvati prikazane podatke
- Specificirajte detalje za svaki prikaz na tabeli za prikaz izveštaj. Uzmite u obzir sledeće teme:
  - Statički ili dinamički izveštaj? Frekvencija osvežavanja izveštaja?



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

Slika 4.3 Primer table sa izveštajima (dashboard)

- Da li korisnik želi da uslovno menja formatiranje izveštaja?
- Da li koristiti horizontalne ili vertikalne klizače?
- Da li korisnik može da uvećava izveštaj?
- Da li korisnik želi menja izgleda tabele i da menja tabele sa grafovima i suprotno?
- Da li korisnik želi da ulazi u dubinu izveštaja i traži detalje?

## VIDEO 23 - THE SOFTWARE REQUIREMENTS SPECIFICATION - WIEGERS (VIDEO)

*Trajanje: 11:30 minuta*

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

## VIDEO 28 - THE DATA DICTIONARY - WIEGERS (VIDEO)

*Trajanje: 8:24 minuta*

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

## ✓ Poglavlje 5

### Alati za razvoj zahteva

## ALATI ZA MENDŽMENT ZAHTEVIMA

*Alat za upravljanje zahtevima skladišti informacije u višekorisničku bazu podataka i rešava probleme koje prati ručno upravljanje zahtevima.*

Alat za upravljanje (menadžment) zahtevima skladišti informacije u višekorisničku bazu podataka i daje robusno rešenje problema koji se javljaju pri skladištenju zahteva u dokumentima. U velikim projektima vrlo je korisno imati mogućnost da korisnici preuzimaju zahteve iz izvornih dokumenata., da definišu vrednosti atributa, da filtriraju i prikažu sadržaj baze podataka, da isporučuju zahteve u različitim formatima, da definišu linkove za praćenje zahteva, i da povežu zahteve za stavke skladištene u drugim softverskim alatima. Ovi alati sa razvijeniji od alata za iziskivanje zahteva, jer je njihov zadatak tehnološki jednostavniji, a rezultat vidljiv i koristan.

#### Korist od upotrebe alata za upravljanje zahtevima (RM):

I ako ste razvili idelanu specifikaciju zahteva. može se desiti da izgubite kontrolu nad njim, kada je projekat u realizaciji, jer dolazi do promena, a i do zaboravljanja početnih zahteva. Tada je alat za upravljanje zahtevima rešenje, jer on obavlja sledeće zadatke:

- **Upravljanje verzijama i promenama:** Vaš projekat bi trebalo da definiše jedan ili više otisaka sistema (**baselines**). Svali otisak sistema utvrđuje specifičnu kolekciju zahteva sistema u određenom izdanju ili iteraciji. Nek RM alati imaju funkciju kreiranje ovih otisaka. Alat takođe održava istoriju promena svakog zahteva

Možete zapisati i razloge za izvršene promene i donete odluke, kao i da vratite na prethodnu verziju zahteva, ako je to neophodno. Neki alati koriste sistem za upravljanje promenama koji povezuje zahteve za promenama direktno sa pogodenim zahtevima

- **Skladištenje atributa zahteva:** Trebalo bi da opišete svaki atribut i zapište u alatu. Svako onda može da ga vidi, a ovlašćeni akteri i da ga menjaju. RM alati sami generišu neke sistemski definisane atributе, kao što su datum kreiranja zahteva i njegovu verziju pri svakom definisanju novih atributa. Definišu veze atributa sa svojim izdanjima, te korisnici mogu da ih selektivno biraju.
- **Olkšavanje analize uticaja** RM alati omogućavaju praćenje zahteva tako što vam omogućava da definišete vezu između različitih vrsta zahteva, između zahteva u različitim podsistemima i između pojedinačnih zahteva i srodnih komponenti sistema (na primer, projektna rešenja, kodni moduli, testovi i korisnička dokumentacija). Ove veze pomažu vam da analizirate uticaj koji će predložena promena određenog zahteva imati

na druge elemente sistema. Takođe je dobra ideja da se prate svaki funkcionalni zahtev do njegovog porekla ili roditelja, tako da znate odakle potiče. Na primer, možete tražiti da vidite listu svih zahteva koji potiču iz određenog poslovnog pravila kako biste mogli da procenite posledice promene tog pravila.

## NEDOSTACI RUČNO PРИPREMLJЕНИХ ИЗВЕШАТА

*Ručni način pripremanja izveštaja ima više nedostataka. Njihovo otklanjanje je moguće primenom softverskih alata za razvoj zahteva i za upravljanje zahtima.*

Ručni način pripremanja izveštaja se odvija tako što čovek, uz pomoć alata, kao što je MS Office, priprema dokument sa izveštajem. Takav način pripreme izveštaja ima više nedostataka:

- Teško je izveštaje ažurirati novim podacima i sinhronizovati ga sa izvorima podataka.
- Promena u izveštaju i obaveštavanje o tome ostalih članova tima se realizuju ručno
- Nije lako smestiti dodatne informacije - attribute - o svakom zahtevu
- Teško je definisati veze između zahteva i drugih elemenata sistema
- Komplikovano je pratiti status pojedinačnih i grupnih zahteva
- Kada se isti zahtev treba da koristi u različitim izdanjima izveštaja, analitičar to prebacivanje mora da radi ručno.
- Teško je da više učesnika projekta, naročito sa različitih lokacija, menja zahteve.
- Nema pogodnog mesta za smeštaj zahteva koji su razmatrani, ali odbačeni, ili koji su obrisani u startu.
- Teško je kreirati, pratiti, i menjati modele analize u istoj lokaciji kao zahteve.
- Teško je utvrditi nedostajuće ili nepotrebne zahteve.

Alati za razvoj zahteva (RD-Requirement Development tools) i alati za upravljanje (menadžment) zahtevima (RM-Requirement Management tools) daju rešenja za ova ograničenja. **Alati za razvoj zahteva** (RD) pomažu da utvrdite prave zahteve za vaš projekat i za procenu da li su ti zahtevi dobro napisani, tj. specificirani. **Alati za upravljanje zahtevima** (RM) pomažu vam da upravljate promenama zahteva, da pratite njihov status, i pratite njihovo korišćenje u drugim projektima. Mali timovi mogu da rade i bez ovih alata, ali u velikim projektima oni su vrlo korisni i potrebni.

Alat je samo alat. To znači on pruža pomoć analitičaru, ali može da bude koristan samo ako analitičar već ima jasan plan u vezi sa zahtevima i jasnu analizu. Članovi tima moraju da slede opisane (ručne) procese za razvoj zahteva, kao i da upravljaju zahtevima. Alati mogu samo da povećaju efikasnost razvoja zahteva i upravljanje zahtevima.

Pre nego što donešete odluku o kupovini alata, treba da napravite analizu isplativosti. S jedne strane imate troškove, pored cene licence, morate da uračunate i cenu održavanja softvera, ažuriranja, instaliranja i konfigurisanja, administriranje, podrška prodavca i konsalting, kao i obuka korisnika. S druge strane, njihova primena povećava efikasnost i kvalitet specifikacije zahteva, što se može odraziti na finansijske uštede kompanije,

# ALATI ZA RAZVOJ ZAHTEVA

*Ovi alati omogućavaju analitičaru da predstavi informaciju na više načina.*

Alati za razvoj zahteva koriste analitičaru u kontaktu sa akterima projekta radi izazivanja zahteva, tj. njihovog utvrđivanja i specificiranja i radi dokumentovanja zahteva. Postoje tri vrste alata za razvoj zahteva:

- alati za izazivanje zahteva
- alati za analizu zahteva, i
- alati za modeliranje zahteva.

