



SE325 - UPRAVLJANJE PROJEKTIMA RAZVOJA SOFTVERA

Uvod- Projekti razvoja softvera

Lekcija 01

PRIRUČNIK ZA STUDENTE

SE325 - UPRAVLJANJE PROJEKTIMA RAZVOJA SOFTVERA

Lekcija 01

UVOD- PROJEKTI RAZVOJA SOTVERA

- ▼ Uvod- Projekti razvoja sotvera
- ▼ Poglavlje 1: Uvod u predmet
- ▼ Poglavlje 2: Opšte o Menadžmentu
- ▼ Poglavlje 3: Opšte o upravljanju projektima
- ▼ Poglavlje 4: Složenost projekta, izvodljivost i upravljanje
- ▼ Poglavlje 5: Specifičnosti projekta razvoja softvera
- ▼ Poglavlje 6: Vežba – Kako početi PM karijeru u industriji
- ▼ 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

Uspešnost rada tima zavisi od organizacije – Menadžmenta tima

Većina kurseva iz softverskog inženjerstva se bavi tehničkim aspektima kako da se korektno i profesionalno urade poslovi u okviru projekta razvoja softvera.

Na pr. kako efikasno izvršiti testiranje softvera. Jasno je da važno znati korektno istestirati sofver i potvrditi da odgovara specifikacijama, barem koliko je važno i znati napraviti softver. No, s druge strane, ako nemate dovoljno vremena da istestirate sofver, ne vredi vam puno poznavanje tehnike. Softver će biti ili isporučen netestiran (što se u praksi i radi, ali zato se i dešavaju havarije nuklearnih centrala, udesi aviona i drugih složenih sistema), ili će biti isporučen sa zakašnjenjem (što nekad ili nije mogućno, ili iako jeste, po pravilu se trpe poslovni gubici).

Očigledno, onaj ko donosi odluke (vođa tima ili vođa projekta ili izvršni direktor firme) može puno uticati na uspešnost posla ili poduhvata, bilo u pozitivnom, bilo u negativnom smislu.

Takođe, vođa ima najveći uticaj na uspešnost datog projekta od svih ostalih učesnika projekta. U ovom kursu ćemo se pozabaviti pitanjima kako da se što bolje organizuje rad na razvoju softverskog proizvoda tj. što bolje iskoriste raspoloživa znanja, ljudi, oprema i vreme.

Materija koje će biti izučavana u ovom kursu je odabrana na osnovu referenci [1] i [2], tj. na bazi preporuka SWEBOK 2004 koja opisuje oblasti znanja koje treba da vrlada softverski inženjer, kao i PMBOK koja opisuje oblasti znanja koje treba da vrlada rukovodilac projekta (project manager).

No, takođe možemo reći da je PMBOK osnova za upravljanje bilo kog projekta, dok je SWEBOK orijentisan na ono što je specifično za softverske projekte, i treba da ga znaju i softverski inženjeri (jer često postaju rukovodioci timova) i rukovodioci softverskih projekata.

Početna predavanja ovog kursa će biti posvećena onom što se primenjuje u upravljanju svih pa i softverskih projekata. Dakle, u prvom delu ćemo se više orijentisati na PMBOK.

Drugi deo kursa će izučavati ono što se primenjuje prevashodno u upravljanju softverskim projektima i kao osnova se koristi SWBOK.

Tokom ovog predavanja će biti reči o opštim pojmovima od kojih se veliki broj koristi u svakodnevnom životu i van stručnog konteksta.

Probaćemo da podsetimo čitaoca toga što verovatno zna (ili bar ima neku predstavu o tome), a što će biti baza za uvođenje u profesionalnu terminologiju, pojmove i tehnike.

✓ Poglavlje 1

Uvod u predmet

PRAVILA VEZANA ZA PREDMET

U ovom delu opisana su pravila vezana za predmet, kao i način polaganja ispita.

Student se ocenjuje u toku celog semestra. Ocenuju se njegovi domaći zadaci, rad na projektu, testovi i aktivnost u nastavi. Na kraju u ispitnom roku, ocenjuje se i pismeni ispit. Ocene se daju u poenima.

Maksimalni broj poena je 100 (uključujući i pismeni ispit). Na pismenom ispitu student može dobiti do 30 poena, a aktivnosti u toku semestra (predispitne obaveze) mogu mu doneti do 70 poena, po sledećoj strukturi:

5 domaćih zadataka, svaki donosi po 3 poena (ukupno 15 poena),

2 Projektna zadataka, prvi donosi 25 poena, drugi 20 poena,

Zalaganje na nastavi, do 10 poena.

Forum

Cilj foruma je da podstiče proaktivni rad studenata i njihovu saradnju tokom nastave na predmetu. Svaki forum omogućava da student postavi problem, vezan za lekciju, i da potraži pomoć svojih kolega. Bilo da se problem odnosi na deo predavanja, bilo na deo vežbi. Za postavljanje problema na forum, studentu se priznaje aktivnost na forumu. Isto se priznaje aktivnost na forumu i studentu koji prvi reaguje na ovaj poziv studenta za pomoć, pod uslovom da je njegov savet uspešno rešio problem koji je student postavio.

Domaći zadaci

Posle izučavanja određene nastavne jedinice, odnosno predviđeno je da studenti dobiju zadatak koji treba samostalno da reše.

Svaki student dobija različit zadatak i kao preduslov za izlazak na ispit u obavezi je da uradi i pošalje sve zadatke.

Predviđeno je ukupno 5 zadataka, a svaki uspešno rešen zadatak predat u zadatom roku obezbeđuje studentu 3 poena, pod uslovom uspešne odbrane urađenog zadatka.

Za studente tradicionalne (u Beogradu) ili hibridne nastave (Centar u Nišu) rok za predaju zadataka je **7 dana nakon izdavanja**, a posle tog roka umanjuje ostvaren broj poena za 50%.

Krajnji rok za predaju domaćeg zadatka i za studente tradicionalne nastave i za studente onlajn nastave je **10 (deset) dana** pre ispitnog roka u kome student polaže ispit.

Studenti online nastave brane domaće zadatke u dogovoru sa predmetnim asistentom ili nastavnikom.

ZALAGANJE

Student dobija poene tokom semestra na predispitnim obavezama.

Aktivnost u nastavi (određivanje poena za zlaganje) se delom ocenjuje drugačije za studente tradicionalne/hibridne nastave (centri u Beogradu i u Nišu), i za online studente.

Kod studenata tradicionalnog i hibridnog oblika nastave (centri u Beogradu i u Nišu) primenjuju se sledeći kriterijumi:

- Redovnost u pohađanju nastave. Student tradicionalne ili hibridne nastave koji, iz bilo kog razloga, nije pohađao nastavu na više od 30% časova predavanja i vežbanja (pet izostanaka), dobija automatski 0 poena za zlaganje.
- Dolazi pripremljen za nastavu.
- Redovnost i kvalitet ispunjenja predispitnih obaveza. **Student koji ne uradi sve svoje predispitne obaveze najkasnije 15 dana po završetku nastave na predmetu, dobija nula (0) poena za zlaganje.**

Kod **online studenata** (studije preko Interneta)

- Ranije (u odnosu na ispit) ispunjenje predispitnih obaveza
- Konsultacije tokom semestra u vezi rada na predispitnim obavezama.

PROJEKTI

Projekti su najvažnija aktivnost studenta tokom semestra.

Zahtevan rok za predaju Izveštaja o urađenom projektnim zadacima je do kraja 14. nedelje nastave za studente tradicionalne studente nastave (Beograd i Niš), a za studente online nastave najkasnije 10 dana pre ispita. **Studentima tradicionalne i hibridne nastave koji predaju Izveštaj o urađenim projektnim zadacima 15 i više dana po završenoj nastavi na predmetu, umanjuje se broj ostvarenih poena za 30%.**

Obrana projekta za studente internet nastave obavlja se online, a ako je za studenta prihvatljivo, u prostorijama Univerziteta.

Krajnji rok za predaju projekta, za sve studente, je 10 dana pre ispitnog roka u kome žele da polažu ispit. Studentima online nastave se ne umanjuje broj poena, jer su oni, po pravilu zaposleni, ili sprečeni da redovno studiraju.

Ako student prilikom ocenjivanja jednog od projekta ne dobije najmanje 50% predviđenih poena, mora da doradi projekat ili projekte. U suprotnom, dobija 0 poena. **Student koji ne dobije više od 50% predviđenih poena ne može izaći na ispit.** Student je dužan da projekte obrani kod asistenta ili nastavnika.

Odbрана пројектата студената традиционалне наставе и хибридне наставе врши се у 15. недељи јесенjег семестра за време вежби, а ако је потребно, и ван вежби – на додатним часовима. Ван овог термина, студенат може да брани пројекат у посебном термину који одреди асистент пред сваки испитни рок.

Циљ одbrane пројекта је да асистент установи да ли је студенат самостално радио пројекат и у том циљу одбрана пројекта подразумева да студенат демонстрира зnanje које је искористио за израду пројекта.

Први пројектни задатак представља **Plan manjeg projekta u nekoliko koraka**. "Кораци" су реšења домаћих задатака (укупно 5).

Други пројектни задатак представља следећих 6 докумената, за који студенти добијају шаблоне:

- Postupak razvoja softvera (SDLC),
- Alati za kolaboraciju na projektu,
- Plan upravljanja konfiguracijom,
- Plan upravljanja ризиком,
- Начерт захтева за softver (SRS),
- Plan testiranja и dokumentacija o testiranju.

✓ Poglavlje 2

Opšte o Menadžmentu

ŠTA JE TO MENADŽMENT

Pod menadžmentom se podrazumeva organizovanje, rukovođenje ili upravljanje (nečim)

Engleski termin **Management** označava ono za šta se kod nas koriste termini **organizovanje, rukovođenje ili upravljanje**.

Project Management označava ono što prevodimo kao rukovođenje projektom ili upravljanje projektom.

Menadžment se primenjuje u raznim oblastima ljudske aktivnosti sa ciljem da te aktivnosti rezultuju u očekivane ili željene ishode.

Otuda, može se reći i da je *menadžment primena znanja, veština i sredstava sa ciljem da data ljudska aktivnost rezultuju u očekivane ili željene ishode*.

Razmotrimo sada šta je to menadžment projekta



Slika 2.1 Management prevodimo kao organizovanje, upravljanje ili rukovođenje. Izvor: Autor.

ŠTA JE TO MENADŽMENT PROJEKTA

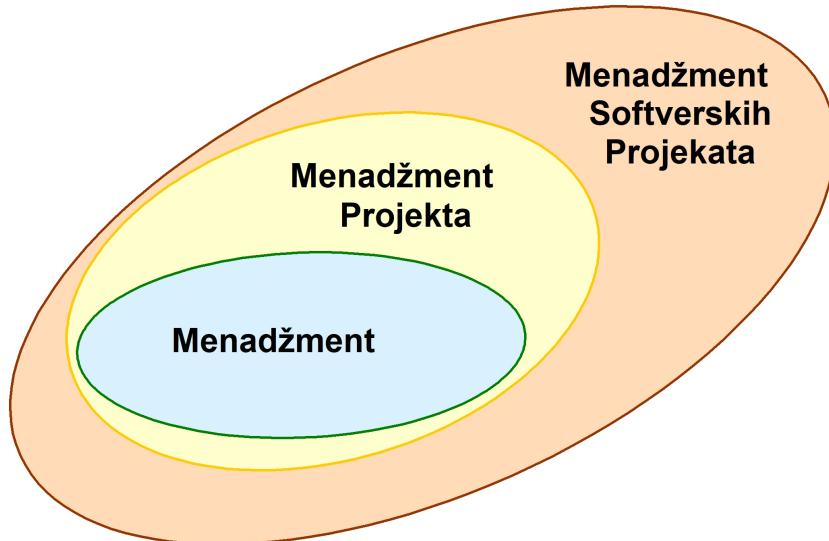
Menadžment projekta je primena menadžmenta na projekte

Upravljanje projektima (eng. **Project Management**) je primena menadžmenta na projekte – privremene poduhvate sa ciljem da se stvori jedinstveni proizvod ili usluga.

Može se reći da je projekat skup aktivnosti koje se izvode da bi se postigli postavljeni ciljevi u okviru planiranog vremena i raspoloživog budžeta.

Projekti razvoja softvera su primeri projekata koji imaju navedena svojstva ali i dodatne specifičnosti koje drugi projekti ne moraju da imaju.

Tokom izučavanja ovog predmeta, fokus će biti na upravljanju ili vođenju projekata razvoja softvera.



Slika 2.2 Menadžment se primenjuje u raznim oblastima. Izvor: Autor.

No, pre nego što pređemo na specifičnosti vođenja softverskih projekata, pozabavimo se onim što je zajedničko za sve projekte. Poći ćemo od intuitivnog shvatanja projekta, koji svako od nas ima, a zatim ćemo preći na izučavanje materije kojom treba da se vlada u profesionalnom svetu.

✓ Poglavlje 3

Opšte o upravljanju projektima

SVOJSTVA PROJEKATA

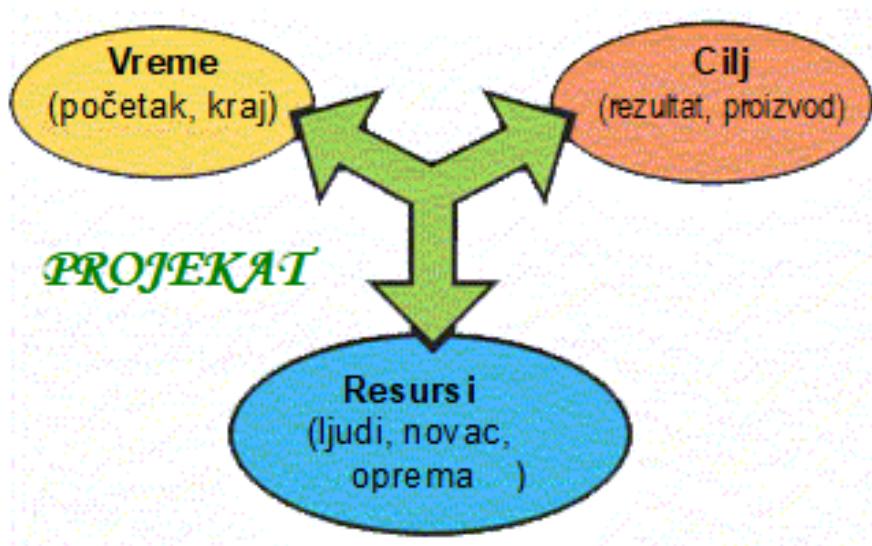
Projekat ima jasno definisani cilj, koji treba da se dostigne za određeno vreme, i uz datu količinu rada i para

Verovatno razvoj softvera nije nešto što vam je potpuno nepoznato. Kao student ste već slušali dosta o ovom procesu. A možda već i imate profesionalnog iskustva u razvoju softvera. Takođe, ni reč projekat ne deluje vam strano, bilo da se govori o projektima razvoja softvera ili o drugim projektima koji nemaju veze sa softverom.

