

1

# Zahtjevi

2020/21.02

## ➤ ISO/IEC/IEEE 24765:2010 Systems and software engineering—Vocabulary:

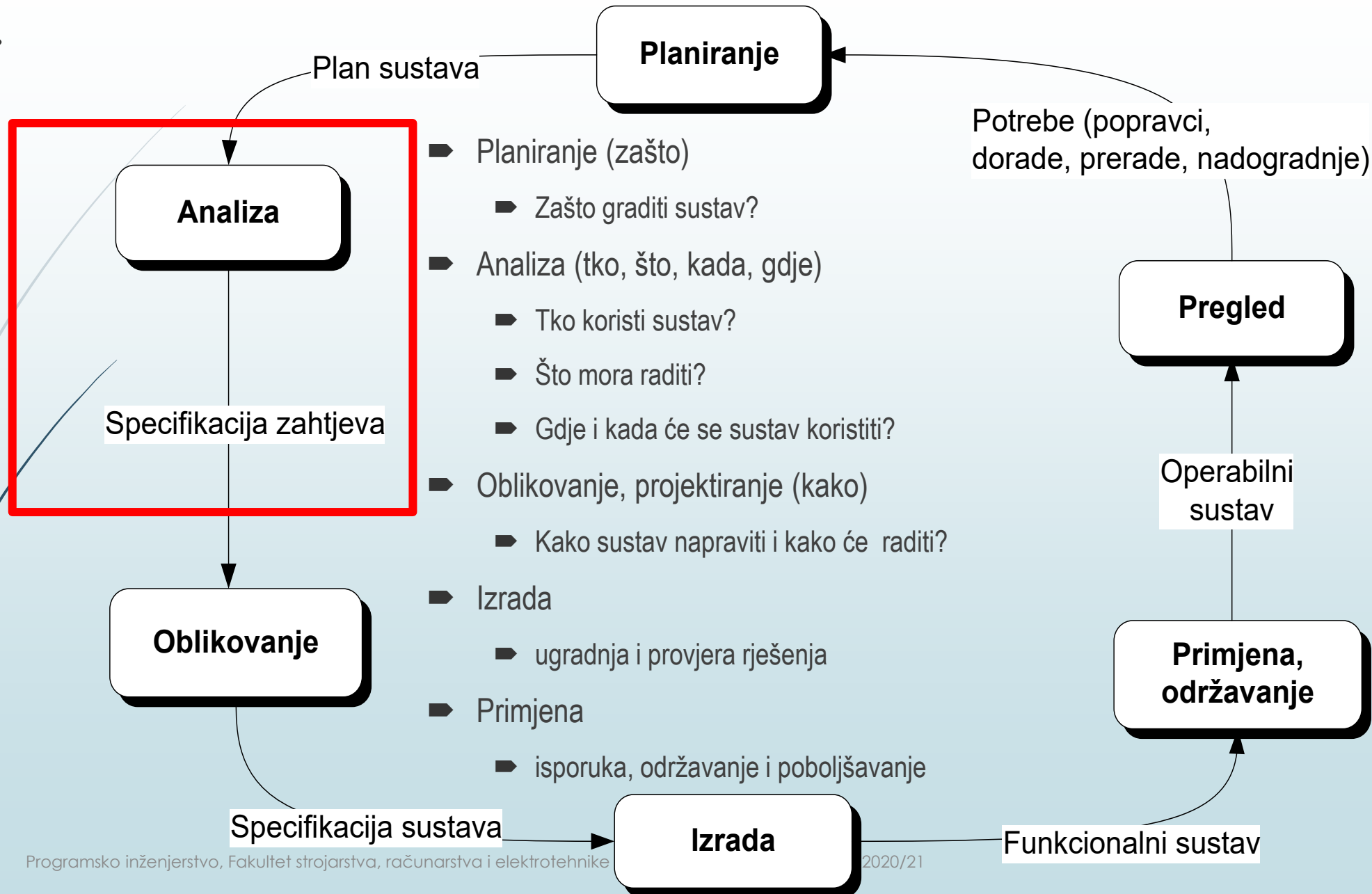
- uvjet ili sposobnost koje korisnik treba da bi riješio problem ili ostvario cilj.
- uvjet ili sposobnost koji mora posjedovati ili zadovoljiti sustav, komponenta sustava, proizvod ili usluga da bi zadovoljila ugovor, standard, specifikacije ili neki drugi ugovoreni dokument.
  - Dodatno, prema PMBOK zahtjevi uključuju nabrojane i dokumentirane potrebe, želje i očekivanja sponzora, korisnika i ostalih dionika u projektu