## **Alati za izazivanje zahteva**

To su alati koje analitičar koristi za vreme razgovora sa korisnicima i drugim akterima. Mape uma mogu biti od pomoći za podstrek brejnstoming diskusijama i za organizaciju dobijenih informacija. Snimači zvuka mogu da snimaju razgovore radi kasnije analize njihovog sadržaja. Postoje alati koji tekstualni zapis odmah pretvaraju u zvučni, tako da delove razgovora možete i da na osnovu toga čujete. Postoje alati za analizu kvaliteta koji skeniraju dokument i traže nejasne ili kontradiktorne reči. Postoje i alati koji pretvaraju tekst u dijagrame, kao i alati za kolaborativno glasanje vezano za određivanje prioriteta zahteva.

## **Alati za izradu prototipova**

Postoje alati koje daju prototipove, počev od imitacije do simulacionog modela aplikacije koji radi. MS Power Point spada u prvu kategoriju, ili Visio, jer omogućavaju grafičku prezentaciju zahteva. Sofisticirаниji alati omogućavaju i prototip aplikacije koja radi. Neki alati podržavaju i kontrolu verzija, povezivanje zahteva, i generisanje koda. Treba voditi računu o isplativosti korišćenja ovih modela. Takođe, treba jasno reći kupcu da se radi samo o prototipu koji radi, ali koji nije finalni proizvod.

## **Alati za modeliranje**

Alati za modeliranje pomažu analitičaru da izradi modele, tj. dijagrame za definisanje i opisivanje zahteva, koje smo proučavali u ranijim lekcijama. Oni obično daju i standardne uzorke dijagrama i primere korišćenja dabi olakšali rad analitičaru. Na primer, Power Designer koji mi koristimo, automatski registruje podatke sa dijagram u svoju bazu podataka i kreira integriran model proizvoda.

Imajte u vidu da ni jedan alat ne može da vam kaže da li nedostaje neki zahtev ili element modela, da je logički nekompletan ili da je nepotreban. Ovi alati omogućavaju analitičaru da predstavi informaciju na više načina i da pronađe neke tipove grešaka i nedostataka, ali ne eliminiše potrebu razmišljanja i recenzije specifikacije zahteva.

# FUNKCIONALNOST ALATA ZA UPRAVLJANJE ZAHTEVIMA

*Alati znatno olakšavaju rad analitičarima zahteva.*

**Prepoznavanje nedostajućih i venserijskih zahteva:** Funkcionalnost praćenja u RM alatima pomaže zainteresovanim stranama da prepoznaju zahteve koji nedostaju, kao što su zahtevi korisnika koji nemaju mapirane funkcionalne zahteve. Slično tome, mogu otkriti zahteve koji se ne mogu razumeti razumnim poreklom, postavljajući pitanje da li su ti zahtevi neophodni. Ako je poslovni zahtev smanjen po obimu, onda se svi zahtevi proistekli iz njega mogu brzo smanjiti.

**Status zahteva za praćenje** Prikupljanje zahteva u bazi podataka omogućava vam da znate koliko ste specificirali diskretnih zahteva za proizvod. Praćenje statusa svakog zahteva tokom razvoja podržava celokupno praćenje statusa projekta.

**Alat za kontrolu pristupa:** RM vam omogućava da definišete dozvole pristupa pojedincima ili grupama korisnika i delite informacije sa geografski disperziranim timom putem veb interfejsa do baze podataka. Neki alati dozvoljavaju većem broju korisnika da istovremeno ažuriraju sadržaj baze podataka

**Zahtevi za ponovnu upotrebu:** Spremanje zahteva u bazu podataka olakšava njihovu ponovnu upotrebu u više projekata ili podprojekata. Zahtevi koji se logično uklapaju u više delova opisa proizvoda mogu se jednom sačuvati i na njih uputiti kad god je potrebno, da se izbegnu dupliranje zahteva.

**Komuniciranje sa zainteresovanim stranama:** RM alat služi kao glavni repozitorijum tako da svi akteri rade iz istog skupa zahteva. Neki alati omogućavaju članovima tima da elektronskim putem razgovaraju o problemima putem nepovezanih razgovora. Automatsko aktivirane poruke e-pošte obaveštavaju pogodene osobe kada se napravi novi unos u diskusiju ili kada se promeni određeni zahtev. Ovo je pogodna metoda za vidljivo praćenje odluka donetih u vezi sa zahtevima. Omogućavanje pristupa dostupnim na mreži može se minimizirati širenje dokumenata i zbrku verzija.

**Stanje izdanja zapisa:** Neki RM alati imaju funkcionalnost za praćenje otvorenih problema i povezivanje svakog broja sa njim povezanih potreba. Kako se problemi rešavaju, lako je utvrditi da li se zahtevi moraju ažurirati. Takođe možete brzo da pronađete istoriju problema i njegovo rešenje. Praćenje problema u alatu omogućava automatsko izveštavanje o statusu problema.

**Generisanje prilagođenih podskupova:** RM alati vam omogućavaju da izdvojite i pregledate skup zahteva koji odgovara određenoj svrsi. Na primer, možda ćete želeti izveštaj koji sadrži sve zahteve za određenu razvojnu iteraciju, sve zahteve koji se odnose na određenu osobinu ili skup zahteva koji se moraju pregledati.

## ▼ Poglavlje 6

### Alati za upravljanje zahtevima

## PREDNOSTI UPOTREBA ALATA ZA UPRAVLJANJE ZAHTEVIMA (RM) - 1. DEO

*RM postaje najvredniji kako vreme prolazi, a sećanja članova tima o detaljima zahteva izblede.*

Alat za upravljanje zahtevima (Requirement Management - RM) skladišti informacije u višekorisničku bazu podataka predstavlja robusno rešenje u odnosu na alternativu da se zahtevi skladište u dokumentima. Mali projektni timovi mogu se izvući samo unošenjem teksta zahteva i nekoliko atributa svakog zahteva, tj. da umesto alata, koriste dokumenta. Veći projektni timovi će imati koristi od omogućavanja korisnicima da uvoze zahteve iz izvornih dokumenata, definišu vrednosti atributa, filtriraju i prikazuju sadržaj baze podataka, zahteve za izvoz u različitim formatima, definišu veze za sledljivost i povezuju zahteve sa stavkama uskladištenim u drugim alatima za razvoj softvera.

Alati za upravljanje zahtevima dostupni su već duži niz godina. Oni su i obilniji i zreliji od alata za razvoj zahteva. Istone radi, problem koji rešavaju je uočljiviji. Lakše je stvoriti bazu podataka u kojoj se smeštaju zahtevi i pružaju neke mogućnosti za upravljanje njima nego pomoći BA-u da otkrije nova znanja, da to znanje razradi u precizne izjave i dijagrame zahteva i obezbedi da dobijene informacije predstavljaju tačne podatke. Neki alati kombinuju RD i RM mogućnosti u jedno snažno rešenje.