Kao primer, često se govori o projektima novog autoputa, ili projektu gasovoda, ili projektu uvođenja nekog stranog jezika (npr. Kineskog) u nastavu ili projektu odlaska na mesec...

Jasno je da navedeni primeri projekata imaju jasno definisani cilj i da je potrebno određeno **vreme**, količina **rada i para** da bi se postigao postavljeni **cilj**.

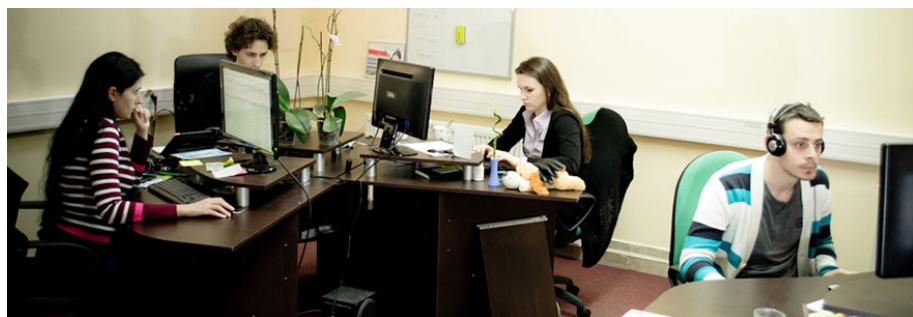
Potrebno ali ne i dovoljno, jer vreme, novac i rad sami za sebe neće dovesti do cilja ukoliko ne postoji odgovarajuća organizacija projekta.



Slika 3.1 Vreme, novac i rad su potrebni ali ne i dovoljni za ostvarenje cilja. Izvor: Autor.

ZNAČAJ ORGANIZACIJE PROJEKTA

Boljom organizacijom se postižu bolji rezultati u projektu



Slika 3.2 Primer dobro organizovanog tima. Izvor: <https://unsplash.com> (stock image)

Boljom organizacijom, ili rukovođenjem (kako još kažemo) se postižu bolji rezultati u projektu.

Obrnuto, ne postoji dovoljno kvalitetan tim, ni dovolno vremena ni para da loše rukovođenje ne može da upropasti projekat.



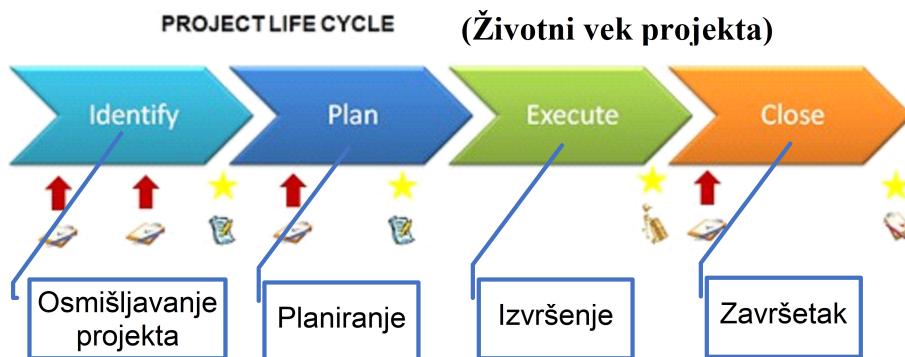
Slika 3.3 Primer slabo organizovanog tima. Izvor: <https://unsplash.com> (stock image)

Otuda se veštini menadžmenta projekta poklanja velika pažnja u svim vidovima ljudske delatnosti, a takođe i u projektima razvoja softvera.

ETAPE PROJEKTA - INICIRANJE I PLANIRANJE

Svaki projekat ima početak i kraj

Svi projekti prolaze kroz nakoliko faza.



Slika 3.4 Tipične etape projekta. Izvor: Autor

1. Faza osmišljavanja (iniciranja)

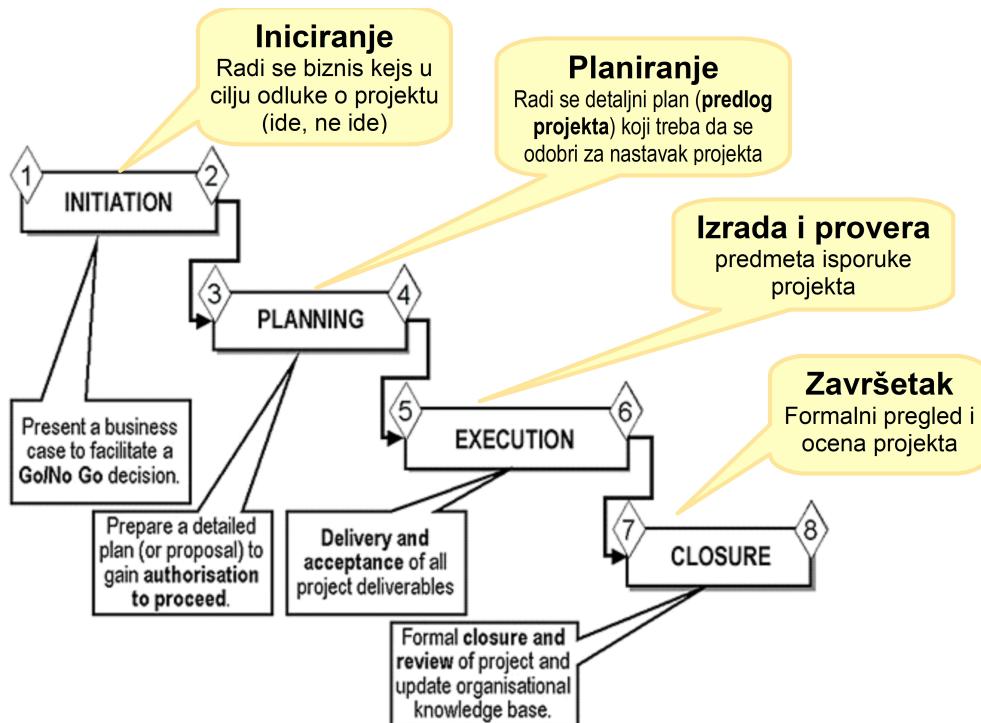
Najpre je potrebno osmisliti projekat, tj. definisati njegov cilj i raspoloživa sredstva i vreme. Zatim se vrši analiza izvodljivosti i ako je njen zaključak da je projekat izvodljiv, nastavlja se.

2. Faza planiranja.

Radi se plan koji poslovi i kojim redom će se obavljati, kada će početi i koliko će trajati, kako će se trošiti sredstva i kako će se proveriti da li je rezultat projekta dobar.

ETAPE PROJEKTA - IZVRŠENJE I ZAVRŠETAK

Planiranje i izvršenje projekta su dve osnovne etape.



Slika 3.5 Opis osnovnih etapa projekta. Izvor: Autor.

3. Faza izvršenja.

Najlakša za opis: postupa se u skladu s planom i teži da se ostvari ono što je zamišljeno. U praksi, najviše rada je uloženo u ovoj fazi.

4. Faza završetka.

Proverava se da li su svi poslovi uspešno završeni, izmiruju se sva dugovanja i rezultat projekta (ostvareni cilj) počinje da se koristi.

Medjutim, intuitivno je jasno da su neki projekti lakši za realizaciju, neki teži, što već zavisi od puno faktora.

✓ Poglavlje 4

Složenost projekta, izvodljivost i upravljanje

PRIMER: JAKO SLOŽENI PROJEKTI

Neki projekti se lakše organizuju i izvedu, neki teže.

Da li vas je neko pitao umete li da napravite avion? Verovatno nije, ali i da jeste, zamislite da ste dobili zadatak da napravite avion.

Pre nego što odgovorite, proanalizirajmo malo zadatak

a) Avion je vozilo koje služi za prevoz putnika, kao na pr. Cessna Citation CJ2 na slici desno. Da bi se takav avion napravio, potrebno je puno kvalifikovanih ljudi (piloti, inženjeri, tehničari, majstori), puno specijalizovane opreme i puno, puno para.

b) No avioni koji služe za zabavu vlasnicima (tzv. Ultra-laki) se mogu kupiti u kitu po ceni jednog putničkog automobila.

Ovaj avion možete registrovati i leteti poštujući pravila avio-saobraćaja. Jedino, treba da ga sastavite sami, ili uz pomoć nekog aviomehaničara.

Verovatno bi ste odgovorili da možda možete da realizujete ovakav projekat (ako bi ste imali dovoljno para, vremena i želje da ga uradite).



Slika 4.1 sl1 Biznis đžet Cessna Citation CJ2. Izvor: <https://unsplash.com> (stock image)



Slika 4.2 Ultralaki avion. Izvor: <https://unsplash.com> (stock image)

PRIMER: MANJE SLOŽENI PROJEKTI

Avion od papira možete napraviti i sami

c) Zamislite da vas je neko dete zamolilo da mu napravite avion.

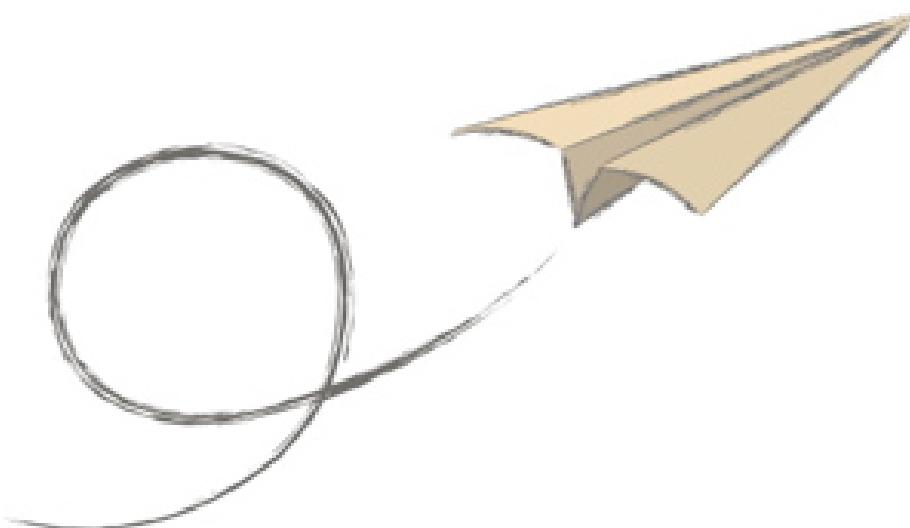
Ne bi vam bila potrebna nikakva analiza izvodljivosti (en. **feasibility study**) da bi ste doneli odluku.

Parče papira, nekoliko minuta posla i avion možete dati detetu da se igra.

Lako zar ne? No nemojte brzati s odgovorom. Postoje takmičenja aviona od papira i to internacionalna i nije nimalo lako napraviti avion od papira sa vrhunskim letnim karakteristikama.

Ako više volite primere iz oblasti softverskog inženjerstva, nama problema.

Zamislite da ste dobili zadatak da razvijete softver kojim će se projektovati avion. Uzmite avione pod a), b) i c), uradite analizu izvodljivosti softvera za njihovo projektovanje i dobićete slične rezultate.



Slika 4.3 Papirni avion. Izvor: <https://unsplash.com> (stock image)

SLOŽENOST I VELIČINA PROJEKTA

Što je projekat složeniji teže ga je uraditi uspešno

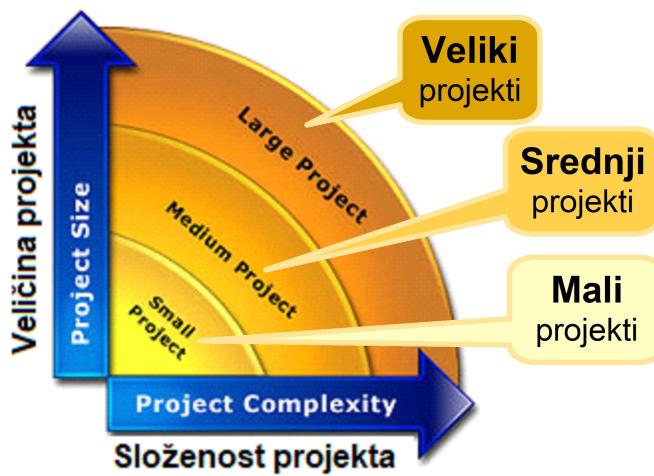
Za nas je bitno da uočimo jednu bitnu karakteristiku projekta (ili zadatka koji treba uraditi) a to je složenost projekta. Što je projekat složeniji teže ga je uraditi uspešno.

No kako složenost nije lako objektivizirati (za pilota je letenje prosta stvar, ali je programiranje složeno), to je pogodnije koristiti posledice te složenosti. Naime, ako je projekat složeniji, biće potrebno više rada (I intelektualnog I manualnog), veći broj učesnika, više vremena i para da bi se uspešno završio projekat.

Time smo de fakto uveli veličinu projekta kao pokazatelj koji je u korelaciji sa složenošću. Kako je veličina projekta neposredno merljiva lakše je da je koristimo umesto složenost.

Veličina projekta se obično izražava preko sledećih pokazatelja:

- Ukupna sredstva za projekat (budžet)
- Broj učesnika projekta
- Složenost proizvoda (**deliverable**)
- Broj i cena proizvoda
- Vreme raspoloživo za projekat



Slika 4.4 Podela projekata po veličini i složenosti. Izvor: Autor.