## ➤ ISO/IEC/IEEE 29148:2011 Systems and software engineering--Life cycle processes--Requirements engineering

- izjava kojom se prevodi ili izražava potreba i potrebi pridružena ograničenja i uvjeti

# Životni ciklus programske potpore

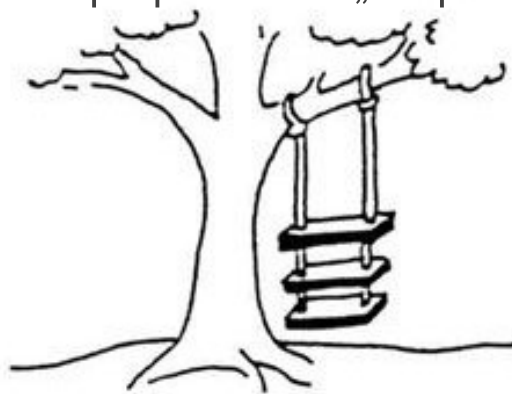
3



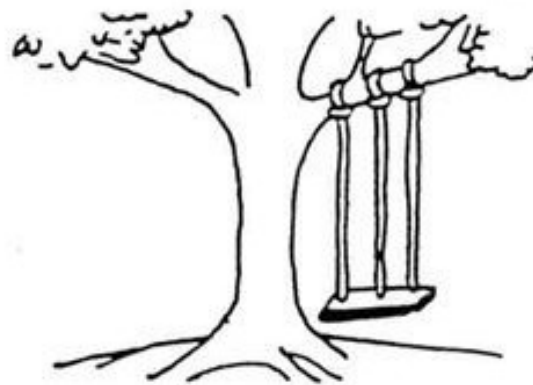
# Pogrešno postavljeni zahtjevi za posljedicu imaju neispunjena očekivanja

4

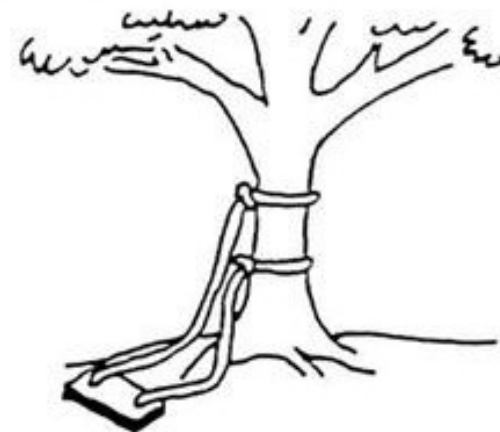
► ... a naknadne prepravke su „skupe”



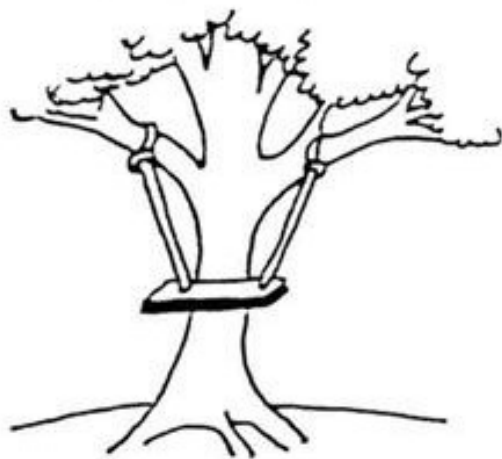
*As proposed by  
the project sponsors*



*As specified in  
the project request*



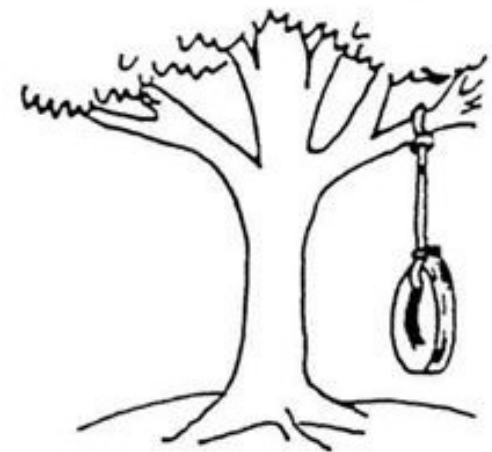
*As designed by  
the senior analyst*



*As produced by  
the programmers*



*As installed at  
the user's site*



*What the user  
wanted*

# Generalna podjela zahtjeva

5

- Generalna podjela na funkcionalne i nefunkcionalne
- Funkcionalni opisuju funkcionalnost (što softver mora raditi) te daju opise interakcije s korisnicima, drugim softverom ili hardverom (tko, što, koji podaci)
- Nefunkcionalni su vezani za svojstva ponašanja
  - performance, sigurnost, attribute kvalitete, ...
  - razna ograničenja (standardi, OS, ...)
  - najčešće se odnose na cijeli sustav, a ne na pojedinačno svojstvo
  - ponekad se nefunkcionalni zahtjevi nazivaju i pseudo-zahtjevi