**Prednosti upotrebe RM alata** Čak i ako obavite sjajan posao u pronalaženju i određivanju zahteva svog projekta, možete izgubiti kontrolu nad njima kako razvoj napreduje. RM alat postaje najvredniji kako vreme prolazi, a sećanja članova tima o detaljima zahteva izblede. Sledеći odeljci opisuju neke od zadataka kojima vam ovaj alat može pomoći.

**Upravljanje verzijama i promenama** Vaš projekat treba da definiše jednu ili više osnovnih odrednica, od kojih svaka identifikuje posebnu kolekciju zahteva dodeljenih određenom izdanju ili iteraciji. Neki RM alati pružaju osnovne funkcije. Alat takođe održava istoriju promena koje su unete za svaki zahtev. Možete da zapišete obrazloženje iza svake odluke o promeni i vratite se na prethodnu verziju zahteva ako je potrebno. Neki alati sadrže sistem predloga promena koji povezuje zahteve za promenom direktno sa pogodenim zahtevima.

**Atributi zahteva za prodavnicu** Trebalо bi da zabeležite nekoliko opisnih atributa za svaki zahtev. Svi koji rade na projektu moraju biti u mogućnosti da vide atrbute, a izabranim pojedincima će biti dopušteno da ažuriraju vrednosti atributa. RM alati generišu nekoliko sistemski definisanih atributa, kao što su datum kada je zahtev stvoren i njegov trenutni broj verzije, i omogućuju vam definisanje dodatnih atributa različitih tipova podataka. Promišljena definicija atributa omogućava akterima projekta da odaberu podskupove zahteva na osnovu

određenih kombinacija vrednosti atributa. Atribut *Broj izdanja* jedan je od načina praćenja zahteva dodeljenih različitim izdanjima.

## PREDNOSTI UPOTREBA ALATA ZA UPRAVLJANJE ZAHTEVIMA (RM) - 2. DEO

*RM olakšava analizu uticaja zahteva, prepoznavanje nedostajajućih zahteva, praćenje statusa zahteva, kontrolu pristupa zahtevima, i komunikaciju sa akterima projekta.*

**Olakšajte analizu uticaja** RM alati omogućavaju praćenje zahteva tako što vam omogućava da definišete vezu između različitih vrsta zahteva, između zahteva u različitim podsistemima i između pojedinačnih zahteva i srodnih komponenti sistema (na primer, dizajni, kodni moduli, testovi i korisnička dokumentacija). Ove veze pomažu vam da analizirate uticaj koji će predložena promena određenog zahteva imati na druge elemente sistema. Takođe je dobra ideja da se prati svaki funkcionalni zahtev do njegovog porekla ili roditelja, tako da znate odakle potiče. Na primer, možete tražiti da vidite listu svih zahteva koji potiču iz određenog poslovnog pravila kako biste mogli da procenite posledice promene tog pravila.

**Prepoznavanje nedostajućih i nepotrebnih zahteva** Funkcionalnost praćenja u RM alatima pomaže zainteresovanim stranama da identifikuju zahteve koji nedostaju, kao što su zahtevi korisnika koji nemaju mapirane funkcionalne zahteve. Slično tome, mogu otkriti zahteve koji se ne mogu razumeti razumnim poreklom, postavljajući pitanje da li su ti zahtevi neophodni. Ako je poslovni zahtev smanjen iz obima, onda se svi zahtevi iz njega mogu brzo smanjiti.

**Status zahteva za praćenje** Prikupljanje zahteva u bazi podataka omogućava vam da znate koliko diskretnih zahteva koje ste naveli za proizvod. Praćenje statusa svakog zahteva tokom razvoja podržava celokupno praćenje statusa projekta.

**Alat za kontrolu pristupa** RM vam omogućava da definišete dozvole pristupa pojedincima ili grupama korisnika i delite informacije sa geografski disperziranim timom putem veb interfejsa do baze podataka. Neki alati dozvoljavaju većem broju korisnika da istovremeno ažuriraju sadržaj baze podataka.

**Komuniciranje sa akterima projekta:** RM alat služi kao glavni repozitorijum tako da svi akteri rade iz istog skupa zahteva. Neki alati omogućavaju članovima tima da elektronskim putem razgovaraju o problemima putem nitnih razgovora. Automatsko aktivirane poruke e-pošte obaveštavaju pogodjene osobe kada se napravi novi unos u diskusiju ili kada se promeni određeni zahtev. Ovo je pogodna metoda za vidljivo praćenje odluka donetih u vezi sa zahtevima. Omogućavanje pristupa dostupnim na mreži može minimizirati širenje dokumenata i zbrku verzija.

**Zahtevi za ponovnu upotrebu** Spremanje zahteva u bazu podataka olakšava njihovu ponovnu upotrebu u više projekata ili podprojekata. Zahtevi koji se logično uklapaju u više delova opisa proizvoda mogu se jednom sačuvati i na njih uputiti kad god je potrebno, da se izbegnu dupliranje zahteva.

# MOGUĆNOST ALATA ZA UPRAVLJANJE ZAHTEVIMA

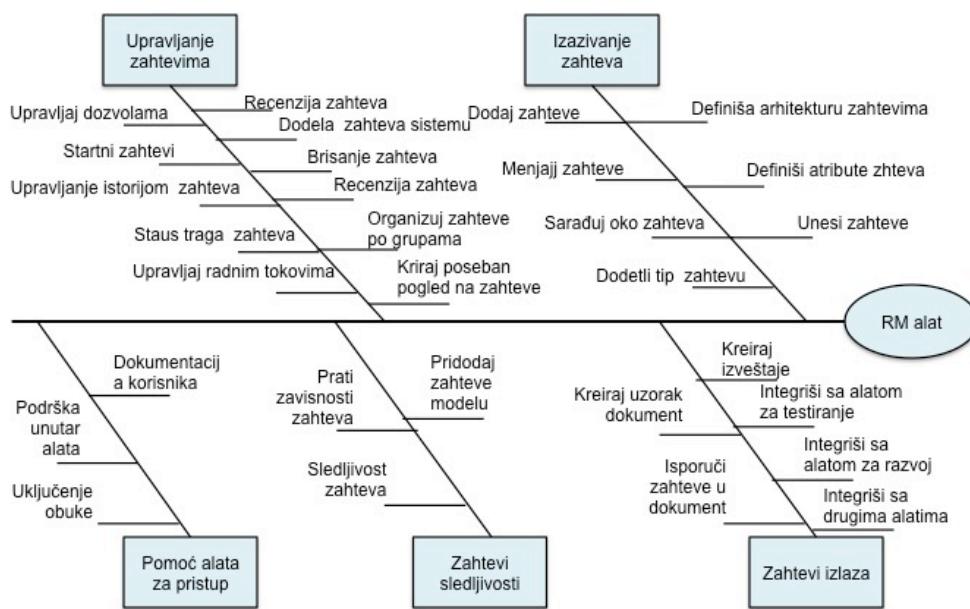
*Alati za upravljanje zahtevima svojim svojstvima vidno olakšavaju rad na upravljanju zahtevima*

**Stanje izdanja zapisa** Neki RM alati imaju funkcionalnost za praćenje otvorenih problema i povezivanje svakog broja sa njim povezanih potreba. Kako se problemi rešavaju, lako je utvrditi da li se zahtevi moraju ažurirati. Takođe možete brzo da pronađete istoriju problema i njegovo rešenje. Praćenje problema u alatu omogućava automatsko izveštavanje o statusu problema.