Projekti bi se mogli podeliti na male (en. **small-scale projects**) i velike (en. **large-scale projects**) već prema tome šta se smatra pod "malim" i "velikim" i kako se određuje veličina projekta.

U ovom kursu ćemo uvesti podelu projekata na vrlo male, male i velike da bi smo ilustrovali tehnike menadžmenta (upravljanja) projektima.

VRLO MALI PROJEKTI

U vrlo malim projektima ima malo (ili nimalo) dokumentacije

Vrlo mali ili individualni projekti bi bili oni koje može pojedinac uglavnom sam da završi (eventualno uz nečiju kratkotrajnu pomoć) u vremenu koje sa intuitivnog aspekta nije veliko a ni angažovana sredstva (novac i oprema) nisu veliki.

Dakle, mali ili pojedinačni projekti se realizuje uglavnom koristeći raspoloživu opremu (bez dodatnih nabavki i troškova) i bez pomoći saradnika.

Upravljanje ovim projektima je najjednostavnije, jer nema puno elemenata na projektu i ne treba voditi pismenu evidenciju, dokumentacija je minimalna (ili je nema), nema (puno) saradnika pa ni potrebe za razmenom informacija.

No, s druge strane, često se pada u zamku smatrajući da se kod vrlo malih projekata može bez planiranja, procena rizika i uspešnosti i drugog a što se po pravilu radi pri upravljanju projektima.



Slika 4.5 Kod vrlo malih projekata nema potrebe za složenom metodologijom upravljanja. Izvor: <https://unsplash.com> (stock image)

MALI PROJEKTI

Tipično, tim za realizaciju je do pet članova a trajanje do nekoliko nedelja

Mali projekti se realizuje u timu koji ima bar dva člana (ali obično ne preko 5), a angažovana sredstva i utrošeno vreme mogu biti značajni. Pored projektnog tima koji realizuje projekt, pojavljuju se i drugi učesnici koji imaju ulogu u odvijanju projekta:

- **Finansijer** (Vlasnik ili investitor) projekta. Njegova uloga je da, sem što obezbeđuje i odobrava sredstva za izvršenje projekta, vrši obezbedi da se projekat fokusira na postizanje cilja u koji su sredstva uložena.
- **Vođa projekta** (en. **project manager**) koji je odgovoran da isporuči rezultat projekta (en. **deliverable**) investitoru. On vrši osmišljavanje projekta i predlaganje investitoru, a nakon odobravanja projekta vrši njegovo planiranje i rukovodi realizacijom, sve do završetka. Veoma često, vođe projekta i sam radi na realizaciji poslova (a ne samo na rukovođenju).
- **Zainteresovani za projekt** (en. **Stakeholders**) su pojedinci ili grupe ljudi koji imaju interesa za rezultate projekta. Dakle, nabrojani učesnici investitor, vođa projekta i članovi tima za razvoj su stejkholderi, ali nisu jedini.

Iako broj članova projektnog tima nije veliki, pojava i drugih učesnika na projektu onemogućava da se projekat vodi "iz glave". Potrebna je i pisana dokumentacija (ugovori, zapisnici, računi itd.), potrebna je komunikacija sa svim učesnicima u cilju izmene informacija, koordinacije i kontrole izvršavanja poslova i praćenja kvaliteta.

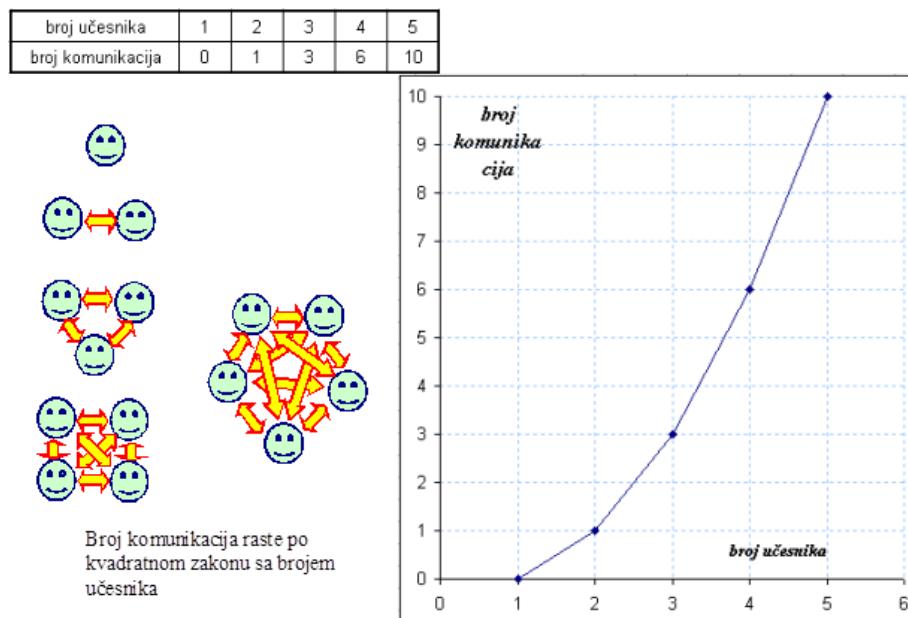
Za razliku od individualnih projekata, vođenje malih projekata zahteva i uspostavljanje procedura upravljanja projektom kao i poznavanje raznih tehnika menadžmenta koje se uče

u školama ili na raznim kursevima. No vođa projekta se ne bavi isključivo rukovođenjem jer tim za realizaciju projekta nije dovoljno veliki.

KOMUNIKACIJA U MALIM PROJEKTIMA

Broj komunikacija raste po kvadratnom zakonu sa brojem učesnika

Ako pogledamo kako raste broj međusobnih komunikacija sa porastom broja učesnika, jasno je zašto se kod individualnih projekata postižu uspešne realizacije iako nema puno primena profesionalnih tehniki menadžmenta projekta. Nema (ili skoro nema) potreba za dogovaranjem i izmenom informacija, pa je samim tim i organizacija efikasnija.



Slika 4.6 Što je više učesnika to je teža komunikacija. Izvor: Autor.

U malom projektu, tačnije u malom timu vođe projekta mora da uspostavi organizaciju i način izmene informacija tako da se projekat efikasno odvija. Ono što dodatno može komplikovati menadžment je mogućnost da tim ne bude na istoj lokaciji, a što je veoma karakteristično za projekte razvoja softvera.



Slika 4.7 Komunikacija je otežana u distribuiranom timu. Izvor: <https://unsplash.com> (stock image)

Razmena informacija i podataka preko interneta mora kompenzirati nedostatak neposrednog kontakta i biti jako dobro organizovana da se ne dođe u zasićenje tj. izbegne prenatrpanost informacijama koja bi apsorbovala veliko vreme učesnika na projektu, tj. smanjila raspoloživo vreme za poslove na projektu.

VELIKI PROJEKTI

Veliki projekti mogu trajati godinama i imati mnogo učesnika

Veliki projekti se realizuje u timu koji ima preko 5 članova ali ih može biti i mnogo, mnogo više.

Kao primer posebno velikih projekata (neki ih zovu i **mega-projekti**) možemo navesti izgradnju egiptskih piramida, kineskog zida ili, u novije vreme, Apolo program - slanje prvog čoveka na mesec.

Angažovana sredstva i utrošeno vreme mogu biti veliki ili čak veoma veliki.



Slika 4.8 Primeri velikih (Mega) projekata. Izvor: <https://unsplash.com> (stock image)

Napomena

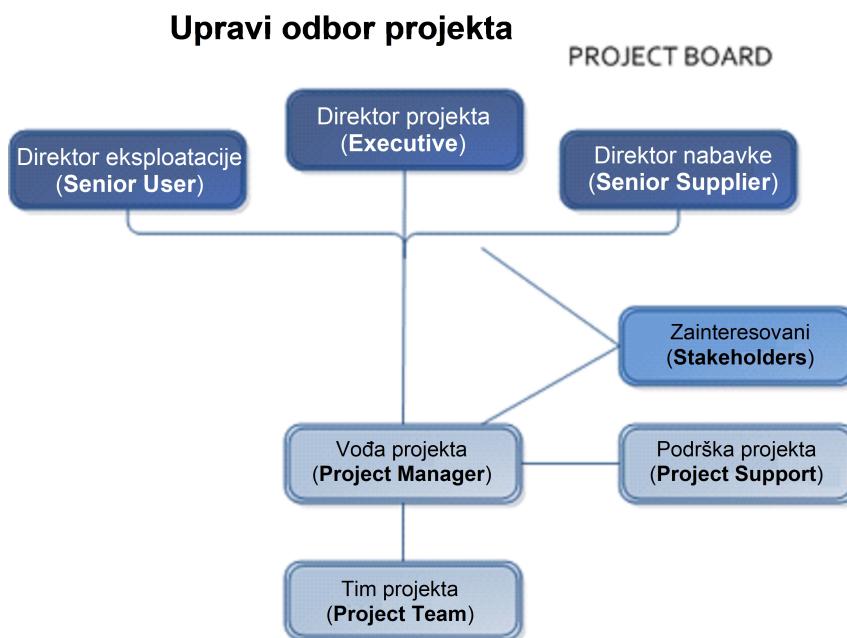
U ovom kursu ćemo ostaviti po strani veoma velike projekte i pozabaviti se onima koji su na donjoj granici velikih (recimo tim za razvoj od 5 do 50 članova, vreme do 1 godine i uložena sredstva 2 puta veća od iznosa plata angažovanih ljudi na projektu).

UPRAVLJANJE SREDNJIM PROJEKTOM

Odbor projekta (Project Board) upravlja projektom

Upravljanje projektom vrši odbor projekta (en. **Project Board**) koji donosi odluke o toku projekta. U odboru su članovi koji zastupaju tri interesa:

- **Izvršni direktor** (en. **Executive**) Njegova uloga je da, sem što zastupa investitora tj. obezbeđuje i odobrava sredstva za izvršenje projekta, vrši obezbedi da se projekat fokusira na postizanje cilja u koji su sredstva uložena. Odgovoran je da se projekat uspešno završi.
- **Interesi korisnika** (en. **Senior User**) Njegova uloga je da zastupa interes korisnika u proizvodu koji će biti isporučen
- **Nabavka** (en. **Senior Supplier**) Njegova uloga je da se bavi podugovaračima koji se angažuju da urade i isporuče ono što je potrebno za realizaciju projekta.
- **Menadžer projekta** (en. **Project Manager**) se bavi operativnim pitanjima realizacije projekta, i obezbeđuje ono što je potrebno za administraciju projekta. Njegovi zadaci su praćenje izvršavanja planova projekta i upravljanje izradom projektne dokumentacije, kao i izveštavanje odbora za upravljanje projektom.

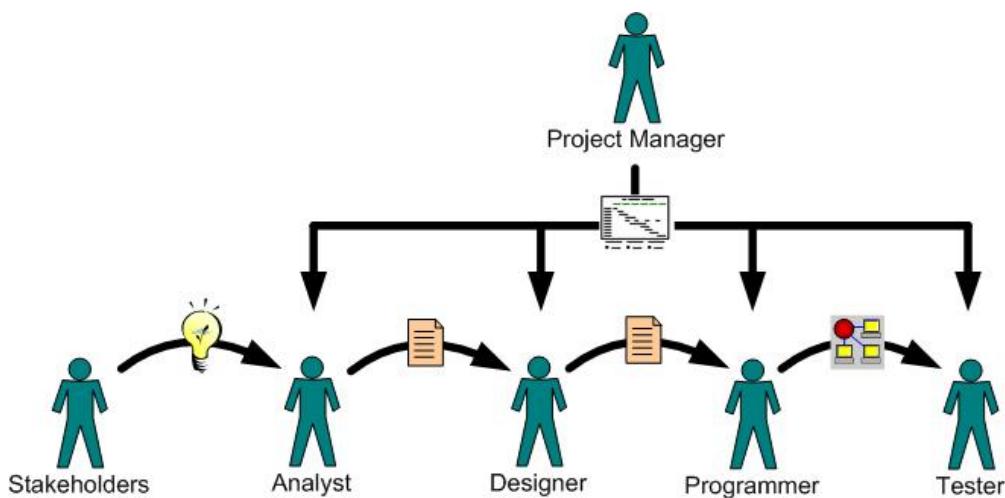


Slika 4.9 Struktura odbora za upravljanje projektom. Izvor: Autor.

ORGANIZACIJA TIMA ZA REALIZACIJU SREDNJEG PROJEKTA

Tim za realizaciju projekta je struktuiran prema tehnološkoj specijalizaciji

Projektni tim realizuje projekat, no za razliku od malih projekata može biti struktuiran prema tehnološkoj specijalizaciji (vidi sledeću sliku).



Slika 4.10 Struktura tima za realizaciju projekta. Izvor: <https://unsplash.com> (stock image)

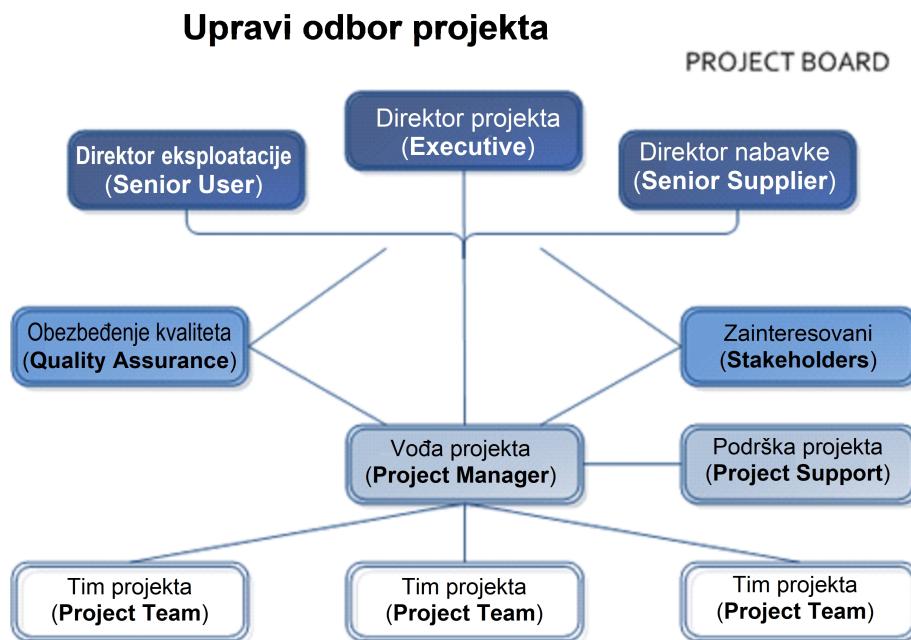
Zainteresovani za projekt (en. **Stakeholders**) su kao i u prethodnom slučaju pojedinci ili grupe ljudi koji imaju interesa za rezultate projekta. Dakle, učesnici investitor, vođa projekta (projekt menadžer) i članovi tima za razvoj, ali se mogu biti i korisnici, predstavnici socijalnih i političkih grupa, već zavisno od prirode projekta.

