Lekcija 3 - Osnove upravljanja projektima

Glavni zadaci kod upravljanja projektima

- Planiranje projekta
- Estimacija i kreiranje rasporeda
- Rukovanje resursima
- Nadzor projekta i sakupljanje metrika (merenje)
- Pregled projekta i prezentacije
- Analiza efikasnosti upravljanja projektom

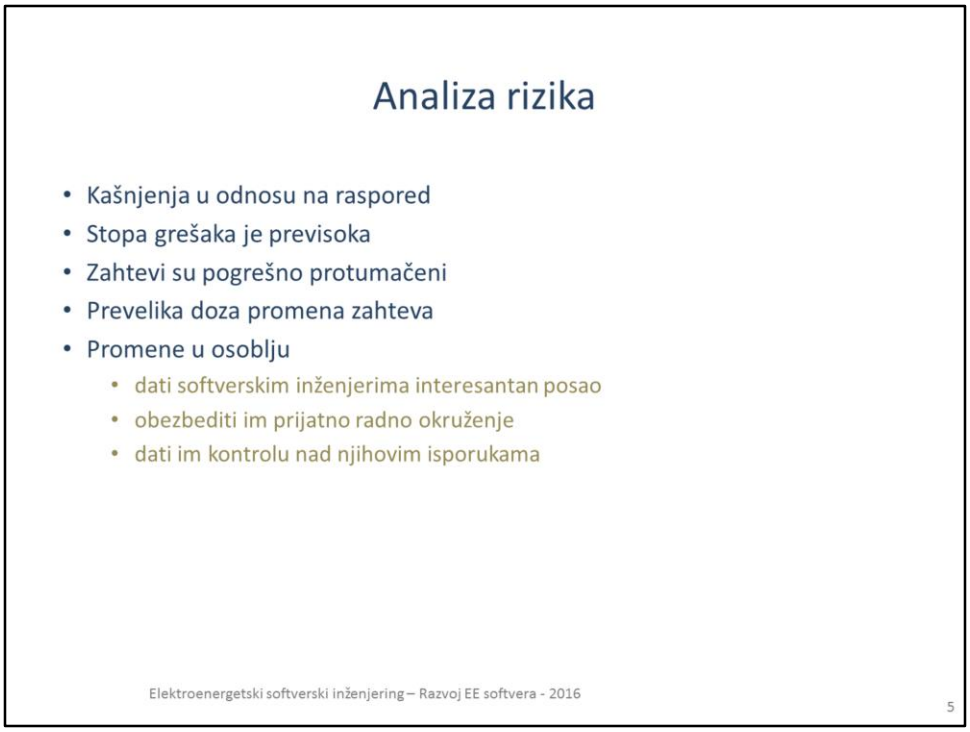
Planiranje projekta

- Uvod i obrazloženje projekta
- Organizacija projekta
- Analiza rizika
- Zahtevi za hardverom, softverom i ljudskim resursima
- Razlaganje posla na delove i estimacija zadataka (tasks)
- Raspored projekta (project schedule)
- Monitoring projekta i mehanizmi izveštavanja, kolektvno poznati kao *nadgledanje projekta*

Projektni plan je u suštini ugovor, doduše kontinualno promenljiv, šta razvojni tim želi postići i način na koji to želi postići. Ono takođe kazuje kako će se upravljati projektom, a u ekstremnim plansko-rukovođenim projektima, čak i kada odnosno kako će se kreirati/modifikovati dokumenti (uključujući i sam plan).

Organizacija projekta

- Kako da se organizuju tim(ovi)?
- Koji procesni model će projekat koristiti?
- Kako će se projekat upravljati po dnevnoj osnovi?



Najbolji način kontrole zahteva (uspešna validacija), jeste, da se klijent integriše u sam razvojni proces (tj., da bude deo tima) i da se što češće klijentima isporuče izvršne verzije programa (nakon svake iteracije). Rukovanje zahtevima je jedno od najbitnijih zadataka projekt menadžera. To je način da se kontroliše nivo promene zahteva.

Tehnike rukovanja rizicima

- Izbegavanje rizika
  - ubacivanje dodatnog vremena u svoj raspored
  - konstantno vršenje pregleda (review) kôda
  - rano zamrzavanje zahteva (barem što većeg dela)
  - česte instalacije
  - zahtevanje programiranja u paru da bi se znanje o proizvodu što više raširilo između članova tima
  - ugradnja kvaliteta kao preventiva (quality assurance)
- Ublažavanje rizika
  - uklananje funkcija iz proizvoda
  - stavljanje prioriteta na ispravljanje grešaka umesto razvoja novih funkcija
  - pregovaranje u vezi novih funkcija u budućim verzijama

Ako projekat koristi iterativni procesni model, dobra praksa je ponovno razmotriti rizike nakon svake iteracije i pogledati koji su se promenili, videti da li ima novih, i ukloniti nepostojeće.