**Generisanje prilagođenih podskupova** RM alati vam omogućavaju da izdvojite i pregledate skup zahteva koji odgovara određenoj svrsi. Na primer, možda biste želeli izveštaj koji sadrži sve zahteve za određenu razvojnu iteraciju, sve zahteve koji se odnose na određenu osobinu ili skup zahteva koji se moraju pregledati.

## Mogućnosti RM alata

Stablo karakteristika na slici 1 prikazuje rezime vrsta mogućnosti koje se obično nalaze u RM alatima. Možete pronaći detaljne poređenja funkcija mnogih RM alata na Internetu



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

Slika 6.1 Uobičajena svojstva alata za upravljanje zahtevima

# SVOJSTA ALATA ZA UPRAVLJANJE ZAHTEVIMA

*RM alati omogućavaju definisanje različitih tipova zahteva, kao što su poslovni zahtevi, primeri korišćenja, funkcionalni zahtevi, hardverski zahtevi i ograničenja.*

RM alati vam omogućavaju da definišete različite tipove zahteva, kao što su poslovni zahtevi, primeri korišćenja, funkcionalni zahtevi, hardverski zahtevi i ograničenja. Ovo vam omogućava da razlikujete sve vrste informacija koje se obično nalaze u SRS-u. Mnogi alati vam omogućavaju da konfigurišete informacionu arhitekturu (koja definiše kako se tipovi zahteva i drugi objekti međusobno odnose) koja je prilagođena vašoj praksi. Većina alata pruža snažne mogućnosti za definisanje atributa za svaku vrstu zahteva, što predstavlja veliku prednost u odnosu na tipičan pristup zasnovan na dokumentima.

RM alati obično podržavaju hijerarhijske numeričke oznake zahteva, uz održavanje jedinstvenog internog identifikatora za svaki zahtev. Ovi identifikatori često se sastoje od kratkog tekstualnog prefiksa koji označava vrstu zahteva - kao što je UR za zahtev korisnika - a prati je jedinstveni celi broj. Neki alati nude prikaze koji vam omogućavaju da manipulišete hijerarhijskim stablom zahteva.

Zahtevi se mogu uvesti u RM alat iz različitih formata izvornih dokumenata. Tekstualni opis zahteva tretira se jednostavno kao obavezni atribut. Nekoliko proizvoda omogućava vam da u spremište zahteva ugradite netekstualne objekte kao što su grafika i proračunske tablice. Ostali proizvodi omogućavaju vam da povežete pojedinačne zahteve sa spoljnim datotekama (kao što su Microsoft Word datoteke, grafičke datoteke itd.) Koje pružaju dodatne informacije koje povećavaju sadržaj skladišta zahteva.

*Izlazne mogućnosti iz alata* obično uključuju mogućnost generisanja dokumenta sa zahtevima u različitim formatima, uključujući unapred definisane ili korisnički određene dokumente, proračunske tabele i veb stranice. Neki alati omogućavaju značajnu prilagodnju za kreiranje predložaka, omogućavajući vam da odredite izgled stranice, tekst na ploči, atribute koje treba izdvojiti iz baze podataka i stilove teksta koji ćete koristiti. Specifikacioni dokumenti su onda jednostavno izveštaji koji se generišu iz alata prema određenim kriterijumima upita, formatirani tako da izgledaju kao tipični SRS. Na primer, možete da kreirate SRS koji sadrži sve funkcionalne zahteve koji su dodeljeni određenom izdanju i dodeljeni određenom programeru. Neki alati pružaju funkciju koja omogućava korisnicima da izvrše promene izvezenih dokumenata van mreže, a zatim se sinhronizuju sa bazom podataka alata kada se korisnik vrati na mrežu.

Većina alata *omogućava stvaranje različitih prikaza zahteva* koji se generišu unutar alata ili izvoze iz alata. Funkcije obično uključuju mogućnost podešavanja korisničkih grupa i definiranja dozvola za odabrane korisnike ili grupe da kreiraju, čitaju, ažuriraju i brišu projekte, zahteve, atribute i vrednosti atributa. *Postavljanje odgovarajućih pogleda* i dozvola olakšava pregled zahteva i saradnju radi poboljšanja tih zahteva. Neki alati uključuju i pomagala za učenje, kao što su tutorijali ili uzorci projekata, kako bi se pomoglo korisnicima da postignu brzinu.

## INTEGRACIJA SA DRUGIMA ALATIMA

*Uz korišćenje alata pojedinačni zahtevi se mogu povezati sa objektima koji mogu biti u ovim drugim alatima*

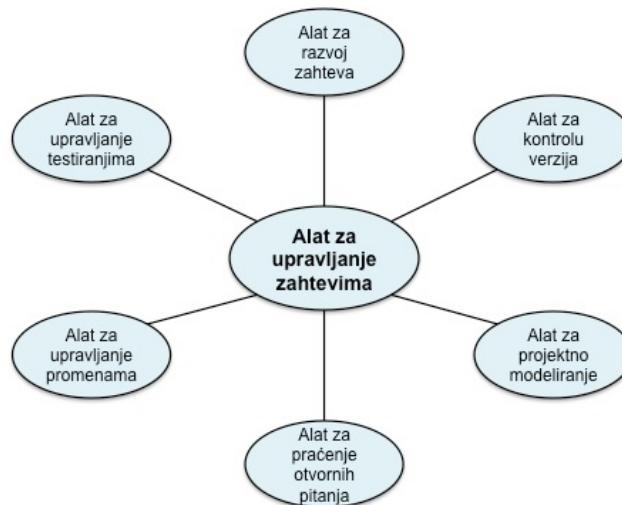
Alati za upravljanje zahtevima uglavnom imaju robusne funkcije praćenja. Praćenjem se vrši određivanje veza između dve vrste objekata ili objekata iste vrste. Neki alati za upravljanje

zahtevima uključuju mogućnosti modeliranja koje takođe omogućavaju da se modeli na nivou elemenata povezuju sa pojedinačnim zahtevima ili drugim elementima modela.

Neki agilni alati za upravljanje projektima takođe pružaju RM mogućnosti. Ovi alati se koriste za upravljanje i određivanje prioriteta zaostajanja, dodeljivanje zahteva iteracijama i generisanje test slučajeva direktno iz zahteva.

RM alati se često integrišu sa drugim alatima koji se koriste u razvoju aplikacija, kao što je prikazano na slici 2. Uz korišćenje alata pojedinačni zahtevi se mogu povezati sa objektima koji mogu biti u ovim drugim alatima. Na primer, možda ćete moći da pratite specifične zahteve za pojedinačne elemente dizajna smeštene u alatu za modeliranje dizajna ili za testove smeštene u alatu za upravljanje testovima.