Podrška projekta mogu biti ljudi privremeno angažovani na projektu, na pr. specijalisti u oblastima kojih nema u timu (jer ne postoji stalna potreba za njihovim angažovanjem - prevodioci, piloti, lekari itd.,)

UPRAVLJANJE VELIKIM PROJEKTOM

Upravni odbor velikog projekta može biti složen.

Kod malo većih i velikih projekata (a kasnije ćemo videti da to zavisi i od organizacije u kojoj se projekt odvija) mogu postojati i dodatne uloge na projektu.



Slika 4.11 Upravni odbor velikog projekta može biti složen. Izvor: Autor.

Obezbeđenje kvaliteta je po pravilu integralni deo projekta (prisutno u svim studijama i preporukama o dobrom menadžmentu projekata). Njegova uloga je da se sprovode postupci i procedure (obično donesene na nivou organizacije) u postizanju i proveri kvaliteta:

- **Kvalitet proizvoda** (treba da odgovara traženom, u protivnom projekat ne može biti uspešan)
- **Kvalitet projekta** (zahtevani kvalitet proizvoda treba da bude ostvaren, ali na što efikasniji način, tj. uz što manje angažovanje resursa i troškove i za što kraće vreme)
- **Kvalitet procesa** (da bi što veći broj projekata organizacije bio što uspešniji, organizacija razrađuju i primenjuju tehnološke postupke i procedure sa ciljem optimizacije realizacije projekata)

Dalje, tim za realizaciju projekta može biti podeljen u manje grupe (setimo se da komunikacija u grupi od 3 do pet članova, pa time i koordinacija aktivnosti, može biti efikasna i bez primene formalnih metoda)

Rukovođenje projektima razvoja softvera ima dosta specifičnosti u odnosu na druge projekte iz prostog razloga što je tehnologija razvoja softvera drugačija od one koja se koristi za druge proizvode, ali i dosta zajedničkih karakteristika.

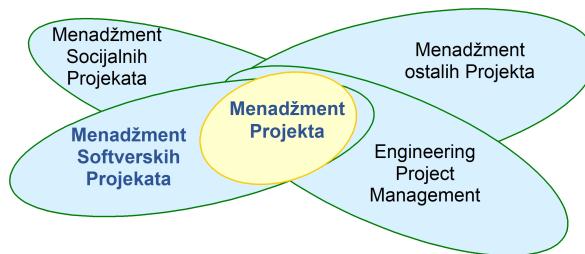
▼ Poglavlje 5

Specifičnosti projekta razvoja softvera

UVODNE NAPOMENE

Realizacija projekata u svakoj oblasti je specifična

Softver se razlikuje od drugih proizvoda, koji su rezultati ne-softverskih projekata, svakako. Više nego što se razlikuju rezultati projekata uvođenja interneta u osnovne škole (i možda obdaništa), od rezultata projekta penjanja na Mont Everest? Hmm... No nije u tome stvar. Projekti se rade u mnogim oblastima, inženjerstvu, industriji, zdravstvu, nauci, kulturi i (da ne zaboravimo nas), u softverskom inženjerstvu.



Slika 5.1 Relacija projekta razvoja softvera i drugih projekata. Izvor: Autor.

Postoji nešto što je zajedničko svim (tj. bar većini) oblasti, što je na slici označeno sa "project management". Postoji i deo tehnika menadžmenta koji je specifičan za razvoj softvera.

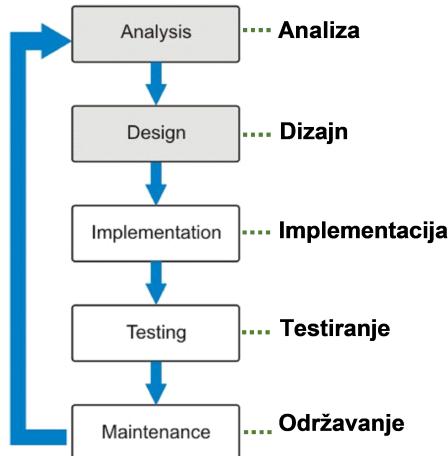
Stoga se u kursevima softverskog inženjerstva češće govori o specifičnim tehnikama menadžmenta i faze razvoja softvera dele na analizu, dizajn, Implementaciju, testiranje i održavanje.

U ovom kursu će biti reči o softverskom inženjerstvu u kontekstu rada na nekom malom projektu (npr. vaše sopstvene računarske igre koju ćete razvijati uz pomoć par kolega ili sami). Dakle, zamislite da želite da napravite vašu igru, kako ćete organizovati projekat?

Softversko inženjerstvo je skup tehnika koje se koriste za proizvodnju kompjuterskih programa a koje su ljudi spremni da kupe. *U jednoj rečenici, cilj softverskog inženjera je: "isporuči softver! "*

Iako slanje proizvoda obično znači stavljanje diska i uputstva u kutiju upakovano u plastičnu foliju, danas isporuka može značiti stavljanje programa da bude dostupan za preuzimanje sa Interneta.

Iako postoji samo jedan cilj, postoji mnogo različitih aspekata u softverskog inženjerstva. Ako idete u veliku kompjutersku knjižaru i pogledajte odeljak o softverskog inženjerstva, naći ćete izuzetan assortiman knjiga.



Slika 5.2 Faze razvoja softvera. Izvor: Autor.

KARAKTERISTIKE USPEŠNOG SOFTVERA

Softver treba da bude dobar i profitabilan

Evo dva razumna kriterijumi za uspešni program.

- Da li donosi pare ?
- Da li je dobar ?

Program koji pišete može vam doneti novac na direktni način, ako je toliko privlačan da su ljudi spremni da plate da ga koristite. Posredno, program vam može doneti novac, ako je dovoljno dobar da ubedi nekoga da vas zaposli na dobro plaćenom mestu.

Takođe je važno za program da bude dobar. Prelepe stvari ne donose uvek novac, ali imaju svoju vrednost. U pokušaju da se napravi program koji je profitabilan i dobar, postoje četiri oblasti koje treba imati na umu.

- Osnovni koncept.
- Interfejs.
- Dokumentacija.
- Stabilnost.

Recimo nekoliko reči o svakom od ovih aspekata razvoja softvera.



Slika 5.3 Ilustracija procesa razvoja softvera. Izvor: Autor.

KONCEPT SOFTVERA

Koncept softvera treba da je usklađen sa potrebama korisnika

Ovo je dosta složeno pitanje. Ako vaš program treba da zaradi novac, treba da postoji neki razlog da ga ljudi žele. Vaš program treba da radi nešto što ljudi vrednuju, i to bolje od konkurenčkih programa. Ako je cilj da vaš program bude dobar, mora biti zasnovan na originalnoj i zanimljivoj ideji - a to nije lako postići.

Ako pogledate vrste programa koje ljudi najviše kupuju, primetićete da oni spadaju u nekoliko glavnih kategorija:

- Alati,
- igre,
- edukacija (obrazovanje) i
- programi sa sadržajem (**content**).

Programi alati (**Tool**) su stvari kao što su word- procesori, unakrsne tabele programa boja, kompjajleri, veb brauzeri, paketi za računarsko projektovanje (CAD – **Computer Aided Design**), i tako dalje. To su veliki programi sa mnogo koda i obično su ih stvorili veliki timovi softverskih inženjera. Programi alati su verovatno najteži da se naprave i najteže da se prodaju. Iz tog razloga jedan od menadžera Autodesk-a ima običaj da kaže, "Apps dobri, alati loši" ('**Apps**

good, tools bad.'). App je skraćenica za **application** (primena), što se pod Windows-om obično misli na izvrši program. I zaista dobra aplikacija se često naziva killer app ("aplikacija ubica"). Igre su programi u rasponu od jednostavnih, tipa pasijans-a za ubijanje vremena do veoma složenih interaktivnih virtualnih realnosti. Internet je otvorio mnogo zanimljivih mogućnosti za kreiranje zajedničkog sveta igre za više učesnika. Kao i sa alatima, ključ uspeha je da se o igrama razmišlja na potpuno nov način. Ali, čak i prilično standardna vrsta igre se može dobro prodati ako se zasniva na originalnom grafičkom konceptu i veštaj implementaciji. Vredi napomenuti da su igre i alati skoro jedine vrste softvera koji su ljudi i dalje spremni da kupuju u kutiji upakovana u plastičnu foliju.

Igra koju igrate na ekranu bežičnog uređaja kao što je mobilni telefon će biti mala, ali komercijalni uspeh konzola i kompjuterskih igara je ogroman. Sem koda, igra će sadržati puno datoteka sa podacima kao što su zvuk, bitmape i dizajn nivoa. Na neke savremeni igre je utrošeno čak 100 čovek- godina rada. Projekti iz softverskog inženjerstva u ovom kursu će se fokusirati na manje igre, složenosti otprilike oko jednog nivoa klasične arkadne igre.

Edukacioni (Obrazovni) programi pokrivaju širok spektar. Najveći deo edukacionog softvera se sastoji od jeftinjih programa tipa "čitaj, ponovi". Proste obrazovne aplikacije mogu uključivati čak manje nego koda nego igre, jer su obično razvijeni u programskom okruženju visokog nivoa, kao što je Microsoft Visual Basic ili Macromedia Flash.

SOFISTICIRANI OBRAZOVNI PROGRAMI

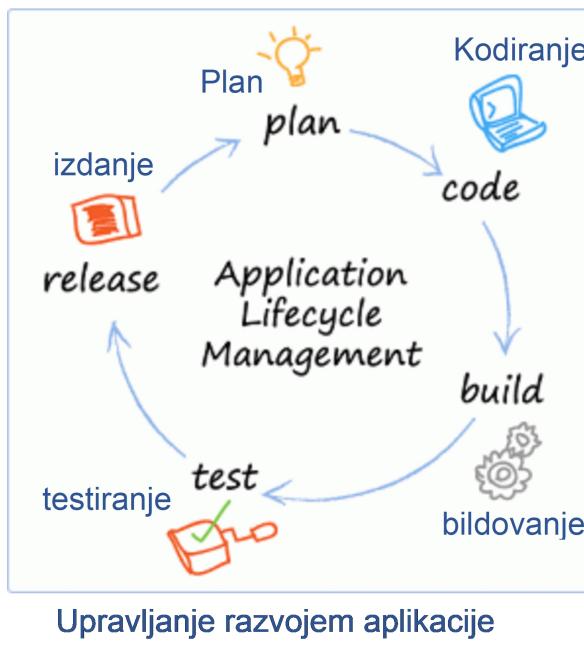
Sofisticirani obrazovni programi često ilustruju naučne koncepte sa računarskim simulacijama

Više sofisticirani obrazovni programi često ilustruju naučne koncepte sa računarskim simulacijama. Simulacije mogu modelirati apstraktne matematičke ideje poput fraktala i teorije haosa, na primer, ili vam obezbediti da učite upravljanje avionom, ili mogu modelirati bioloških sistema koristeći tehnike poznate kao veštački život (en. **artificial life**). Ovo su neki od primera proračuna za koje su računari dobri, a koje bi bilo apsolutno nemoguće izvršiti ručno.

Programi Sadržaja (en. **Content programs**) su multimedijalne paketi teksta, slike, video i audio snimaka sa računarskim interfejsom. Program sadržaja može biti enciklopedija ili turistički vodič. Uobičajeno je da ima veliki broj datoteka, uključujući bitmape, zvuk i animacije. Mnogi programi su u stvari mali motori (en. **engines**) čiji je glavni cilj da se krećete kroz sadržaj baze podataka.

Enciklopedija i referentne programi su primeri. Moderna komercijalna kompjuterska igra često uključuje dovoljno medijske resurse da bude neka vrsta programske sadržaja po sebi, uzgred, sa prezentacijom sadržaja u kontekstu odvijanja igre.

Iako nema zarade u njima, treba napomenuti da možete da kreirate programe koji su umetnost. Najpoznatiji primeri umetničkih programa su skrin-sejveri. Ovi programi stvaraju slike koje ljudi vole da gledaju. Stvarno dobar umetnički program može da proizvede takav niz efekata da su ljudi prosto hipnotisani.



Slika 5.4 "Životni vek" softvera.Izvor: Autor.

INTERFEJS KORISNIKA

Većina aplikacija zahteva dobar grafički korisnički interfejs

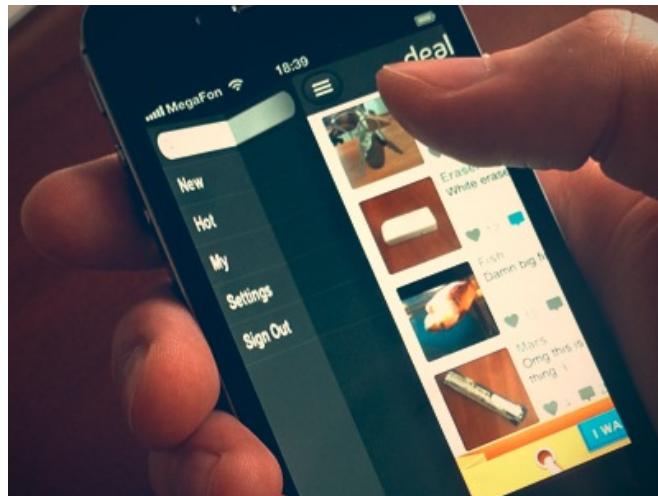
Malo ko je zadovoljan sa programom koji se pokreće iz komandne linije i kontroliše posebnim kombinacijama tastera na tastaturi. Svaki ozbiljan program se mora pokrenuti u grafičkim prozorima, podržava miša i uključuje menije i dijaloge..