# Analiza (a ne dizajn)

6

- Zahtjevi odgovaraju na pitanje ŠTO sustav mora raditi, bez ulaženje u implementacijske detalje
  - odnos ulaza i izlaza, sadržaj formi, ali ne izgled formi i razmještaj kontrola
  - slijed operacija, ali ne i detalji implementacije
  - popis provjera ulaznih podataka, ali ne i način prikaza poruka za neispravne podatke
  - što treba poslati vanjskom sučelju, ali ne i kako se šalje
- Rezultat analize služi kao podloga za dizajn
  - Dio odluka prilikom dizajna može biti posljedica rezultata analize, odnosno nefunkcionalnih zahtjeva
    - npr. podržani uređaji, format izvještaja i zapisa, licence, pravila, certifikati ...

# Vrste zahtjeva

7

- Osim generalne podjele na funkcionalne i nefunkcionalne neki autori razlikuju
  - [Sommerville] korisničke i systemske zahtjeve ovisno o detaljnosti zahtjeva i ciljanoj publici
    - Tko je korisnik? Krajnji korisnik ili vlasnik?
  - [Dennis, Wixom, Tegarden] poslovne i systemske ovisno o količini tehničkih detalja
  - ...
- Dobro oblikovani zahtjevi bi trebali imati sljedeću podjelu
  - **poslovni**
  - **korisnički**
  - **funkcionalni**
  - **nefunkcionalni**

# Poslovni zahtjevi

8

- Odgovaraju na pitanje zašto (se radi neki sustav)
  - predstavljaju ciljeve organizacije ili korisničke zahtjeve na višoj razini i ukratko opisuju problem koji treba riješiti
  - U idealnom slučaju zahtjevi vlasnika podudaraju se s poslovnim ciljevima!
- Sadržani u dokumentima u kojima se opisuje vizija i opseg projekta



# Primjeri poslovnih zahtjeva (1)

9

## ➤ Sustav za potporu rada udruga

- evidencija članstva i automatizacija postupka primanja novih članova neke udruge
- praćenje finansijskih podataka udruge i njenih članova
- poboljšanje procesa prodaje
- omogućavanje internetske prodaje
- podrška organiziranju natjecanja i okupljanja

# Primjeri poslovnih zahtjeva (2)

10

## ➤ Sustav subvencionirane prehrane

### ➤ Očekivana novčana ušteda

- Sustav mora biti tako koncipiran da prava na subvencioniranu prehranu može koristiti samo student koji ih je stekao i da ih može koristiti samo u svrhu prehrane.

### ➤ Sustav mora onemogućiti:

- korištenje subvencije od strane osoba koje nemaju na to pravo
- zaradu ilegalnih posrednika
- korištenje subvencije za druge svrhe osim prehrane
- naplatu usluga koje nisu pružene

## ➤ Zahtjevi krajnjih korisnika

- opisuju zadatke koje korisnik mora moći obaviti služeći se aplikacijama
- sadržani u opisima slučajeva korištenja, tj. opisima scenarija rada
- obično se izražavaju u obliku „Korisnik želi/treba/mora moći obaviti...”
  - Može (treba) se navesti i cilj, odnosno koristi koje korisnik ima od te akcije - opravdanje zahtjeva

# Primjeri korisničkih zahtjeva

12

- Korisnik mora moći ostvariti pravo na prehranu kod bilo kojeg pružatelja usluge
  - Novi sustav mora omogućiti da student ostvaruje svoje pravo kod bilo kojeg pružatelja usluge subvencionirane prehrane. Dosadašnja praksa je bila da svaki pružatelj usluga izdaje svoje bonove koji se mogu koristiti samo u određenim restoranima
- Korisnik treba plaćati obroke nakon korištenja pojedinog obroka.
  - Treba izbjeći bilo kakvo plaćanje od strane studenata za potrebe ostvarivanja prava, a posebice unaprijed.
- Korisnik mora moći prijaviti gubitak kartice
  - Potrebno je smanjiti rizik gubitka ostvarenih prava te sustav mora onemogućiti zluporabu stečenih prava.