Kada birate RM proizvod, utvrdite da li će alat moći da razmenjuje podatke sa ostalim alatima koje koristite. Razmislite o tome kako ćete iskoristiti ove integracije proizvoda dok izvodite zahteve, inženjering, testiranje, praćenje projekata i druge procese. Na primer, razmislite o tome kako biste definisali veze u tragovima između funkcionalnih zahteva i specifičnih dizajnerskih ili kodnih elemenata i kako biste proverili da li su svi testovi povezani nazad sa određenim funkcionalnim zahtevima uspešno izvedeni.



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

Slika 6.2 Integracija RM alata i drugih alata

## ▼ Poglavlje 7

### Izbor i primena alata

#### IZBOR ALATA

*Izaberite alat na osnovu kombinacije željenih funkcija, platforme i cena koje najbolje odgovaraju vašem razvojnom okruženju i kulturi*

Bilo koji od ovih alata za zahteve može unaprediti vaš rad na viši sofisticiraniji nivo Međutim, uspeh zavisi od izbora najprikladnijeg alata za vašu organizaciju i od vaših timova da to usvoje kao deo svoje rutinske prakse

Izaberite alat na osnovu kombinacije željenih funkcija, platforme i cena koje najbolje odgovaraju vašem razvojnom okruženju i kulturi. Poslovni analitičari treba da vode napore u izboru definisanjem kriterijuma za ocenu i vršenjem stvarne procene. Neke kompanije određuju procene alata za angažovanje konsultanata koji mogu sveobuhvatno proceniti potrebe kompanije i dati preporuke od kandidata za dostupne alate. Da biste sumirali postupak izbora:

1. Identifikujte zahteve vaše organizacije da alat služi kao kriterijum za ocenu.
  2. Postavite kriterijume na prioritete i odmerite ih u skladu sa mogućnostima ili drugim faktorima koji su najvažniji za vašu organizaciju.
  3. Podesite demonstracije ili nabavite evaluacione kopije alata koje želite da razmotrite.
  4. Svako uporedite kriterijume prema kriterijumima na dosledan način.
  5. Izračunajte ukupni rezultat za svaki alat koristeći svoje kriterijumske rezultate i utege koje ste im dodelili.
  6. Za svaki alat koji je postigao dobar rezultat, koristite ga na stvarnom projektu da biste videli da li se ponaša onako kako ste predviđali iz objektivnih rezultata.
  7. Da biste napravili konačni izbor, kombinujte ocene, troškove licenciranja i tekuće troškove sa informacijama o podršci dobavljača, podacima trenutnih korisnika i subjektivnim utiscima vašeg tima o proizvodima.
- Dva dobra finalna pitanja koja treba da postavite ljudima koji procenjuju alate su: „Koji alat biste najviše želeli da koristite?“ I „Koji alat bi vas najviše uzrujao ako bi bili primorani da ga koristite?“

## POSTAVLJANJE ALATA I PROCESA

*Kada nabavite alat za upravljanje zahtevima, očekujte da ćete morati da unesete neke promene u postupke, imena atributa i redosled aktivnosti procesa.*

Priznajte da će trebati napor da instalirate alat, u njega uložite zahteve projekta, definišete atributе i veze za praćenje, držite sadržaj aktuelnim, definišete pristupne grupe i njihove privilegije i prilagodite svoje procese za upotrebu alata. Konfigurisanje alata može biti složeno; postoji strma krivulja učenja samo da biste postavili sofisticirani alat za zahteve. Rukovodstvo mora da dodeli resurse potrebne za ove operacije. Obavežite se za celokupnu organizaciju da zaista koristite proizvod koji ste odabrali, umesto da dopuštate da postane "skup ukras" na polici. .

Čak i ako odaberete najbolji dostupni alat, neće nužno pružiti sve mogućnosti koje vaša organizacija želi ili treba. Možda ne podržava vaše postojeće postupke ili procese zahteva. I dalje ćete verovatno morati da prilagodite neke svoje postojeće procese da biste u njih ugradili alat. Očekujte da ćete morati da unesete neke promene u postupke, imena atributa i redosled aktivnosti aktivnosti razvoja. Razmotrite sledeće predloge za prevazilaženje problema procesa dok težite da maksimalizujete povraćaj ulaganja od alata za zahteve:

- Dodelite iskusnom BA da poseduje podešavanje alata i prilagodbe procesa. Razumeće uticaj izbora konfiguracije i promene procesa.
- Dobro razmislite o različitim vrstama zahteva koje definišete. Ne tretirajte svaki odeljak vašeg trenutnog SRS dokumenta kao zasebnu vrstu zahteva, ali nemojte samo da sav sadržaj SRS-a ubacujete u jednu vrstu zahteva.
- Koristite alat da olakšate komunikaciju sa zainteresovanim stranama na različitim lokacijama. Podesite privilegije pristupa i promene tako da dopustite dovoljan unos zahteva različitim ljudima ne dajući potpunu slobodu svima da menjaju sve u bazi podataka.
- Nemojte pokušavati da direktno unesete zahteve u alatu za upravljanje računarom tokom vaših radionica za rano izazivanje. Kako se zahtevi počinju da se stabilizuju. Međutim, njihovo skladištenje u alat čini ih vidljivim za usavršavanje učesnika radionice.
- Koristite RD alate tokom aktivnosti izvlačenja samo ako ste sigurni da oni neće usporiti postupak otkrivanja i izgubiti vreme zainteresovanih strana.
- Ne definišite veze u tragovima dok se zahtevi ne stabilizuju. U suprotnom, možete računati na to da ćete puno raditi na reviziji veza, jer se zahtevi nastavljaju razvijati.
- Da biste ubrzali prelazak sa paradigme zasnovane na dokumentima na upotrebu alata, odredite datum posle kojeg će se baza podataka alata smatrati konačnim spremištem potreba projekta. Nakon tog datuma zahtevi koji ostaju samo u dokumentima za obradu teksta neće biti priznati kao validni uslovi.

## OLAKŠAVANJA PRIHVATANJELATA OD STRANE KORISNIKA

*Kupovina alata je laka; mnogo je teže promeniti svoju kulturu i procese da biste prihvatili alat i iskoristili njegovu najbolju prednost.*

Važno Nemojte čak ni pilotirati upotrebu RM alata dok vaša organizacija ne kreira razumne specifikacije softverskog zahteva na papiru. Ako su vaši najveći problemi sa postavljanjem i pisanjem jasnih, visokokvalitetnih zahteva, RM alat vam neće pomoći (iako bi RD alat mogao).

Posvećenost korisnika vaših alata za zahteve je presudan faktor uspeha. Posvećeni, disciplinovani i dobro upućeni ljudi će ostvariti napredak čak i sa osrednjim alatima, dok najbolji alati neće platiti za sebe u rukama nemotivisanih ili loše obučenih korisnika. Ne kupujte alat ukoliko niste voljni da poštujete krivu učenja i uložite vreme. Kupovina alata je laka; mnogo je teže promeniti svoju kulturu i procese da biste prihvatili alat i iskoristili njegovu najbolju prednost.