Naravno za dobar grafički korisnički interfejs, tzv. GUI (en. **graphic user interface**), potrebno je dosta razmišljanja. Obično je potrebno da isprobate nekoliko interfejsa dok ne dobijete pravi. Crtanje skice interfejsa pre nego što ga je izgraditi je veoma dobra ideja. Postoje neki zajednički dogovoreni principi Windows interfejs dizajna, ali zaista lep i jednostavan dizajn za korišćenje je umetnost sama po sebi.

Ukratko, recept da se razvije dobar set kontrola i selekcijama u menijima: testiraj, gledaj, obrati pažnju, koriguj, koriguj, koriguj,...

Možete da zamislite nekoliko različitih načina na koji neko može da žele da koriste svoj program. U ovom slučaju to je dobra ideja da se napravi interfejs prilagodljiv korisniku. To jest, možda ćete imati da selekcije u menija sa izmenljivim odgovorom programa na određene vrste kontrole miša ili tastature. Ili će možda selekcije menija da kontrolišu način prikazivanja programa.

Pored korisničkih kontrola, možda ćete smatrati da je izbor grafičkih i audio fajlova vaše igre deo dizajna interfejsa. Dobro je imati neku praktično iskustvo sa interfejsom, tako da imate jasnu ideju kako od koje vrste interfejsa su moguće, a takođe i kako da se obezbedi koda kuke za rad sa različitim vrstama kontrola interfejsa.



Slika 5.5 Efikasnost aplikacija zavisi od kvaliteta GUI. Izvor: <https://unsplash.com> (stock image)

Konačna karakteristika interfejsa je rukovanje fajlovima. Ako možete podržite rukovanje filovima, tj. dozvolite i korisniku da sačuva i ponovo učita trenutno stanje programa, uključujući sve trenutne vrednosti parametara.

Alternativno, rukovanje fajlovima može da se koristi kao način da se produži prilagodljivost korisniku, možda omogućavajući korisniku da učita razne grafičke pozadine ili zvučne datoteke u svoj program. U zavisnosti od toga šta se i učita, vaš program može da se ponaša na veoma različite načine. Treba napomenuti **debug-free file** - rukovanje može da bude teško da se održi, pa programi često se zadovolje pojednostavljenim „**save-load**“.

DOKUMENTACIJA I STABILNOST

Podrazumeva se da vaš program ne treba krahira i da je dobro dokumentovan

Dokumentacija je često poslednja stvar o kojoj softverski inženjeri razmišljaju. Nasuprot tome, korisnici softvera polaze od dokumentacije. U ovom kursu ćemo naglasiti važnost rada na uputstvu za korisnika istovremeno dok radite na vašem kodu. A o tome treba da razmišljate od samog početka

Važna stvar o dokumentaciji je da ima korisne i tačne informacije. Korisnik više brine o stvarnoj informaciji nego o tačnom načinu na koji su prozori indeksirani i povezani. Programeri imaju tendenciju da se ponekad izgube u labyrintru dizajna **help fajlova**.

Priročnik korisnika (en. **User's Guide**) treba da sadrži objašnjenje zašto je vaš program zanimljiv, uputstvo za instalaciju i brz start i funkciju za osobinu objašnjenje svih menija i dijaloga kontrole.

Uputstvo za korisnike može biti štampano i online. Oba su obično zasnovani na istom dokumentu. Vaš rad, ponovo, treba da se fokusira na to da dokument bude dobar, a ne da ima atraktivni interfejs i širok spektar funkcija hiperteksta.

Vredi napomenuti takođe da je poželjno što bolje dokumentovati j kod u obliku komentara, u cilju pomoći drugim programerima koji možda rade na vašem projektu - ili da vam pomognu kada se vratite u kod nakon šest meseci.

Stabilnost

Podrazumeva se da vaš program ne treba krahira. Njegovo ponašanje treba da bude stabilno i konzistentno bez obzira šta korisnik radi. Ostvarenje ovog cilja podrazumeva pažljivo i kompletno testiranje i otklanjanje grešaka.

Naravno svaki program ima *bagove* (ili defekte), no svesti ih na prihvatljiv nivo je trajan zadatak softverskog inženjera. Kontrola defekata ima nekoliko delova: defanzivno kodiranje, inspekcije i testiranje kod pronalaženja postojećih grešaka i defekata, i otklanjanje tih defekata.

Još jedan aspekt stabilnosti znači da, kao programer, treba da budete spremni da se izostavite ekstravagantne i čudne segmente koda koji uvek izazivaju nevolje.

Tendencije da se uključi nepotrebno komplikovane funkcije se poznati fenomen kod programera (" **developer gold-plating**").

▼ Poglavlje 6

Vežba – Kako početi PM karijeru u industriji

URAĐENI PRIMER: SPECIFIČNOSTI PM U STUDIJU ZA VIDEOIGRE (25 MIN)

O zapošljenju u industriji igara se puno toga može naći na internetu

Zamislite da ste u sledećoj situaciji.

Vi ste na probnom radu u novoosnovanom indi studiju za izradu igara, kao pomoćnik direktora. Vaš zadatak je da pomognete šefu u razradi metoda menadžmenta softverskih projekata i njihovoj primeni na proces razvoja videoigara u studiju.

Vaš prvi zadatak je da se upoznate sa specifičnostima razvoja softvera u malom indi (eng. **Indie**) timu.

Zaduženi ste da odgledate sledeći video

Indie Game Basics - Vlambeer, <https://www.youtube.com/watch?v=o7KSbdIEAOU>

i da napišete kratak Word dokument (oko 1 strane) o sadržaju videa i korisnim preporukama za vaš dalji rad.

Pogledajte i video na linku desno.

Takođe, potražite još neki koristan video ili materijal s interneta na temu kako postati project manager i indi developer i napravite izbor za svoje kolege iz tima.

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

ZADATAK ZA SAMOSTALNI RAD: (20 MIN)

Napravite analizu prednosti i nedostatka PM-a

Pripremite kratak izveštaj za svoje kolege iz tima na temu šta ste našli na internetu od korisnih preporuka kako da uvedete metode PM-a u vaš indi tim.

Izveštaj bazirajte na video snimku

Indie Game Basics - Vlambeer,

<https://www.youtube.com/watch?v=o7KSbdIEAOU>

i drugim izvorima koje ste našli na internetu.

U izveštaju (obima oko 1 strane) navedite linkove i vaš kratak komentar šta taj link sadrži i zašto će vam koristiti.

Ako imate kolege koji bi mogli biti zainteresovani za indi poduhvat, navedite njihova imena.

Izveštaj treba da sadrži naslov, vaše ime i sadržaj koji odgovara zadatku koji ste dobili.

Meilom ga pošaljite asistentu, stavljajući nastavnika u Cc:

Ako imate kolege koji bi mogli biti zainteresovani za indi poduhvat, pošaljite im vaš izveštaj i zatražite mišljenje. Ukoliko ste zadovoljni onim što vam odgovore (a i ako niste) pošaljite te odgovore asistentu.

Vaš prvi zadatak je da odgledate video iz koga su preuzete lekcije o kojima ste prethodno diskutovali:

Rami Ismail - 20 lessons I learned while running a chaotic game studio,

<https://www.youtube.com/watch?v=fQhPGJQhtak>

Napišete kratak Word dokument (do 1 strane) o sadržaju videa, odnosno o pravom tumačenju Ramijevih lekcija i saveta.

Izveštaj treba da sadrži naslov, vaše ime i broj indeksa i sadržaj koji odgovara zadatku koji ste dobili.

Podelite sa asistentom i kolegama svoje zaključke nakon uporedne analize.

PRIMER: VLAMBEER STUDIO (20 MIN)

Vlambeer studio je indi tim za razvoj igara

Rami Ismal i Jan Willem Nijman su pokrenuli nezavisni studio za razvoj video igara pod imenom Vlambeer. Osnovan je 2010. godine i poznat je po igrama: Ridiculous Fishing, Super Crate Box, Nuclear Throne, Serious Sam.

Rami Ismal je počeo da programira kada je imao šest godina i tada je, još kao dete, promenio svoj osnovni cilj razbijanja stvari u kreiranje novih stvari. Kasnije se dublje uključio u proizvodnju softvera, preduzetništvo i marketing. Kao savesni građanin, u jednom trenutku je odlučio da bi edukacija drugih na temu igara i preduzetništva bila dobra ideja. U školi je upoznao Jan Willem Nijmana i zajedno su pokrenuli Vlambeer studio za razvoj igara. Rami je ujedno poslovan čovek i programer. Radeći više različitih poslova, kao i zbog toga što je bio primoran da postane pravi lider, došao je do 20 lekcija koje je morao da nauči vodeći haotičan gejming studio. Te lekcije deli sa drugima u sledećem videu:

Indie Game Basics - Vlambeer, <https://www.youtube.com/watch?v=o7KSbdIEAOU>

Ovo su lekcije koje Rami navodi:

Lesson 1: "Why" is the biggest question?

Lesson 2: Design is communication

Lesson 3: Be a little less normal

Lesson 4: Give more than you take

Lesson 5: Honesty makes better games

Lesson 6: Take business seriously

Lesson 7: Don't take business seriously

Lesson 8: I'm a fraud & so are you

Lesson 9: Currency is your motivation

Lesson 10: Ideas are worthless

Lesson 11: Failure is good

Lesson 12: An industry is people

Lesson 13: Everything is a remix

Lesson 14: Your game can talk to you, too

Lesson 15: Don't rely on patterns

Lesson 16: The world is out there

Lesson 17: You can't make the game in your head

Lesson 18: Don;t define things for others

Lesson 19: Decide - dont accept!

Lesson 20: Dare to ask questions

Pitanja za diskusiju : Iznesite svoje tumačenje Ramievih lekcija

1. Proučite listu lekcija koje je Rami izneo na osnovu svog upravljanja malim timom.
2. Vratite se na prvu lekciju. Dajte pretpostavke na šta se ova "lekcija" odnosi, kako bi mogla da bude protumačena. Kako vi razumete Ramijev zaključak?
3. Uradite isto za svaku od 20 lekcija. Diskutujte sa asistentom o sopstvenom tumačenju ovih saveta.

URAĐENI PRIMER: PRAVILA LIČNE I PROFESIONALNE ORGANIZACIJE (25 MIN)

Koliko se pravila upravljanja projekata mogu primeniti na svakodnevni život

Bilo da radite individualne projekte za potrebe predmeta na studijama ili razvijate nešto u timovima od dvoje-troje ljudi, sigurno ste već primetili da vam je za uspeh potrebna dobra organizacija. Verovatno ste se već susretali sa time da morate mnogo više napora da uložite u poslednjoj etapi razvoja, zato što niste dobro isplanirali vreme ili niste pravilno rasporedili zadatke u zadatom roku.

Ako malo bolje razmislite, iste zakonitosti važe i za bilo koje druge organizacije koje ste imali u privatnom životu. Kad god niste dobro isplanirali svoje vreme, aktivnosti ili pak raspoloživi budžet, doveli ste sebe u haotičnu situaciju. Morali biste da rešavate problem primenom plana B, angažovanjem dodatnih ljudi da vam pomognu ili prosto produženjem roka, ako je to bilo moguće. Nekada ste se susretali i sa apsolutnim neuspehom, na primer kada ne biste mogli da izađete na ispit zato što niste uspeli da završite sve predispitne obaveze u propisanom roku.

Sada možete da zaključite da za bilo koje projekte koje razvijate: privatne, individualne ili timske, važe naka zajednička pravila. Jedno od njih je da morate napraviti dobar plan i da ga se pridržavate. Drugo je da morate dobro da izbalansirate tri najvažnije stvari: vreme, aktivnosti (zadatke) i budžet (ako postoji).

Pitanja za diskusiju

Dobre i loše prakse lične organizacije

1. Kada biste imali svoj start-up, kako biste organizovali rad u timu kao lider?
2. Kada biste radili u tuđoj kompaniji, kako biste se postavili prema sopstvenim zaduženjima, drugim članovima tima, menadžeru projekta?
3. Kako organizujete rad na svojim individualnim projektima za potrebe studijama? Koji su vam propusti i nedostaci u organizaciji? Šta nameravate da promenite?
4. Koje su dobre prakse koje biste podelili sa kolegama?

Zadatak za samostalni rad

Kako biste unapredili menadžment Vlambeer studia

1. Zamislite da ste u sledećoj situaciji.

Vi ste na probnom radu u Vlambeer studiju za izradu igara, a vaš zadatak je da pomognete Ramiju u razradi boljih metoda menadžmenta softverskih projekata i njihovoj primeni na

proces razvoja video igara u studiju.

Napišite kratak Word dokument (do 1 strane) o korisnim preporukama za dalji rad ovog studia za razvoj igara. Koje biste promene vi uveli? Koje biste preporuke za poboljšanje timskog rada dali?

▼ Poglavlje 7

Zaključak

ZAKLJUČAK

Rezime lekcije

Svaki projekat ima cilj, zahteva vreme, rad i materijalna sredstva za realizaciju. Svi projekti prolaze kroz faze iniciranja, planiranja, izvršavanja i završetka.

Upravljanje projektima razvoja softvera izučava potrebna znanja iz oblasti planiranja, organizacije i praćenja svih faza životnog ciklusa softvera.

Rukovođenje (Management) je od presudnog značaja da bi razvojni projekti softvera bili odgovarajući za organizaciju, da rad u različitim organizacionim jedinicama bude koordinisan, da se održava konfiguracija softvera i verzije softvera, da resursi budu na raspolaganju kada je to potrebno, da rad na projektu bude raspodeljen na odgovarajući način, komunikacija učesnika olakšana, a napredak se precizno prati i beleži.