# Funkcionalni zahtjevi

13

- Odgovaraju na pitanje što (se može/mora napraviti koristeći sustav)
  - definiraju softversku funkcionalnost (očekivano ponašanje i operacije koje sustav može izvoditi) koju treba ugraditi u proizvod da bi omogućio korisnicima obavljanje njihovih zadataka
  - posebno zanimljiva mogućnost programa (*feature*) – skup logički povezanih funkcionalnih zahtjeva koje korisniku omogućuju ispunjavanje poslovnih zahtjeva
- Primjeri:
  - *Nakon što se studentu jednom zavedu prava na matičnoj ustanovi, sustav mora proslijediti informaciju svim pružateljima usluga, odnosno omogućiti distribuirane upite*
  - *Sustav na dnevnoj bazi mora kreirati izvještaje sa statistikom prehrane po pružateljima usluge i vrsti obroka*

# Nefunkcionalni zahtjevi

14

- Odgovaraju na pitanje kako (ili kako dobro sustav mora raditi)
  - posljedica standarda, pravila i ugovora kojih se proizvod mora pridržavati
  - opisi vanjskih sučelja
  - zahtjevi na performanse
  - ograničenja na dizajn i implementaciju te svojstva kvalitete
    - preciziraju opis proizvoda navodeći karakteristike proizvoda u različitim dimenzijama koja su važne ili korisniku, ili razvojniku.
- Primjer
  - U sustavu prehrane nefunkcionalni zahtjevi mogu biti vezani za oblik korisničke kartice, protokol povezivanja, obvezu fiskalizacije itd...

# Kriteriji kvalitete zahtjeva

15

Zahtjevi moraju odražavati korisnikove želje (**ispravnost**) i sljedeće kriterije

- **Singularnost**

- Zahtjev ne smije biti kombinacija više zahtjeva

- **Potpunost**

- opisati sve moguće scenarije, uključujući i onu u slučaju pogreške

- **Konzistentnost**

- zahtjevi ne smiju međusobno biti kontradiktorni

- **Nedvosmislenost**

- svaki zahtjev se mora moći interpretirati na točno jedan način

- **Izvedivost** (ostvarivost)

- zahtjevi se moraju moći implementirati

- **Provjerljivost**

- za zahtjeve se mogu napisati (ponavljajući) testovi za provjeru ispravnosti

- **Sljedivost**

- Zahtjev mora se moći upariti s izvorom, povezanim zahtjevima ali i budućom implementacijom

# Izbjegavati nejasne i općenite izraze

16

- Primjeri riječi koje treba izbjegavati:
  - najviše, najbolje, lagano za korištenje, efikasno, *user friendly*, visoke kvalitete, on, taj, to, ako je moguće, primjereno, brzo, malo, ...
- Ovakvi izrazi se mogu interpretirati na različite načine i/ili nije moguće provjeriti je li zahtjev ispunjen



# Primjeri metrika za nefunkcionalne zahtjeve

17

## ➤ Brzina

- broj transakcija u jedinici vremena, vrijeme odziva

## ➤ Veličina

- broj ekrana, komponenti, vrijednost u (kilo/mega/...) bajtovima, ...

## ➤ Jednostavnost upotrebe

- vrijeme treninga, broj pogrešaka prilikom upotrebe

## ➤ Pouzdanost

- prosječno vrijeme prije kvara, postotak dostupnosti

## ➤ Robusnost

- vrijeme oporavka nakon pogreške, postotak događaja koji mogu uzrokovati pogrešku

## ➤ Portabilnost

- broj ili postotak podržanih uređaja (sustava)

# Primjer neostvarivog zahtjeva

18

- „Proizvod će se trenutno prebaciti između ispisivanja i skrivanja znakova koji se ne mogu tiskati.”
  - Računala ništa ne mogu napraviti trenutno te je ovaj zahtjev neostvariv.
  - Da li programska podrška sama odlučuje kad će se prebaciti iz jednog stanja u drugo ili je to inicirano akcijom korisnika?
  - Na koji dio teksta će se primijeniti promjena prikaza: da li samo označeni tekst, cijeli dokument ili nešto treće?
  - Jesu li „znakovi koji se ne mogu tiskati” skriveni znakovi, posebne oznake ili kontrolni znakovi? (dvosmisleno, nepotpuno)
- Bolji zahtjev:
  - „Korisnik će posebno dogovorenom akcijom, odabrati da li će se HTML oznake u trenutno otvorenom dokumentu prikazivati ili ne.”
  - Sad je jasno da je riječ o HTML oznakama te da korisnik može obaviti određenu akciju, ali nije točno navedeno kakvu (npr. kombinacija tipki, klik miša, potez prsta), što se prepušta dizajnerima.