Većini organizacija je već ugodno da nose beleške u dokumentu za obradu teksta ili ručno i da čuvaju svoje zahteve u dokumentima. Promena upotrebe alata zasnovanih na softveru zahteva drugačiji način razmišljanja. Korišćenje RD alata zahteva kršenje starih navika za pokretanje sesija za izazivanje. RM alat čini zahteve vidljivim svim učesnicima koji imaju pristup bazi podataka. Neki akteri tumače ovu vidljivost kao smanjenje kontrole koju imaju nad zahtevima, procesom inženjeringu zahteva ili oboje.

Neki radije ne dele sa svetom nepotpuni ili nesavršen skup zahteva, ali sadržaj baze podataka je tu da ih svi vide. Ako zadržite zahteve kao privatne dok ih „ne urade“, znači da propuštate priliku da drugi parovi očiju skeniraju zahteve zbog mogućih problema.

Ljudi su često otporni na promene stvari po kojima su upoznati i obično imaju komfor sa radom na zahtevima u dokumentima. Oni mogu imati percepciju - čak i ako je pogrešna - da će im korišćenje alata za zahtev biti teže. Takođe, ne zaboravite da je većina korisnika alata već zauzeta. Mora se izdvojiti vreme da se oni naviknu da koriste alat u svojim svakodnevnim poslovima. Na kraju, alat verovatno neće zahtevati više vremena od korisnika, ali prvo moraju da pređu krivu učenja i razviju nove radne navike pomoću alata.

## SAVETI ZA PRIDOBIJANJE KORISNIKA ALATA

*Setite se samo da alat ne može zameniti čvrst proces ili članove tima s odgovarajućim veštinama i znanjem.*

Evo nekoliko saveta koji će vam pomoći u rešavanju problema u vezi sa usvajanjem korisnika i promenama kulture:

- Identifikujte advokata za alate, lokalnog entuzijastu koji uči o dodavanjima alata, mentorije druge korisnike i vidi da se zaposli kako je planirano. Ova osoba treba da bude iskusan poslovni analitičar koji može biti jedini vlasnik za osiguravanje usvajanja alata. Ovaj inicijalni zagovornik alata sarađivaće s drugim korisnicima na njihovim projektima kako bi alat uključio u svoje svakodnevne aktivnosti. Zatim će obučiti i mentorisati druge da podrže alat dok ga drugi projekti usvoje.
- Jedan od najvećih izazova za prevladavanje usvajanja je to što korisnici ne veruju da će alat zaista dodati nikakvu vrednost. Možda nisu prepoznali bol zbog ograničenja u svojim postojećim priručnicima. Podelite sa njima priče o tome gde je nedostatak alata negativno uticao i zamolite ih da razmisle o svojim primerima.
- Članovi vašeg tima su pametni, ali bolje je da ih obučite nego da očekujete od njih da shvate kako će najbolje koristiti alat sami. Oni nesumnjivo mogu zaključiti osnovne operacije, ali neće naučiti o potpunom setu mogućnosti alata i kako ih efikasno iskoristiti.
- Budući da ne možete očekivati trenutne rezultate, ne bazirajte uspeh projekta na alatu koji prvi put koristite. Započnite s pilot primenom alata na nekritičkom projektu. Ovo će pomoći organizaciji da nauči koliko truda je potrebno za administraciju i podršku alata.

Širenje i povećana upotreba alata za pomoć u razvoju i upravljanju zahtevima predstavljaju značajan trend u softverskom inženjeringu koji će se nesumnjivo nastaviti. Previše organizacija, međutim, ne uspeva da iskoristi prednosti svojih ulaganja u takve alate. Oni ne razmatraju adekvatno kulturu i procese svoje organizacije i trud potreban za prelazak sa paradigme zahteva zasnovanog na dokumentima na pristup zasnovan na alatima. Uputstva u ovom poglavlju pomoći će vam da odaberete odgovarajuće alate i da ih efikasno koristite. Setite se samo da alat ne može zameniti čvrst proces ili članove tima s odgovarajućim veštinama i znanjem.

## VIDEO 28 - THE DATA DICTIONARY - WIEGERS (VIDEO)

*Trajanje: 8:24 minuta*

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

## VIDEO 31 - REQUIREMENTS MANAGEMENT TOOLS - WIEGERS (VIDEO)

*Trajanje: 5:02 minuta*

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

## ▼ Poglavlje 8

### Vežba

## REQVIEW - ALAT ZA ČUVANJE I UPRAVLJANJE ZAHTEVIMA

*Upoznavanje sa alatom i primer korišćenja*

Dokumentovanje zahteva u Word i Excel fajlovima nije uvek praktično, posebno kada su u pitanju veliki sistemi sa obimnom dokumentacijom. Tada mogu da budu jako korisni alati za skladištenje, praćenje i upravljanje zahtevima, bilo da su u desktop ili onlajn varijanti. Jedan od takvih alata je **ReqView**, koji se može preuzeti kao desktop aplikacija.



Slika 8.1 Početna stranica veb prezentacije ReqView alata za čuvanje i upravljanje zahtevima [Izvor: Marina Damnjanović]

## Software Download

ReqView 2.6.2

2019-05-28 [Release Notes](#)



Windows 64-bit Application  
Windows 7+

[Download](#)



Linux 64-bit Application  
Ubuntu 16.04+ LTS, Debian 9+ LTS

[Download](#)



Mac 64-bit Application  
macOS 10.12+

[Download](#)



Web Application  
Latest version of Chrome or Firefox

[Open](#)

Slika 8.2 Preuzimanje ReqView alata [Izvor: Marina Damjanović]

## REQVIEW - PREUZIMANJE ALATA

*Svaki korisnik može da preuzme demo desktop aplikaciju sa sajta*

Svaki korisnik može da preuzme demo aplikaciju sa sajta <https://www.reqview.com>. Dostupna je i besplatna PRO probna verzija u trajanju od 14 dana. Najbolji način da se iskoristi PRO verzija je da se izvrši prijava na njihovom sajtu, pri čemu se kao svrha prijave može odabratи edukacija.

## Request PRO Trial License

Free 14 days evaluation of all PRO features

### About You

<input type="text"/>	What is the purpose of your evaluation? *
<input type="text"/>	Organization name
<input type="text"/>	First and last name *
<input type="text"/>	Email *

We need your email address to send you the trial license and information on how to get started. We'll always treat your personal details as strictly confidential and never provide them to other companies.

[Privacy Policy](#)

Slika 8.3 Prijava za PRO verziju alata [Izvor: Marina Damjanović]

Demo desktop aplikacija ima ograničen spektar mogućnosti, ali je grafički korisnički interfejs jedan od boljih koji se može pronaći za besplatno preuzimanje. Prilikom prvog pokretanja alata, potrebno je da unesete nekoliko svojih podataka, a zatim vas sam alat kroz niz uputstava upoznaje sa svojim mogućnostima.

## Welcome



### ReqView

Version: 2.6.2, Build: eadd8ba2, Date: 2019-05-28

© Copyright Eccam s.r.o. All Rights Reserved.



#### Simple Yet Powerful Requirements Management Tool



##### Get Started

- [Requirements Project](#)
- [Review Document](#)
- [Edit Document](#)



##### Manage Requirements

- [Custom Attributes](#)
- [Traceability Links](#)
- [Track Changes](#)



##### Import / Export

- [Import from MS Word](#)
- [Import from MS Excel](#)
- [Export to HTML](#)

You are using unregistered FREE license with limited functionality.