Projekti mogu biti različite složenosti, odnosno obima (klasifikovali smo ih kao vrlo male, male i velike). Sa složenošću projekta raste i složenost upravljanja.

U kursevima softverskog inženjerstva češće govoriti o specifičnim tehnikama menadžmenta i faze razvoja softvera dele na analizu, dizajn, Implementaciju, testiranje i održavanje. Kriterijumi za uspešni program su da li je profitabilan (da li donosi pare) i da li je dobar . Da bi softver bio dobar treba da ima odgovarajući koncept (zavisno od namene), da ima odgovarajući interfejs koji korisniku omogućuje lako korišćenje, da ima adekvatnu dokumentaciju i da pokazuje stabilnost u radu. Ponašanje softvera treba da bude stabilno i konzistentno bez obzira šta korisnik radi, a ostvarenje ovog cilja podrazumeva pažljivo i kompletno testiranje i otklanjanje grešaka.

Mali projekti (tim 2 - 5 člana, nekoliko meseci) mogu osetno podići efikasnost primenom tehnika menadžmenta, jer je potrebna komunikacija između projektnog tima, finansijera projekta i zainteresovanih (stakeholders).

U malom projektu se kao česnici pored tima za realizaciju projekta pojavljuju i finansijer (project owner), stejkholderi (zainteresovani) i menadžer projekta. Iako broj članova projektnog tima nije veliki, pojava i drugih učesnika na projektu onemogućava da se projekat vodi "iz glave". Potrebna je i pisana dokumentacija (ugovori, zapisnici, računi itd.), potrebna je komunikacija sa svim učesnicima u cilju izmene informacija, koordinacije i kontrole izvršavanja poslova i praćenja kvaliteta. Vođenje malih projekata zahteva i uspostavljanje procedura upravljanja projektom kao i poznavanje raznih tehnika menadžmenta koje se uče u školama ili na raznim kursevima. No vođa projekta se ne bavi isključivo rukovođenjem jer tim za realizaciju projekta nije dovoljno veliki.

Veliki i vrlo veliki projekti po pravilu imaju profesionalno obučen menadžerski tim.

REFERENCE

Korišćena literatura i linkovi

- [1] Guide to the Software Engineering Body of Knowledge (SWEBOK), Chapter 8: Software Engineering Process http://swebokwiki.org/Chapter_8:_Software_Engineering_Process
- [2] Project Management Institute Standards Committee, A Guide to the Project Management Body of Knowledge (PMBOK), Project Management Institute, 2000.
- [3] Project Sizes, <http://www.mpmm.com/project-sizes.php>
- [4] Rudy Rucker, Software Engineering and Computer Games, Addison-Wesley, 2002
- [5] Paulk M. C., Using the Software CMMÒ in Small Organizations, Eighth International Conference on Software Quality, October, Portland, Oregon, 1998
- [1] Indie Game Basics – Vlambeer, <https://www.youtube.com/watch?v=o7KSbdIEA0U>
- [2] Rami Ismail, https://en.wikipedia.org/wiki/Rami_Ismail
- [3] Vlambeer, <https://en.wikipedia.org/wiki/Vlambeer>
- [4] Watch Vlambeer's 17 lessons for indie developers, <https://www.gamesindustry.biz/articles/2014-07-24-watch-vlambeers-17-lessons-for-indie-developers>



SE325 - UPRAVLJANJE PROJEKTIMA RAZVOJA SOFTVERA

Specifičnosti softverskih projekata

Lekcija 02

PRIRUČNIK ZA STUDENTE

SE325 - UPRAVLJANJE PROJEKTIMA RAZVOJA SOFTVERA

Lekcija 02

SPECIFIČNOSTI SOFTVERSkiH PROJEKATA

- ✓ Specifičnosti softverskih projekata
- ✓ Poglavlje 1: Osnovni proces upravljanja projektima
- ✓ Poglavlje 2: Inicijalni plan projekta
- ✓ Poglavlje 3: Osnove projekata softverskog inženjerstva- SDLC
- ✓ Poglavlje 4: Projektni zahtevi
- ✓ Poglavlje 5: Procesi softverskog inženjerstva
- ✓ Poglavlje 6: Pokazna vežba: Vaš prvi PM zadatak u industriji
- ✓ 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

Izlažemo proces upravljanja projektima, izložen sa najmanje detalja, da bi se stekla globalna slika

Tokom prethodnog izlaganja je napomenuto da sa stanovišta upravljanja ili menidžmenta, po mnogim aspektima projekti softverskog inženjerstva, (što ovde zovemo kraće softverski projekti), "liče" na projekte iz drugih oblasti.

To znači da se mogu primeniti iste ili slične procedure, uz eventualne manje (ili veće, ako se dogovrimo šta je to malo a šta veliko) izmene.

Kao da ste napravili jednu opštu klasu (za one koji vole Objektno Orientisan (OO) prilaz), koju kasnije instancirate i dodeljujete specifične vrednosti parametrima i funkcijama.

U prvom delu ovog izlaganja ćemo dati jednu globalnu proceduru (ili tok – flow) projekta koja je primenljiva kako na softverske tako i na druge projekte

Za one koji baš vole OO prilaz, evo i jednog izazova. Možete li da prema izloženoj proceduri napravite arhitekturu softvera koji će simulirati jedan projekat (makar i veoma mali) u smislu da ima iste ulaze i izlaze kao i realni projekat? Ako makar samo i pokušate da to uradite, biće vam puno jasnija materija koja će biti izlagana.

Tim pre, što u veštini upravljanja projektima postoji težnja da se uvedu procedure ili procesi, ulazi izlazi, uvode se templejti kao i u softverskom inženjerstvu. Jedino što ih (za sada) izvršavaju ljudi a ne računar. Međutim, već postoje "inteligentni" softveri koji pomažu menadžeru u upravljanju.

Možda ćete baš vi raditi na razvoju softvera koji će upravljati projektom. Ili, ako radite na razvoju igre koja simulira upravljanje projektom (pogledate internet), možda će baš vaš zadatak biti razvoj virtuelnog menažera.

Videćete da je ono što je tema sledećeg poglavlja potrebno ali ne i dovoljno.

No, malo smo se udaljili od tema.

Najpre pogledajmo proces upravljanja projektima, izložen sa najmanje detalja, da bi se stekla globalna slika. Kasnije (u narednim izlaganjima) će taj proces biti detaljnije izložen i elaboriran, da bi mogao biti primenjen u realnim projektima.

Nakon toga ćemo izložiti kako izgleda projekat sa aspekta softverskog inženjera, a to je upravo ono što oni rade u projektu koji vode profesionalni menadžeri.

U malom, ili veoma malom projektu, nema baš puno mesta za usku specijalizaciju, pa se obično dešava da softverski inženjer i vodi kompletan projekat, tj. preuzima ulogu menadžera projekta.

✓ Poglavlje 1

Osnovni proces upravljanja projektima

POSTUPAK REALIZACIJE PROJEKTA

Projekat se sastoji iz niza etapa poređanih logičnim redosledom

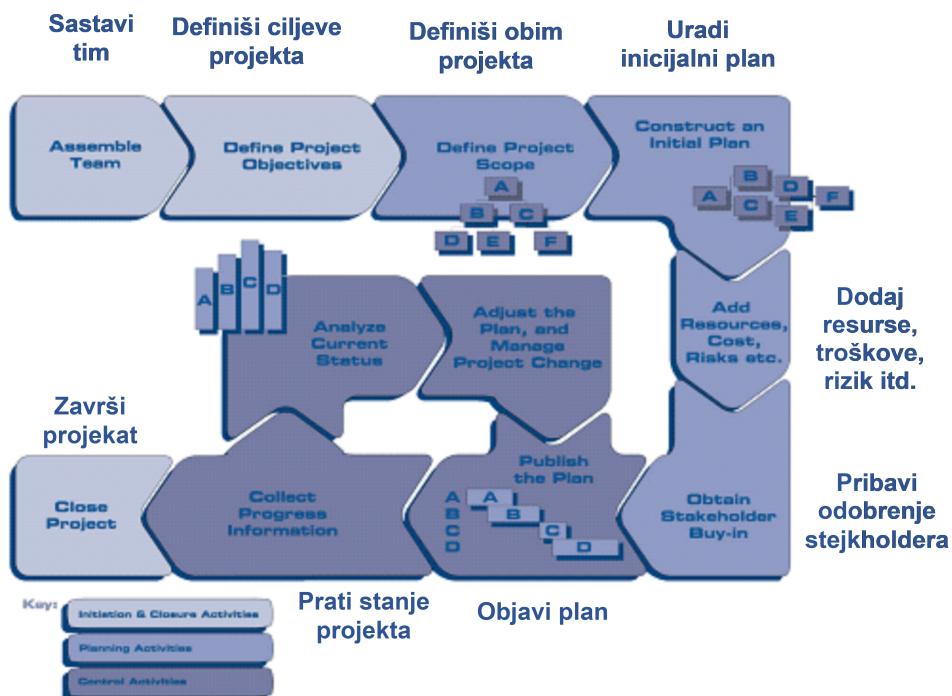
Projekat se sastoji iz niza etapa poređanih logičnim redosledom

Upravljanje većinom projekata se može opisati globalnim modelom prikazanom na sledećoj slici.

Naime, upravljanje se može shvatiti kao proces koji se sastoji od etapa (ili faza) od koji svaka (sem početne) ima ulaze iz prethodne faze i svaka ima izlaze (ili rezultate).

Etape realizacije projekta su sledeće.

- Oformljuje se tim za pokretanje projekta
- Definišu se ciljevi projekta
- Definiše se obim (**scope**) projekta
- Izrada inicijalnog plana
- Razrada resursa, troškova i rizika
- Pribavljanje saglasnosti zainteresovanih
- Objavljivanje plana projekta
- Praćenje napretka realizacije projekta
- Analiza trenutnog stanja
- Korekcije plana
- Završetak projekta



Slika 1.1 Osnovni proces upravljanja projektima. Izvor: Autor.

OPIS ETAPA FORMIRANJE TIMA I DEFINISANJE CILJEVA PROJEKTA

Najpre se formira tim koji treba da osmisli kako će se projekat izvesti

1. etapa-Tim za pokretanje projekta

Tim za pokretanje, planiranje i vođenje projektni treba da se sastavi, uključujući i odgovarajuće zainteresovane kupce / klijente, a ponekad i podizvođača i prodavaca. Dodeljuju se početne uloge i odgovornosti.

Rezultati etape (en. **deliverables**): Početna (inicijalna) dokumentacija projekta.

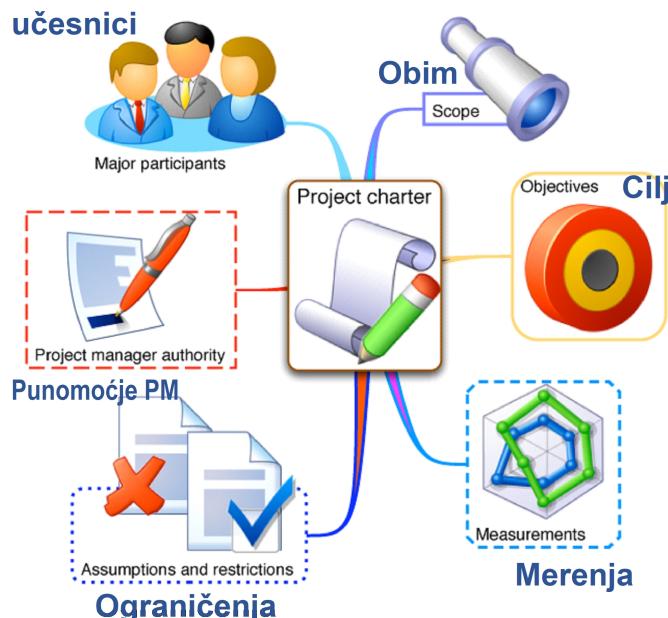


Slika 1.2 Dodeljuju se početne uloge u timu. Izvor: <https://unsplash.com/images/stock>

2. etapa - Definisanje ciljeva projekta

Nakon formiranja tima za organizaciju projekta, tim verifikuje opšti cilj projekta i razrađuje detaljno ciljeve projekta. Vrši se provera rezultata date faze kako bi se utvrdilo da je projekat spremna da kreće u narednu fazu, koja se planira.

Rezultati etape (**Deliverables**): Definicija Projekta (**project charter**), Kontrolna lista pregleda rezultata faze



Slika 1.3 Project charter ili definicija projekta. Izvor: Autor.

DEFINISANJE OBIMA (SCOPE) PROJEKTA

Obim (scope) projekta je lista šta će se raditi i šta će biti urađeno

3. etapa - WBS tj. obim (en. scope) projekta

Radi se lista poslova, tzv. WBS (en. **Work Breakdown Structure**) koje treba uraditi da se ostvarili ciljevi projekta.

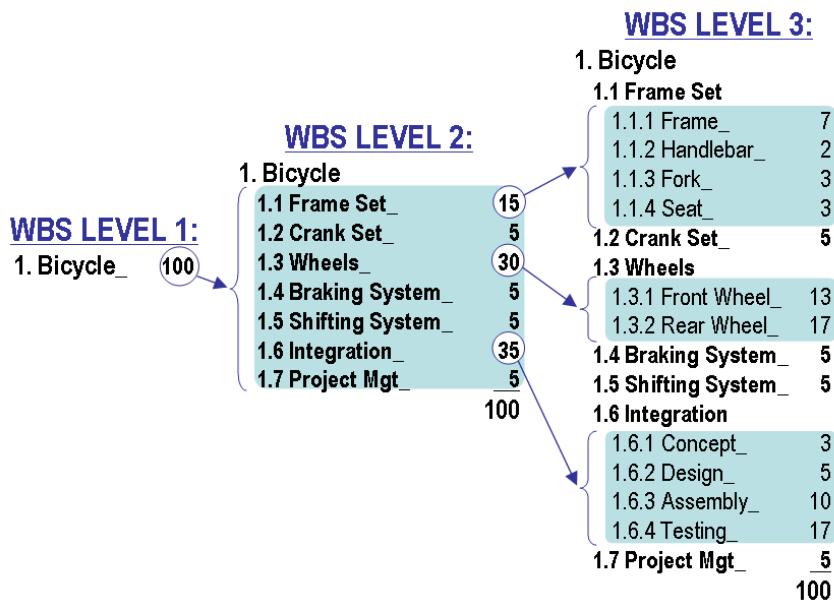
Posebno se obraća pažnja da se precizno utvrdi šta treba da bude rezultat svakog posla (**deliverable**).