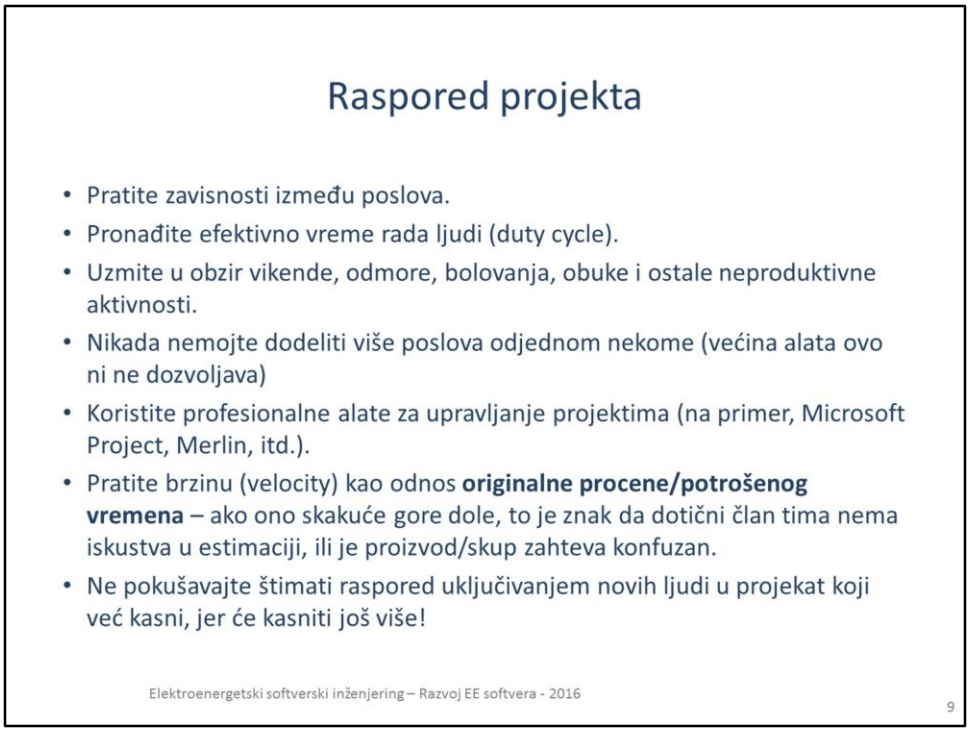
Zahtevi za resursima

- Koliko ljudi je potrebno za projekat?
- Da li oni svi startuju odjednom, ili datumi mogi biti rastegnuti u skladu sa fazama projekta?
- Koliko računara (servera) je potrebno?
- Koji softver(i) će se koristiti za razvoj?
- Koje razvojno okruženje je potrebno?
- Da li su svi obučeni za to okruženje?
- Koje nivoe softversko/hardverske podrške treba obezbediti timovima?

Pitanja koja se ovde nameću su u vezi sa analizom rizika. Na primer, ako tim nije iskusan sa ciljnim okruženjem onda je to dodatni rizik. Jedan način njegovog ublažavanja je formalna obuka.

Razbijanje posla na manje delove i estimacija pojedinačnih zadataka

- Nikada nemojte estimirati napor i predložiti raspored pre nego što provedete neko vreme na dizajnu.
- Najtačnije estimacije se mogu izvesti za poslove koji nisu duži od nedelju dana. Sve iznad ovoga je puko pogađanje (*wild-assed guess*).
- Složeni poslovi treba da se razlože na zadatke (dužine trajanja oko 8 sati).
- Uvek estimirajte veličinu pre napora i cene. Raspored treba da dođe tek na kraju.
- Estimaciju treba da urade softverski inženjeri zaduženi za posao! Jedna od efikasnih metoda estimacije je Delphi metoda.