# Primjer nepotpunog zahtjeva

19

- „Status pozadinsko posla dostavlja se u redovitim intervalima ne kraćim od 60 sekundi.”
  - Što je statusna poruka i pod kojim uvjetima će biti dostavljena? Koliko dugo ostaje vidljiva? Koji dio proizvoda će dostaviti poruku? Koliko dosljedni intervali moraju biti?
- Preciznije i detaljnije bi bilo
  - Modul za nadzor će ispisivati statusnu poruku u za to određeni dio sučelja.
  - Poruka će se ažurirati svakih 60 sekundi (plus minus 10 sekundi) nakon što započne izvođenje pozadinskog zadatka i bit će vidljiva cijelo vrijeme.
  - Ako se pozadinski zadatak izvodi normalno, modul za nadzor će ispisivati postotak obavljenog posla.
  - Modul za nadzor će ispisati „Zadatak obavljen.” nakon što se zadatak obavi.
  - Modul će ispisati poruku o pogrešci ukoliko dođe do zastoja u izvođenju.
- Problem je rastavljen u više zahtjeva jer će svaki zahtijevati posebno testiranje.
  - Ako je više zahtjeva grupirano u jedan lakše je previdjeti neki od njih tijekom izrade ili testiranja.
- Primijetiti da u zahtjevu nije detaljno opisano kako će se poruka i gdje ispisivati. To će biti odlučeno tijekom dizajna!

# Primjer nepotpunog i neprovjerljivog zahtjeva

20

- „Parser će brzo generirati izvješće o pogreškama HTML oznaka, koje omogućava brzi ispravak pogrešaka kada program koriste početnici u HTML-u.”
  - Zahtjev je neprovjerljiv! Kako testirati ovaj zahtjev?
    - Pronaći nekoga tko se smatra početnikom u HTML-u i zatim vidjeti kako brzo će, uz pomoć izvješća, ispraviti pogreške?
    - Što znači „brzo”? 5 sekundi ili 5 minuta?
  - Nije definirano što i kada se tvori izvješće i to čini zahtjev nepotpunim.
- Bolje:
  - *„Na zahtjev korisnika sustav će analizirati HTML datoteku te generirati izvješće koje sadrži broj linije i tekst pronađenih HTML pogrešaka, te opis svake pogreške.”*
    - Ako nema pogrešaka prilikom analize, neće se generirati izvješće

# Nekoliko primjera iz nedavne prakse (1)

21

- U postojećem sustavu forma je sadržavala polje za unos nalazišta i polje za unos staništa. Korisnik je u pisanom zahtjevu za promjenu naveo sljedeće:  
*„preimenovati opis nalazišta u opis nalazišta i staništa”*
- Treba li novi sustav imati samo jedno zajedničko polje za unos nalazišta i staništa ili 2 polja s nazivima „opis nalazišta i staništa” te „opis staništa”
- Razjašnjenje zahtjeva:
  - „Ostaju dva polja: 1. opis nalazišta i staništa te 2. Opis staništa, ali tako da na printanoj etiketi ne piše zasebno opis staništa već se pridružuje tekstu pod 1.”
  - Ne samo da prvi zahtjev nije bio jasan i nedvosmislen, nego je u sebi krio jedan inicijalno neotkriveni zahtjev

## Nekoliko primjera iz nedavne prakse (2)

22

- „Dodati opciju za unošenje podzbirki, mora biti moguće dodati nekoliko opcija uz jedan herbarijski primjerak”

=> „Uvesti herbarske podzbirke i omogućiti povezivanje herbarskog primjerka s više podzbirki”

- Može li herbarski primjerak istovremeno pripadati i zbirci i podzbirci?
- Može li herbarski primjerak pripadati podzbirkama različitih nadređenih podzbirki?

# Postavljanje prioriteta

23

- Nužno svojstvo - Da li korisnik nešto stvarno mora imati?
  - Postoji tendencija da se previše zahtjeva proglasi nužnim!
  - Po definiciji, ako sustav ne uključuje nužne zahtjeve, taj sustav ne može ispuniti svoju svrhu.
  - Treba testirati svaki zahtjev koji se smatra nužnim i probati ga rangirati.
    - Ako se zahtjev može rangirati onda nije obavezan!
    - Nužni zahtjevi se ne mogu rangirati jer su nužni za prvu verziju sustava!
- Poželjno svojstvo - Funkcije koje korisnik želi na kraju imati
  - Ranije verzije sustava mogu biti bez tih zahtjeva.
  - Poželjni zahtjevi mogu i trebaju biti rangirani.
- Neobvezna svojstva - Proizvoljni zahtjevi
  - Svojstva i mogućnosti bez kojih se može (npr. ostvarivanje ostalih prava iz studentskog standarda iz primjera zahtjeva krajnjih korisnika)
  - Iako bi ih lijepo bilo imati, to nisu pravi zahtjevi.
  - Ovi zahtjevi također mogu biti rangirani.
- Alternativno označavanje prioriteta: numerički, po verzijama, ...