Show next time

Try PRO

OK

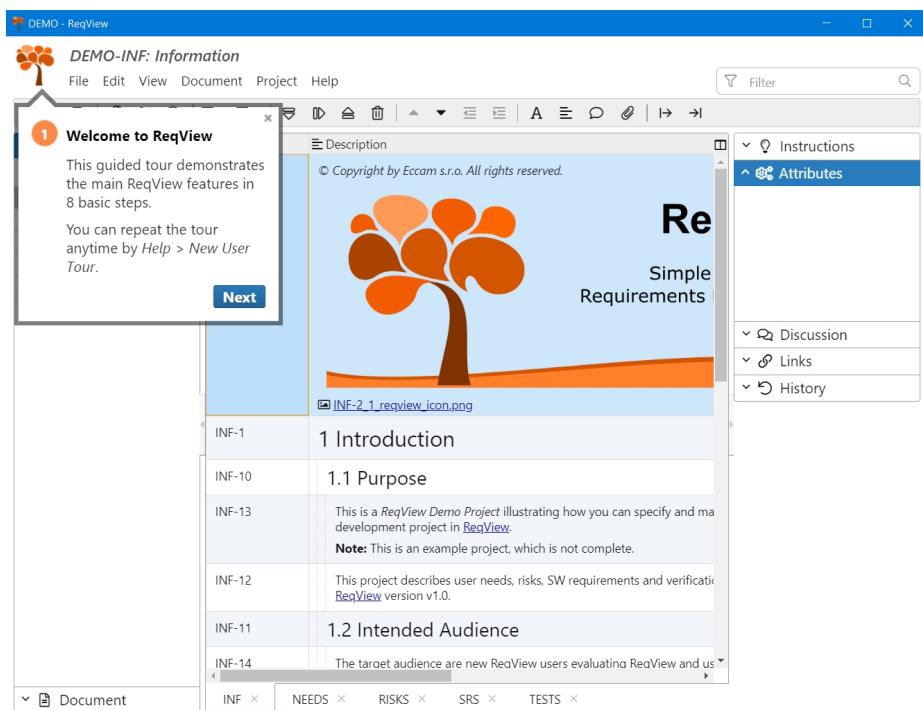
Slika 8.4 Demo desktop aplikacije [Izvor: Marina Damnjanović]

## REQVIEW - MOGUĆNOSTI ALATA

### *Mogućnosti demo verzije alata*

#### **Neke od mogućnosti koje pruža ReqView alat:**

- Skladištenje projektne dokumentacije (SRS, testovi, planovi...)
- Organizovanje svakog dokumenta po sekcijama. Svakoj sekciji je moguće pristupiti preko sadržaja dokumenta.
- Dokument se sastoji od objekata koji su hijerarhijski strukturirani. Svaki objekat predstavlja jedan zahtev. Savki objekat i njegov podobjekat su automatski označeni jedinstvenim ID-jem.
- Zahtev može imati proizvoljan broj atributa. Za zahtev se može vezati slika. Zahtevi se mogu proizvoljno menjati i brisati.
- Za svaki objekat se može vezati komentar, odnosno diskusija o tom zahtevu.
- Svaki objekat (zahtev) se može povezati (linkovati) sa nekim drugim dokumentom skladištenim u okviru projekta ili eksternim linkom.



Slika 8.5 Interfejs demo desktop verzije [Izvor: Marina Damnjanović]

## JOŠ ALATA ZA UPRAVLJANJE ZAHTEVIMA (1)

*Predstavljamo još nekoliko alata za upravljanje zahtevima koji nude besplatnu probnu verziju (1)*

**ReQtest** - <https://reqtest.com/>

Pored modula za upravljanje zahtevima, nudi modul za upravljanje testiranjem, modul za praćenje problema i agilnu tablu za vizualizaciju podataka.

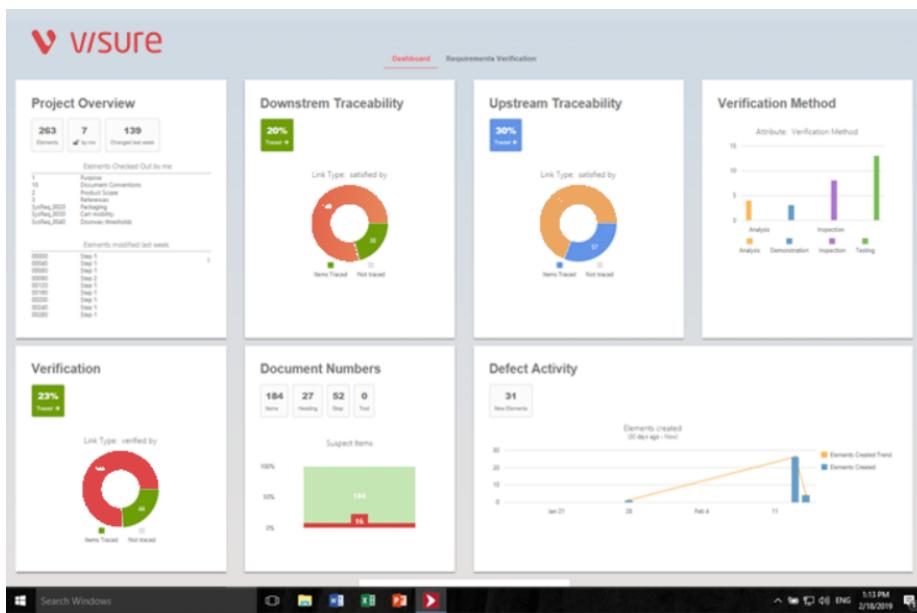
**Requirements management with full traceability**

- Prioritize & plan requirements effectively
- End-to-end requirements traceability
- Organize requirements effectively in Requirements Hierarchy
- Requirements visualization for actionable insights

Slika 8.6 Modul za upravljanje zahteva ReQtest alata [Izvor: Marina Damnjanović]

### Visure Requirements Management ALM platform - <https://www.visuresolutions.com/>

Jednostavan alat za prikupljanje zahteva, upravljanje zahtevima, njihovo praćenje i utvrđivanje. Integriše u istu okruženje podršku za druge procese, kao što su upravljanje rizicima, upravljanje testovima, praćenje defekata i upravljanje promenama. Usmeren je na vizualizaciju podataka.



Slika 8.7 Visure alat za prikupljanje, upravljanje i praćenje zahteva [Izvor: Marina Damnjanović]

## JOŠ ALATA ZA UPRAVLJANJE ZAHTEVIMA (2)

*Predstavljamo još nekoliko alata za upravljanje zahtevima koji nude besplatnu probnu verziju (2)*

**Modern Requirements** - <https://www.modernrequirements.com/>

Proizvod je Microsoft-a, često prepoznat kao Modern Requirements4DevOps. Predstavlja rešenje za upravljanje zahtevima i praćenje izmena u zahtevima i razvijen je kao ugrađeno proširenje alata Azure DevOps. Omogućava automatizaciju procesa, ima podršku za kolektivno pisanje dokumentacije i vizualizaciju.