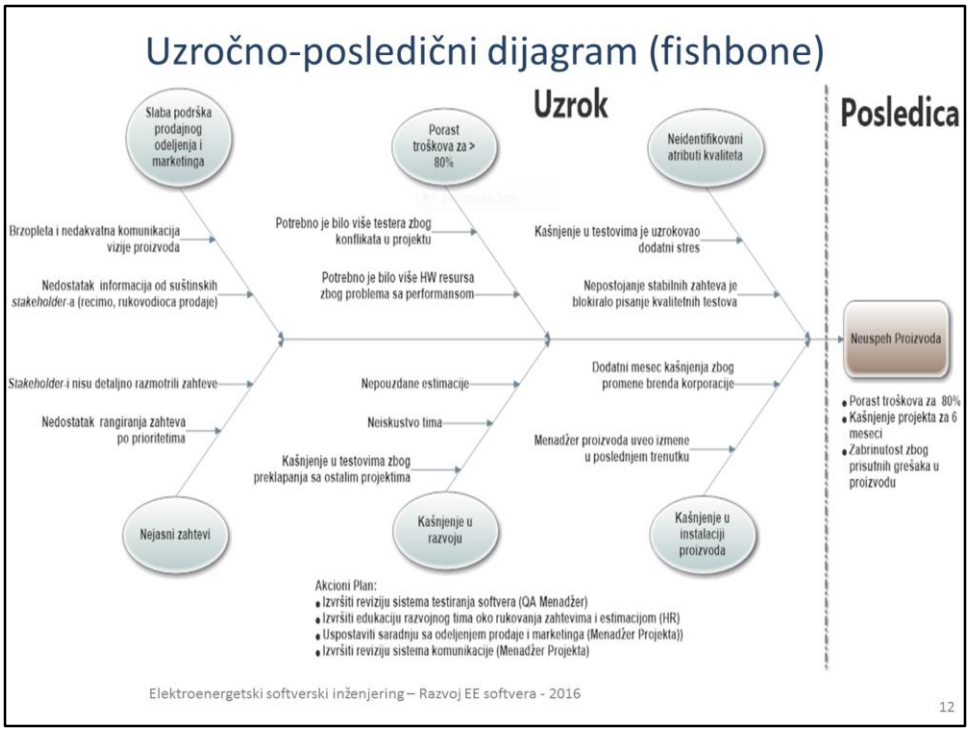
Kod brzine (velocity) je idealna situacija 1.0 (sve se odvija kao što je estimirano). Ovo uglavnom nije slučaj čak i kod eksperata (doduše vrednost se kod njih kreće oko 1.0). Kad se ljudi primoravaju da rade u režimu vremenske oskudice (underestimation), tada se penal za netačnu estimaciju eksponencijalno povećava, dok u drugom slučaju (overestimation) je efekat linearan zbog Parkinsonovog zakona.

Defekti (greške)

1. Fatalna
2. Veoma ozbiljna (Major)
3. Ozbiljna (Minor)
4. Uznemiravajuća/kozmetička (Cosmetic)
5. Promena zahteva (novi zahtev ili izmena postojećeg)

Post-analiza projekta

- Šta je išlo valjano? Da li su se naši procesi ponašali u skladu sa očekivanjima? Da li smo se držali rasporeda? Da li smo isporučili sve tražene funkcije klijentu?
- Šta je krenulo naopako? Zašto smo imali toliko grešaka? Zašto smo često radili prekovremeno?
- Koji procesni problemi su se pojavili? Da li smo dosledno sledili proces? Ako nismo, koji delovi su bili problematični?
- Šta treba da popravimo za sledeći put?
- Ko je odgovoran za popravke/poboljšanja?



Dijagram je generisan upotrebom alata *SmartDraw* (www.smartdraw.com) na osnovu gotovog šablona *Product Failure*.

Naredna lista opisuje glavne korake metode za kreiranje ovakvog dijagrama:

1. Postići saglasnost oko *posledice*, zapisati je na dijagramu, uokviriti je i povući horizontalnu strelicu ka njoj. Podeliti dijagram na dva dela vertikalnom linijom koja neposredno prolazi sa leve strane *posledice*. Označiti levi segment sa "Uzroci", a desni sa "Posledica".
2. Odrediti glavne kategorije *uzroka* (ovo se može uraditi i putem *brainstorming*-a). Poželjno je imati već podrazumevajuću istu ovakvih kategorija (vremenom se ove kategorije po domenu prilično dobro stabilizuju, i ne treba ponavljati ovaj korak). Na primer, "Nedostatak povratnih informacija", "Nedostatak smernica (*guidelines*)", "Zanemarivanje smernica", "Loša dokumentacija", "Zaboravljene stavke", itd.
3. Zapisati na dijagramu svaku glavnu kategoriju *uzroka*, i povući liniju od centralne strele do date kategorije.
4. Enumerisati sve moguće uzroke, grupisati ih po prethodno utvrđenim kategorijama (jedan uzrok može biti svrstan u više kategorija, ako su konteksti

različiti), i povlačiti grane sa linija tih kategorija.

5. Ponavljati rekurzivno korake 4-5, i time raščlanjivati pojedine uzroke na više poduzroka. Pri tome sve vreme povlačiti i adekvatne linije grana.
6. Nakon što je lista uzroka iscrpljena, treba se fokusirati na oblasti dijagrama koje su nepopunjene. Po potrebi, ponoviti prethodna dva koraka.