# Postupci prikupljanja informacija (1)

24

- *Prikupljanje informacija se ne odnosi samo na fazu analize, već i planiranja i oblikovanja*
- Intervjui s ključnim korisnicima
  - uključiti predstavnike svih grupa korisnika i lokalne informatičare
  - zatvoreni (korisnici odgovaraju na unaprijed pripremljena pitanja) ili otvoreni
- Repertoar i vrste pitanja
  - Pitanja zatvorenog tipa: Koliko ...?, Kako se ... ?
  - Pitanja otvorenog tipa: Što mislite o ... ?, Koji su najveći problemi ... ?
  - „Probna” pitanja: Zašto?, Primjer?, Objašnjenje za ...
- Klasične poteškoće:
  - specifična terminologija nerazumljiva programskim inženjerima
  - ispuštanje informacija dobro poznatim korisnicima, ali ne i ostalima
  - razlučiti činjenice od mišljenja, provjeriti moguće kontradikcije
  - iznošenje „službene” istine, odnosi među ljudima i organizacijama
- Prikupljene informacije dopuniti iz drugih izvora: postojeća baza podataka, dokumentacija i (papirnat) obrasci, literatura, zakoni, propisi, ...



# Postupci prikupljanja informacija (2)

25

- Radne sjednice
  - zajednička analiza korisnika i analitičara, *brainstorming*
- Upitnici i ankete
  - Nadomjestak za intervju ili prikupljanje informacija o resursima.
- Analiza dokumentacije
  - Treba prikupiti sve dokumente značajne za poslovanje (pravila, strukture podataka)
- Promatranje (etnografski pristup)
  - Neposredni uvid u poslovne procese promatranjem radnih sredina
- Ostale tehnike
  - Prosudba postojećih aplikacija ili evidencija na računalu • analiza funkcionalnosti, strukture podataka te podataka koje treba “spasiti”
  - Prototipiranje – kada nema uzora ili korisnik ne zna što hoće
- Postupak analize mora biti prilagođen korisniku !

# Zahtjevi – što, a ne kako!

26

- Prijedlog rješenja nije zahtjev!
- Zahtjevi ne smiju sadržavati detalje dizajna ili implementacije
  - ali ako su nepotpuni onemogućuju planiranje projekta i praćenje promjena
- Usmjeriti se na ono što je potrebno obaviti, a ne na način realizacije
- Istražiti zašto korisnik predlaže određenu ugradnju – razumijevanje potrebe
- Ne izjednačavati „tako se uvijek radi” s „tako treba raditi”!
  - Može dovesti do cementiranja loših navika

# Dokumentiranje analize (zahtjeva) (1)

27

## ➤ Definicija zahtjeva

(*Requirements Definition*)

- izjava o stanju i ograničenjima sustava te potrebama
- narativni dokument namijenjen korisniku ili ga piše korisnik
  - poslovni i korisnički zahtjevi te njihovi prioriteti
  - uočeni problemi, ključne pretpostavke i preporuke rješenja

## ➤ IEEE standard ISO/IEC/IEEE 29148:2011

Stakeholder Requirements Specification  
StRS

## 1. Introduction

- 1.1 Business purpose
- 1.2 Business scope
- 1.3 Business overview
- 1.4 Definitions
- 1.5 Stakeholders

## 2. References

## 3. Business management requirements

- 3.1 Business environment
- 3.2 Goal and objective
- 3.3 Business model
- 3.4 Information environment

## 4. Business operational requirements

- 4.1 Business processes
- 4.2 Business operational policies and rules
- 4.3 Business operational constraints
- 4.4 Business operational modes
- 4.5 Business operational quality
- 4.6 Business structure

## 5. User requirements

## 6. Concept of proposed system

- 6.1 Operational concept
- 6.2 Operational scenario

## 7 Project Constraints

## 8. Appendix

- 8.1 Acronyms and abbreviations


# Dokumentiranje analize (zahtjeva) (2)

28

## ➤ Specifikacija zahtjeva *Requirements Specification*

- naziva se i funkcionalnom specifikacijom
- strukturirani dokument s detaljnim opisom očekivanog ponašanja sustava
- namijenjen ugovarateljima i izvoditeljima razvoja
- ugradbeno nezavisan pogled na sustav
  - funkcionalni i nefunkcionalni zahtjevi te njihovi prioriteti
  - model organizacijske strukture (strukturni dijagrami)
  - opis protoka dokumenata (dijagrami toka)
  - model procesa (dijagram toka podataka)
  - konceptualni model podataka (ERD)