Slika 8.8 Modern Requirements kao deo Azure DevOps alata [Izvor: Marina Damnjanović]

Više o korišćenju Modern Requirements možete pogledati na videu gde je prezentovana demo verzija.

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

## ZADACI ZA VEŽBU

*Tekst zadatka za vežbu*

### ZADATAK 1.

Preuzmite besplatnu demo verziju ReqView alata, pokrene ga i isprobajte njegove mogućnosti. Listu identifikovanih zahteva za E-student modul sa vežbe br. 8, prečitati u dokument u ovom alatu. Probajte da primenite različite mogućnosti alata, poput linkovanja drugih dokumenata i dodavanja slike i slično. (15 min)

### ZADATAK 2.

Pretražite koji su još popularni (i dobri) alati za upravljanje zahtevima. Izdvojite jedan koji Vam je najviše privukao pažnju i koji biste koristili. Obrazložite zbog čega biste isprobali baš taj alat. (15 min)

### **ZADATAK 3.**

Odaberite jedan alat koji daje besplatnu demo verziju i podnesite zahtev za dobijanjem pristupa demo verziji. Upravo taj alat treba da iskoristite ili za izradu domaćeg zadatka ili da ga na nekom od narednih časova prezentujete kolegama, kako biste svi naučili nešto novo. (15 min)

Primere aktuelnih alata možete naći na sledećim lokacijama:

- <https://blog.testlodge.com/requirements-management-tools-list/>
- <https://www.softwaretestinghelp.com/requirements-management-tools/>

## ✓ Poglavlje 9

### Domaći zadatak

#### DOMAĆI ZADATAK 10

##### *Tekst domaćeg zadatka*

Na raspolaganju su vam dve opcije za izradu DZ10.

Preuzmite kod kuće besplatnu demo verziju nekog alata za upravljanje zahtevima. Možete koristiti ReqView ili bilo koji drugi alat iz vežbe ili koji ste sami pronašli i uspeli da dobijete demo verziju.

##### **OPCIJA 1**

Ukoliko radite sa ReqView alatom, kreirajte dokument Slučajevi korišćenja i upišite u njemu slučajeve korišćenja koje ste identifikovali u DZ06. Zatim, kreirajte novi dokument pod nazivom Funkcionalni zahtevi i u njemu popišite funkcionalne zahteve koje ste identifikovali prilikom izrade DZ08 za jedan od slučajeva korišćenja sistema. Ako to niste već uradili u DZ08, obratite pažnju da zahtevi budu izraženi tako da imaju karakteristike kvalitetnih zahteva.

Povežite dokument Funkcionalnih zahteva sa slučajem korišćenja u dokumentu Slučajevi korišćenja na koji se ti zahtevi odnose. U dokumentu Slučajevi korišćenja dodajte u vidu slike svoj dijagram slučajeva korišćenja za sistem na kome radite.

Pošaljite fajlove, odnosno ReqView projekat predmetnom asistentu.

##### **OPCIJA 2**

U slučaju da koristite neki drugi alat za upravljanje zahtevima, pokušajte da примените što više zahteva koji su dati pod OPCIJOM 1.

Testirajte odabrani alat. Istražite njegove funkcionalnosti i kreirajte nekoliko pokaznih primera za rad sa alatom.

Kreirajte dokument koji će nositi naziv SE322-DZ12-Ime Prezime Indeks.docx i u njemu predstavite:

- alat koji ste koristili
- funkcionalnosti odabranog alata
- slikama ilustrujte navedene funkcionalnosti i svoje primere upotrebe alata

Pošaljite .docx fajl predmetnom asistentu.

Dodatna napomena:

Rok za izradu je definisan Plan i programom predmeta.

## ✓ Poglavlje 10

# Projektni zadatak

## ZADATAK ZA RAD NA PROJEKTU

### *Tekst zadatka za rad na projektu*

Obratite pažnju na **Poglavlje 4 - Zahtevi za podatke** u uzorku SRS dokumenta. U svom SRS dokumentu, koji ste započeli u prethodne dve nedelje nastave, popunite poglavlje 4. Uključite odgovarajući model podataka, bilo u samom poglavlju 4 ili kao dodatke koje ćete navesti pod **Dodatak B: Modeli analize**.

## ✓ Poglavlje 11

### Zaključak

## ZAKLJUČAK

1. Model entiteta i relacija povezuje entitete sa relacijama koje na krajevima daju kardinalnost, tj. broj primeraka entiteta koji se može povezati relacijom.
2. UML dijagram klasa na visokom nivou apstrakcije se koristi pri utvrđivanju zahteva podataka.
3. Rečnik podataka je kolekcija detaljnih informacija o entitetima podataka koji se upotrebljavaju u nekoj aplikaciji.
4. Rečnik podataka sadrži najčešće sledeće tipove podataka: primitive, strukture i ponavljajuće grupe.
5. CRUD matrica je efikasan način za utvrđivanje zahteva koji nedostaju.
6. Analitičar mora da utvrdi koje izveštaje korisnici očekuju od novog sistema.
7. Razmatrajte i druge mogućnosti, ispitajte raspoloživost podataka, uzmite u obzir i rast kompanije, mogućnost primene i dinamičkih, pored statičkih izveštaja, kao i korišćenje prototipa.
8. Dat uzorak dokumenta za specifikaciju izveštaja pruža preporuke za specifikaciju izveštaja.
9. Table za prikazivanje izveštaja daje jedan konsolidovan višedimenzionalni pogled na ono što se dešava u organizaciji ili u nekom procesu.
10. Ručni način pripremanja izveštaja ima više nedostataka. Njihovo otklanjanje je mogućno primenom softverskih alata za razvoj zahteva i za upravljanje zahtevima.
11. Alati za razvoj zahteva omogućavaju analitičaru da predstavi informaciju na više načina i da pronađe neke tipove grešaka, ali ne eliminiše potrebu razmišljanja i recenzije specifikacije zahteva.
12. Alat za upravljanje zahtevima postaje najvredniji kako vrieme prolazi, a sećanja članova tima o detaljima zahteva izblede. On olakšava analizu uticaja zahteva, prepoznavanje nedostajućih zahteva, praćenje statusa zahteva, kontrolu pristupa zahtevima i komunikaciju sa akterima.
13. Alati za upravljanje zahtevima svojim svojstvima vidno olakšavaju rad na upravljanju zahtevima. Omogućavaju definisanje različitih tipova zahteva, kao što su poslovni zahtevi, primeri korišćenja, funkcionalni zahtevi, hardverski zahtevi i ograničenja. Uz korišćenje alata pojedinačni zahtevi se mogu povezati sa objektima koji mogu biti u ovim drugim alatima.
14. Izaberite alat na osnovu kombinacije željenih funkcija, platforme i cena koje najbolje odgovaraju vašem razvojnom okruženju i kulturi. Kada nabavite alat za upravljanje zahtevima, očekujte da ćete morati da unesete neke promene u postupke, imena atributa i redosled aktivnosti procesa razvoja.
15. Kupovina alata je laka; mnogo je teže promeniti svoju kulturu i procese da biste prihvatali alat i iskoristili njegovu najbolju prednost.

## REFERENCE

Nastavi 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 13 i 30** 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,