Na osnovu WBS se vrši provera da li su se zainteresovane strane pravilno dogovorile i razumele šta sve projekat sadrži.

Projekat A Lista aktivnost

No.	Aktivnost	Trajanje (dana)
1	Aktivnost 1	3
2	Aktivnost 2	5
3	Aktivnost 2	1
4	Aktivnost 4	2
5	Aktivnost 4	7

Slika 1.4 Lista poslova (ili aktivnosti). Izvor: Autor.



Slika 1.5 WBS (Work Breakdown Structure). Izvor: <https://projectsheets.wordpress.com/2008/01/13/work-breakdown-structure-2/>

WBS takođe omogućava da kompletan projekat bude podeljen na odgovarajuće potprojekte i / ili faze.

Rezultati etape (**deliverables**): Llista poslova - WBS.

✓ Poglavlje 2

Inicijalni plan projekta

IZGRADA INICIJALNOG PLANA

Nakon kreiranja WBS tim procenjuje trajanje poslova i redosled izvršenja

Inicijalni plan projekta

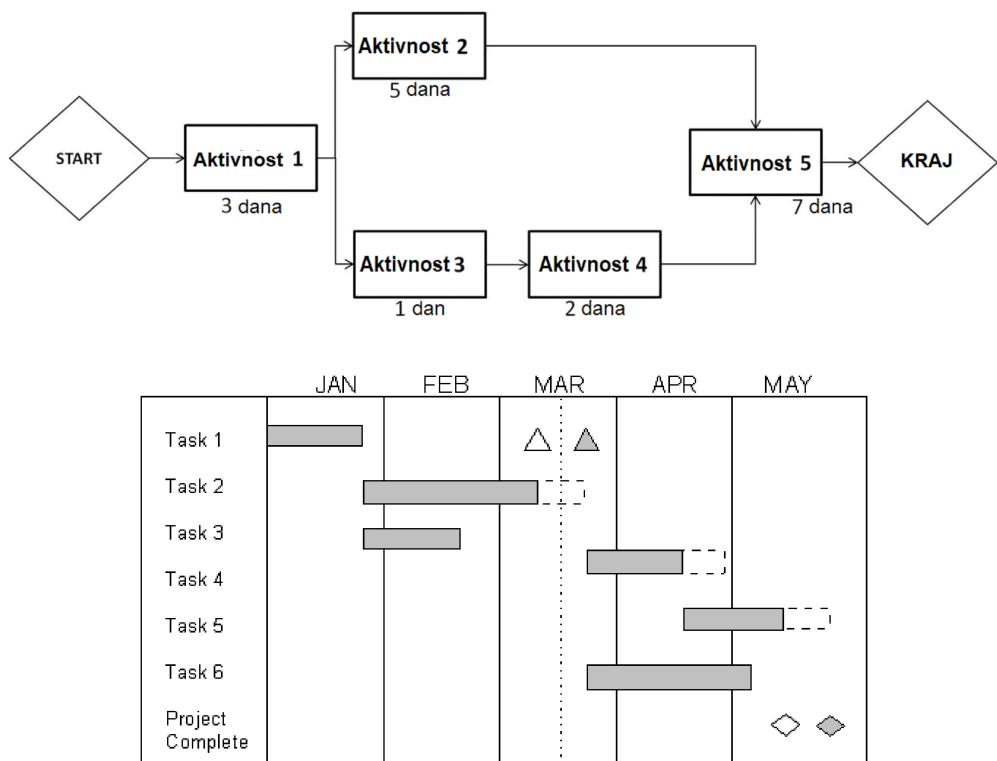
Dakle, najpre se utvrđuju poslovi ili zadaci (tasks) i kreira tzv. WBS (Work Breakdown Structure).

Nakon toga projektni tim radi logičke mreže i dijagrama tih poslova i zadataka da bi se odredila zavisnost jednih poslova od drugih i odredio redosled izvršavanja, sa procenjenih trajanja pojedinih poslova.

Ovo omogućava menadžeru projekat da predvidi kada će sve aktivnosti biti završene.

Nakon toga se procenjuje izvodljivost postavljenih rokova, identificuje kritični put za projekat (tj. Niz aktivnosti od koga zavisi trajanje projekta).

Rezultati etape (Deliverables): Početni plan rada.



Slika 2.1 Zavisnost, redosled i trajanje poslova. Izvor: Autor.

RAZRADA RESURSA, TROŠKOVA I RIZIKA

Troškovi rizik projekta su veoma bitni detalji definicije projekta

5. etapa - Razrada resursa, troškova, rizika i ostalih detalja projekta

Određene resursi projekta se mogu definisati kao kritični (znanja, oprema i drugo bez čega se projekat ne može realizovati). Konkretno, rukovodilac projekta može proceniti da će ključno osoblje projekta biti suočeno sa previše posla. Ako je tako, planom projekta se predviđa proširivanje resursa (angažovanje novih ljudi, firmi podugovarača, opreme itd.).

Cena (tj. troškovi) je očigledno od presudnog značaja, a rashodi procenjuju i dodaju u plan trošenja sredstava predviđenih za projekat.

Upravljanje rizikom se takođe može koristiti na projektima da obezbedi bolje upravljanje projektom za slučaj pojave nepredviđenih događaja, koji se dešavaju van kontrole projektnog tima.

Rezultati etape (**Deliverables**): Dostupnost i profili resursa, identifikacija rizika, strategije i kontrola, procena utroška sredstava.

6. etapa - Pribavljanje saglasnosti zainteresovanih (en. **stakeholder-a**)

Da bi se obezbedila što je moguće ravnomernija realizacija projekta, neophodno je da se preispitaju prvobitni planovi zajedno sa svim glavnim akterima projekta i dobije podrška i saglasnost uključenih strana. Izvodi se pregled-provera rezultata ove faze kako bi se utvrdilo da je projekat je spremna da pređe u sledeću fazu.

Rezultati etape (**Deliverables**): Odobreno konačni plan, Kontrolna lista pregleda rezultata faze.



Slika 2.2 Identifikacija rizika je veoma važna za uspeh projekta. Izvor: <https://unsplash.com/images/stock>

7. etapa - Usvajanje i objavljivanje plana

Kada su planovi usaglašeni, moraju biti dostavljeni svim zainteresovanim stranama (**stakeholders**). Ovo se može uraditi u papirnoj formi ili preko elektronskih medija, u zavisnosti od raspoloživih resursa. Na većini projekata je razvijen i plan komunikacije, a distribucija planova će pratiti smernice date u planu komunikacije.

Rezultati: plan dostavljen svim zainteresovanim stranama.

PRAĆENJE PROJEKTA

Projektni tim redovno izveštava menadžera projekta o napretku projekta

8. etapa Prikupljanje informacija napretku realizacije projekta

Menadžer projekta prati informacije o napretku projekta koje projektni tim dostavlja u svojim redovnim izveštajima. Na osnovu njih menadžer može da sastavi izveštaje o napretku projekta da bi informisao zainteresovane, kao što su *aktivnosti završene u protekle dve nedelje, prognoza učinka po aktivnostima aktivnosti za naredne dve nedelje, sa fokusom na aktivnosti na kritičnom putu, odnos utrošeni sredstava i predviđenih troškova, izveštaj o prioritetnim problemima*

Može takođe biti primjenjeno merenje napretka projekta i na druge načine, kao što su zarađena vrednost ili statistika opterećenja resursa. Ako menadžer projekta na osnovu podataka o napretku zaključi daje projekat urađen, svi rezultati će biti pregledani da se utvrdi da su svi ciljevi ispunjeni pre prelaska u fazu konačnog završetka. Rezultati etape: Set izveštaja o toku projekta, set od izveštaja o vanrednim događajima i odstupanjima, izveštaj o merenju, (Kontrolna lista rezultata faze).

9. etapa Analiza trenutnog stanja

Analizirajući dobijene podatke o napretku, menadžer projekta će biti u stanju da utvrdi preciznije podatke za pomenute izveštaje, npr. o tome koje su detalji projekta od prioritetnog značaja, gde će se verovatno u budućnosti pojaviti problemi i slično.

Ovo omogućava menadžerima da se fokusiraju na važne i kritične detalje projekta. Rezultati: Izveštaj Evaluacija projekta.

10. etapa Prilagođenje plana, i upravljanje izmenama projekta

Na osnovu ukazanih potreba i analize, a uz podršku projektnog tima, menadžer projekta će prilagoditi plan u cilju smanjenja rizika, usled promena obima projekta, ili da nadoknadi aktivnosti koje nisu izvršene po rasporedu i u zadatim rokovima. Nakon izmena, plan se ponovo objavljuje. Ovaj ciklus se ponavlja sve dok projekat ne bude završen.

Napomena: svaka promena plana je nepoželjna, jer zahteva dodatno vreme i mere, te se izvodi samo ako se mora (tj. bez izmena bi efikasnost rada na projektu bila niža).

Rezultati: Zahtevi za izmenama plana, Ažurirani plan.

11. etapa Završetak projekta

Kada su postignuti ciljevi projekta, rukovodilac projekta prelazi na fazu završetka projekta. To uključuje završetak ugovora i zadatke zatvaranja finansijskih aktivnosti (isplate, naplate), kao i arhiviranje materijala projekta. Rade se analize i оформљује dokument o iskustvima projekta (**lessons-learned**) koji će se koristiti za buduće projekte. A ako je moguće projektni tim će održati proslavu završetka projekta.

Rezultati: Završni izveštaj projekta uključujući „naučene lekcije“.

▼ Poglavlje 3

Osnove projekata softverskog inženjerstva- SDLC

UVODNE NAPOMENE

Životni vek softvera je niz etapa razvoja i korišćenja softvera

U ovom poglavlju ćemo prodiskutovati neke od osnovnih tehnika i alata softverskog inženjerstva koji su potrebni da bi se upravljalo softverskim projektom, zadržavajući se najpre na opštim stvarima, da bi ste stekli uvid u celinu. Kasnije ćemo razrađivati detalje, fokusirajući se pre svega na one koji su bitni za male projekte (3 do 5 članova u timu, do 15 čovek-meseci).

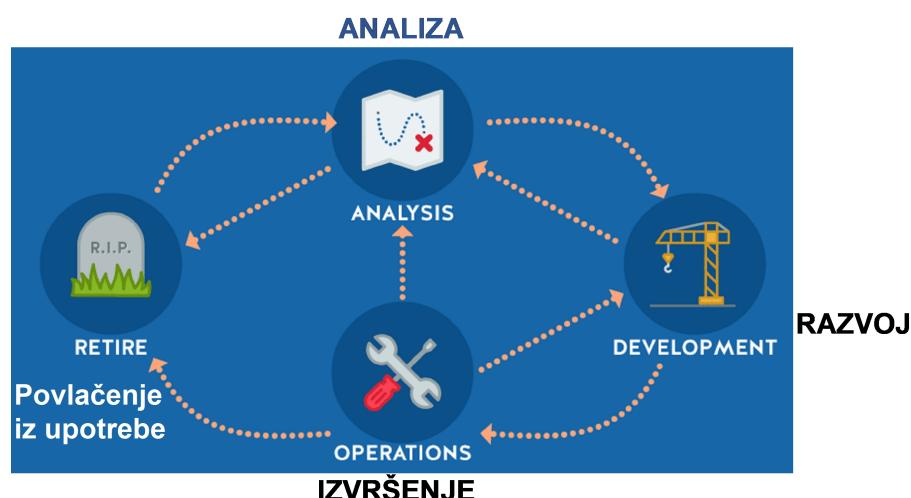
Dakle, šta je to specifično za softverske projekte? Pre svega, sa stanovišta teoretičara softverskog inženjerstva ceo razvoj se posmatra iz ugla proizvoda softvera koji treba da nastane.

Dakle, kao da se softver začinje, rađa, odrasta i odlazi od roditelja, govori se o životnom veku softvera (**Software Life Cycle**).

Životni vek softvera je period od nastanka ideje, preko razvoja, upotrebe i održavanja do prestanka korišćenja softvera.

Češće je u upotrebi termin životni vek razvoja softvera (**Software Development Life Cycle, SDLC**), koji praktično znači isto, s tim što je dat naglasak na razvoju.

U ovom kursu će oba termina biti korišćena kao sinonimi.



Slika 3.1 Životni vek softvera. Izvor: Autor.

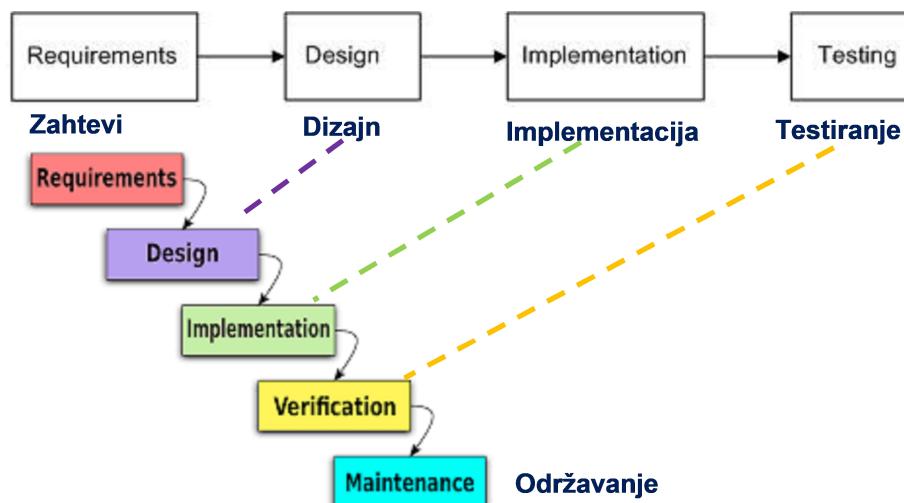
MODEL ŽIVOTNOG VEGA RAZVOJA SOFTVERA

Model je apstrakcija kojom se opisuje realna pojava

Modeli životnog veka softvera opisuju faze ciklusa softvera i redosleda u kome se izvode te faze.