## ➤ Primjeri:

-  Firma-Zahtjevi.doc (SpecifikacijaZahtjeva.dot)
- Software Requirements Specification SRS
  - IEEE standard ISO/IEC/IEEE 29148:2011

## 1. Introduction

- 1.1 Purpose
- 1.2 Scope
- 1.3 Product overview
  - 1.3.1 Product perspective
  - 1.3.2 Product functions
  - 1.3.3 User characteristics
  - 1.3.4 Limitations
- 1.4 Definitions

## 2. References

## 3. Specific requirements

- 3.1 External interfaces
- 3.2 Functions
- 3.3 Usability Requirements
- 3.4 Performance requirements
- 3.5 Logical database requirements
- 3.6 Design constraints
- 3.7 Software system attributes
- 3.8 Supporting information

## 4. Verification

(parallel to subsections in Section 3)

## 5. Appendices

- 5.1 Assumptions and dependencies
- 5.2 Acronyms and abbreviations

# Dokumentiranje analize (zahtjeva) (3)

29

- Agilne metode često umjesto formalnog dokumenta evidentiraju zahtjeve u obliku korisničkih priča
  - služe kao podloga za procjenu posla
  - razlažu se u manje zadatke
  - odabiru se za izradu po prioritetu
  - pogodno za sustave s nestabilnim ili nejasnim zahtjevima
- Prednosti:
  - jednostavno uređivanje
  - nije potrebno tehničko predznanje za evidentiranje
- Nedostatak:
  - korisnička priča je samo podsjetnik (obećanje) da će se taj zahtjev razraditi detaljnije
  - preporuča se u nekom trenutku imati jasno definiran dokument sa zahtjevima

- Uzastopni brojevi
  - novi zahtjev dobiva raspoloživi serijski broj (npr. Z01, Z02, ... ili FZ01,... ).
  - nepregledno, teško za pratiti veći broj zahtjeva, nepraktično brisanje
- Hijerarhijsko numeriranje (X.Y.Z)
  - jednostavno uređivanje (Word), problem brisanja koje uzrokuje posmak
  - poboljšanje: tekstovne oznake unutar brojčanih hijerarhija
    - Primjer: "3.2. - Funkcije editora", sa zahtjevima "ED-01", "ED-02", itd.
- Hijerarhijske tekstovne oznake
  - „objektno”, npr. ISPIS.KOPIJE.POTVRDA
  - prednost - strukturiranost, jednostavnost, nedostatak: nezgrapnost
- Potpunost
  - nijedan zahtjev ili informacija ne smiju nedostajati – teško uočljivo
  - kontrola „praćenjem” poslovnog procesa
  - nepotpune posebno označiti (npr. TBD - "to be determined")
    - • razriješiti prije ugradnje

- Scenarij – neformalni opis zamišljene interakcije korisnika i sustava
  - Korisničke priče kao jednostavna vrsta scenarija
- Tipični sadržaj
  - polazne pretpostavke koje vrijede na početku interakcije
  - normalni tok događaja
  - varijabilni (alternativni) ili iznimni tok događaja
  - popis drugih istovremenih aktivnosti koje mogu utjecati na trenutnu interakciju
  - stanje sustava nakon što interakcija završi



# Primjer scenarija korištenja

32

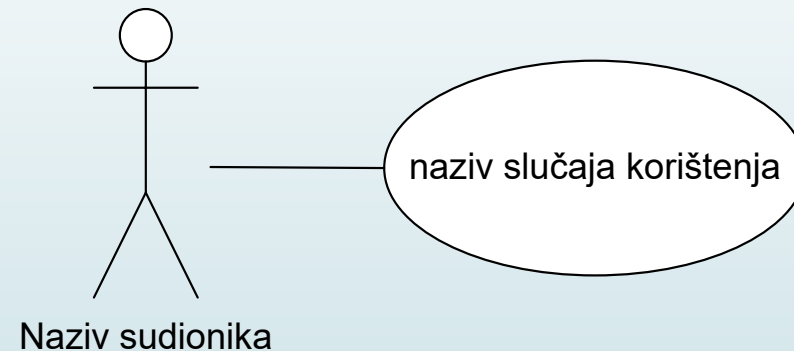
- Plaćanje proizvoda iz košarice (korisnička priča: *Korisnik mora moći platiti odabrane proizvode karticom*)
- Preduvjet: Kupac je dodao proizvode u košaricu te se odlučuje za plaćanje
- Normalni tok:
  1. Kupac unosi podatke o otpremi
  2. Sustav prikazuje punu informaciju o cijeni, uključujući otpremu
  3. Kupac unosi podatke o kreditnoj kartici
  4. Sustav autorizira kupnju
  5. Sustav potvrđuje prodaju interaktivno
  6. Sustav šalje elektroničko pismo potvrde
- Alternativa: Neuspješna autorizacija
  - U koraku 4, sustav ne uspijeva autorizirati karticu
  - Dozvoliti kupcu ponovni unos podataka o kartici
- Alternativa: Postojeći kupac
  - 1a. Kupac se prijavljuje te sustav prikazuje informaciju o otpremi i cijeni te zadnje četiri znamenke broja kreditne kartice
  - Kupac potvrđuje prikazane pretpostavljene podatke ili unosi nove
- **Primijetiti da scenarij ne sadrži implementacijske detalje**



# Dijagrami slučajeve korištenja

33

- engl. *use-case diagram*
- Jedan slučaj obično predstavlja jednu interakciju između sustava i okoline (sudionika)
  - Sudionik, glumac, akter, ...
- Svaki slučaj korištenje opisuje se scenarijem ili dijagramom slijeda (engl. *sequence diagram*)



- Pojedini slučajevi mogu
  - uključivati («include») zajedničke dijelove ponašanja
  - biti proširenje («extend»), tj. specijalni slučaj nekog postojećeg slučaja

34

- 
- ```
graph TD
    Kupac((Kupac))
    Administrator((Administrator))
    Prijava(Prijava korisnika)
    Registracija(Registracija korisnika)
    Placanje(Plaćanje)
    Izracun(Izračun troškova dostave)
    Pregled(Pregled narudžbi)

    Kupac --> Prijava
    Kupac --> Registracija
    Kupac --> Placanje
    Administrator --> Pregled

    Registracija -.->|<<extend>>| Prijava
    Placanje -.->|<<extend>>| Registracija
    Placanje -.->|<<include>>| Izracun
```

- Provjera dokumentacije sa zahtjevima
  - jedna od najkorisnijih softverskih tehnika
  - mikro ekipa (analitičari, projektanti, ... , korisnici) ispituje specifikacije i modele
- Pisanje testova, probnih slučajeva
  - prototipiranje, izvođenje scenarija korištenja – provjera očekivanog ponašanja
  - slučajeve povezati s funkcionalnim zahtjevima da se osigura potpunost
- Pisanje korisničkog priručnika
  - skica korisničkog priručnika - kao specifikacija ili dio analize
  - dobre upute opisuju svu vidljivu funkcionalnost – ostalo je u SRS
- Definiranje kriterija prihvatljivosti
  - definiranje (s korisnicima) uvjeta pod kojima će proizvod zadovoljiti zahtjeve
  - na osnovu slučajeva/scenarija korištenja

# Upravljanje zahtjevima (1)

36

- Definiranje postupka za promjenu zahtjeva
  - postupak kojim se novi zahtjev ili promjena postojećeg analizira i prihvaća
  - predložene promjene moraju slijediti unaprijed definiranu proceduru
- Uspostava odbora za promjene (*Change Control Board*, CBB)
  - ključni članovi projekta
  - odlučuje o usvajanju zahtjeva te postavlja prioritete i rokove
- Analiza utjecaja promjena zahtjeva
  - procjenjuje se utjecaj promjene na organizaciju, raspored ili drugo
  - služi donošenju dobrih (ispravnih, mogućih) odluka o (ne)prihvaćanju promjena
- Praćenje promjena zahtjeva na svim proizvodima
  - za prihvaćenu promjenu se kroz matricu praćenja zahtjeva pronalaze sve ovisne komponente (izvorni kod, testni slučajevi, neki drugi zahtjev,...).

# Upravljanje zahtjevima (2)

37

- Uspostava vremenske osnovice (baseline) i kontrole verzija
  - definiranje slike dogovorenih zahtjeva u određenom trenutku
  - nakon osnovice, zahtjevi prolaze postupak promjena
  - verzioniranje specifikacije – uklanjanje neodređenosti
- Praćenje povijesti promjena zahtjeva
  - zapis o vremenu, vrsti i sadržaju, razlozima, verziji i autoru promjene
- Praćenje statusa zahtjeva
  - status (predložen, odobren, ugrađen, provjeren) + statistika
- Mjerenje stabilnosti zahtjeva
  - brojanje promjena – mjera da je problem shvaćen, opseg definiran
- Korištenje alata za upravljanje zahtjevima
  - evidencija zahtjeva, međusobne povezanosti i statusa
  - verzioniranje zahtjeva i dokumentacije
- **Inženjerstvo zahtjeva** (requirements engineering) = razvoj zahtjeva (određivanje zahtjeva, analiza, specificiranje (dokumentiranje) i verifikacija) + upravljanje zahtjevima.