Model (kao model) je apstrakcija kojom se opisuje realna pojava (u datom slučaju razvoj softvera). U upotrebi postoji veoma mnogo modela, a mnoge kompanije usvoje svoje modele, ali svi su veoma slični.

Generalno , osnovni model (i najčešće korišćeni) je prikazana sledećom slikom :



Slika 3.2 Osnovni model životnog veka softvera. Izvor: Autor.

Svaka faza proizvodi ono što zahteva sledeća fazu u životnom ciklusu . Zahtevi se prevode u dizajn . Kod se proizvodi tokom implementacije koji se vrši u skladu s dizajnom . Testiranje proverava rezultate faze implementacije (**deliverable**) u odnosu na zahteve .

Zahtevi

u ovoj fazi se prikupljaju poslovni zahtevi. Ova faza je glavni fokus menadžera projekta i zainteresovanih . Održavaju se sastanci menadžera , zainteresovanih (**stakeholders**) i korisnika kako bi se utvrdilo zahtevi koje proizvod-softver treba da zadovolji .

Ko će da koristi sistem ? Kako će sistem biti korišćen? Koji podaci treba da budu ulaz u sistem ? Koji podaci treba da budu izlaz iz sistema ?

Ovo su opšta pitanja koja se odgovara tokom faze prikupljanja zahteva. Dobija se veliki spisak funkcionalnosti koje sistem treba da obezbedi , funkcija koje sistem treba da obavlja , poslovnu logiku da obrade podataka , koji podaci se čuvaju i koriste u sistemu , i kako treba da izgleda korisnički interfejs . Ukupan rezultat je opis sistema u celini i šta i kako obavlja , ali ne i kako će to da radi .

DIZAJN, IMPLEMENTACIJA, TESTIRANJE I ODRŽAVANJE SOFTVERA

Model životnog veka razvoja softvera je apstrakcija kojom se opisuje realni razvoj softvera

Dizajn

Na bazi rezultata faze zahteva radi se dizajn softverskog sistema. Ovo je faza gde se određuju detalji kako će sistem raditi . Arhitektura sistema, uključujući hardver i softver , komunikacije , dizajn softvera su deo produkata faze dizajna .

Implementacija

Na osnovu dokumenata iz faze dizajna radi se kod , i to je najduža faza razvoja životnog ciklusa softvera . Za programera , to je glavni fokus životnog ciklusa , jer ovo je mesto gde se vrši kodiranje . Implementacija se može preklapati i sa fazom dizajna i sa fazom testiranja.

Postoji mnogi alati ([CASE](#) alati) koji automatizuju generisanje koda koristeći informacije prikupljene u fazi zahteva ili kreirane tokom faze dizajna .

Testiranje

Tokom testiranja se proverava da li je implementacija urađena u skladu sa zahtevima. Cilj je da se proveri da li proizvod zapravo rešava potrebe utvrđene tokom faze prikupljanja zahteva . Tokom ove faze se vrše Unit (jedinični) testovi, testovi sistema i primopredajni testovi . Unit testovi proveravaju posebnu komponentu sistema , a sistem testovi proveravaju sistem u celini .

Dakle, u najkraćem , ovo je veoma osnovni pregled opšteg modela razvoja softvera ([SDLC](#))

Korišćenje i održavanje softvera

Korišćenje softvera je razlog zašto se softver pravi. S druge strane, neočekivano ponašanje softvera pri korišćenju je pre pravilo nego izuzetak, što upravo čini razvoj softvera komplikovanim a otuda i menadžment softverskih projekata.

▼ Poglavlje 4

Projektni zahtevi

ZAHTEVI I SPECIFIKACIJA

Zahtevi su opis toga šta softver treba da radi jezikom korisnika

Iako na prvi pogled može da izgleda da su zahtevi i specifikacije isto, u softverskom inženjerstvu postoji jasna razlika.

Zahtevi su opis toga šta softver treba da radi jezikom korisnika.

Specifikacija je opis toga šta softver treba da radi jezikom softverskog inženjerstva koji je daleko precizniji. Često se koristi skraćenica **SRS** (en. **Software Requirement Specification**).



Slika 4.1 Izrada specifikacije softvera. Izvor: Autor.

Razvoj softverskog proizvoda počinje izradom zahteva za određenu vrstu programa i kratke specifikacije kakav bi mogao biti taj program. Zahtevi su nešto kao pitanje a specifikacija je kao odgovor. Malo drugačije rečeno, zahtevi su kao zadatak ili porudžbina a i specifikacija je predloženo rešenje. Upotreba "zahtevi" reči i "specifikacija" je donekle fluidna, i naći ćete različite konvencije u različitim knjigama o softverskog inženjerstva.

Veoma često zahtevi su izrečeni jezikom korisnika i mogu predstavljati željena svojstva proizvoda, koja se možda mogu ispuniti, a možda i ne, dok je specifikacija iskazana jezikom softverskog inženjerstva i to tako da se može ispuniti. No jasno je da postoji značajna interakcija između zahteva i specifikacije.

Tokom procesa prikupljanja zahteva, učesnici u projektu (korisnici i učesnici) pokušavaju da kroz iterativni proces usklade zahteve i specifikaciju. Zainteresovane strane (**stakeholders**) mogu uključivati korporativne klijente, investitore, menadžere, programera i možda uzorak eventualnu korisnika.



Slika 4.2 Izvor: Autor.

FAZA UTVRĐIVANJE ZAHTEVA

Proces da se usklade zahteve i specifikacija je iterativni

Tokom procesa prikupljanja zahteva, učesnici u projektu (korisnici i učesnici) pokušavaju da kroz iterativni proces usklade zahteve i specifikaciju. Zainteresovane strane (**stakeholders**) mogu uključivati korporativne klijente, investitore, menadžere, programera i možda uzorak eventualnu korisnika.

Dakle, još jednom, softverski zahtevi kažu šta program treba da uradi.

Zahtevi mogu biti nešto kao,

"Napisati program koji prikazuje naše podatke o zalihamama u atraktivnom formatu", '

ili nešto manje precizan kao,

"Veb browser koji radi na mobilnim telefonima",

ili dosta neprecizno,

"Uradite zaista lepu igru".

User story („priča korisnika“ koju prevodimo sa zahtev korisnika) je forma za iskazivanje zahteva korisnika, iskazanih jezikom razumljivim korisniku. Primeri:

- "Napisati program koji prikazuje naše podatke o zalihamama u atraktivnom formatu"
- "Veb browser koji radi na mobilnim telefonima"
- "Uradite zaista lepu igru"

Na preliminarnom nivou, početni zahtevi se mogu nazvati vizijom ili konceptom softvera (**vision or a software concept**).

Sem što kažu šta program zahtevi mogu da sadrže i opis svojstava ili specifičnih funkcija koji se očekuju da ima program.

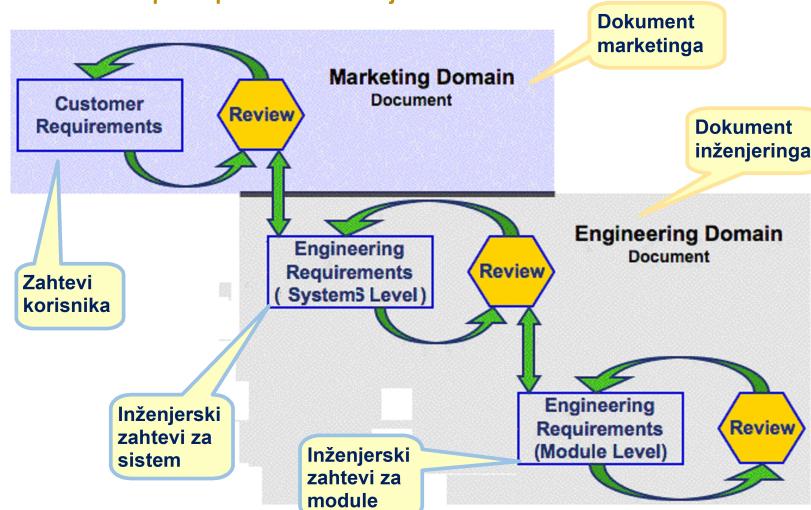
Ponekad softver zahtevi mogu biti veoma detaljni, ali češće su kratki.

U industriji, postoji još jedan aspekt softverskih zahteva a to je potreba konkurentnosti proizvoda na tržištu. Zahtevi industrijskog softvera će reći i zašto vredi da se program uradi i zašto će korisnici željeti da ga imaju. Menadžeri imaju veoma ustaljeno pitanje, " Koje je ciljno tržište ?"

Dakle, obično softverski zahteva uključuju odgovor na ovo pitanje, kao i na pitanje koje su koristi od upotrebe programa.

Typical Requirements Process

Tipičan proces utvrđivanja zahteva



Slika 4.3 Tipični process utvrđivanja zahteva. Izvor: Autor.

SRS- UML DIJAGRAMI

UML je pokušaj da se usklade različite vrste načina kako softverski inženjeri govore o tome šta rade

Poslednjih godina je došlo do pokušaja da se ujednače različite vrste načina kako softverski inženjeri govore o tome šta rade. Rezultat je labavo definisan skup imena i konvencijam koji se zove **Unified Modeling Language**, ili **UML**. UML se prvenstveno koristi kao metodologija za crtanje dijagrama koji se odnose na kreiranje programa. Činjenica da je "ujednačena" znači da UML obuhvata doprinos i usaglašavanje mnogih računarskih eksperata, do tačke u kojoj je postignuto kakvo-takvo jedinstveno rešenje. Postoji najmanje devet različitih vrsta UML dijagrama :

- Dijagram slučaja korišćenja (**use case**),
- Dijagram klase (**class**),
- Dijagram objekta (**object**),
- Dijagram aktivnosti (**activity**),
- Dijagram Stanja (**statechart**),
- Dijagram sekvencije (**sequence**),
- Dijagram saradnja (**collaboration**),
- Dijagram komponente (**component**), i
- Dijagram raspoređivanja (**deployment**).

Takođe, treba napomenuti da veći deo UML, mada ne sav, prepostavlja da koristite objektno - orijentisani stil softverskog inženjerstva.

U ovom kursu ćemo koristiti samo sledećih pet vrsta UML dijagrama.

Use Case dijagrami - za zahteve softvera.

dijagrami komponenti - za zavisnosti od izvornog koda datoteka.

dijagrame aktivnosti - za izvršenje programa dijagramima toka.

dijagrami klase - za strukture na visokom nivou.

dijagrame Sekvenca - za interakcije programskih objekata.

Osnovno što treba imati u vidu o UML dijagramima je da su zamišljeni kao prilično jednostavni. Osnovna svrha UML dijagrama je da olakšavaju raznim učesnicima i zainteresovanim (**stakeholders**) na projektu da što lakše komuniciraju jedni sa drugima. Imajte na umu da ne mali broj stakeholdera ne mora biti kvalifikovan za softversko inženjerstvo. Posebna vrednost UML dijagrama je za vreme prikupljanja zahteva.

Druga svrha UML dijagrame je za ono što se zove forward inženjering (inženjering napred), gde krećemo od koncepta ka stvarnom kodu. UML dijagrami klase su naročito korisni kada ste u fazi razrade dizajn za vaš program na visokom nivou. UML dijagrami aktivnosti su korisne za razumevanje ukupnog toka programa. UML dijagrami sekвенце су добри за rad na detaljima o tome kako objekti sarađuju (inter-reaguju).

Treći primena je UML dijagrama za reverzni (obrnuti) inženjering, koji ima za cilj razumevanje kako postojeći program radi.

Ovo je u stvari situacija u kojoj su oni koji pokušavaju da shvate kako je napravljena neka igra. Ovo je takođe situacija u kojoj ste obično kada počnete da radite za velike kompanije. One imaju neku veliku količinu već razvijenog koda, a vi bi trebalo da počne održavanje ili nastavak razvoja. UML dijagrami su savršen alat za početak rada sa procesom koji se uveliko odvija.

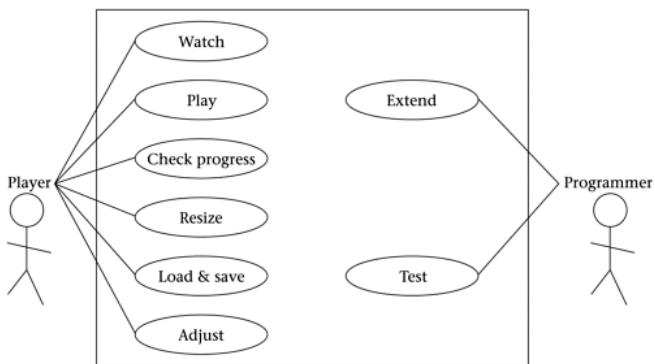
USE CASE DIJAGRAMI

Dijagram slučaja upotrebe kaže kako se očekuje upotreba softvera

Dijagram slučaja upotrebe (**Use Case**) može vam pomoći da razumete šta program treba da uradi. dijagram Komponenti postavlja zavisnostima fajlova s izvornim kodom. Dijagram aktivnosti prikazuje ukupan tok programa. Dijagrami klase vam pokazuju međusobne relacije između objekata koje program koristi. Dijagrami sekвенце mogu razjasniti detalje o tome u kakvoj su interakciji objekti.

Use Case dijagramom predstavljate svoj program kao veliki pravougaonik. Izvan programa možete staviti jednu ili više „čića Gliša“ figurica koje predstavljaju tzv. Aktere (**actors**), to jest, ljudi ili druge programe koji mogu da komuniciraju s vašim programom. Unutar pravougaonika programa se stavljuju ovali koji odgovaraju slučajevima korišćenja programa, tj. ono što akteri mogu tražiti da program uradi. Takođe možete povući linije od aktera do slučajeva korišćenja koje oni koriste da bi pokazali kojim akterima odgovaraju koje vrste slučaja upotrebe.

Dijagrami slučajeva korišćenja su izuzetno jednostavni, čak i više od drugih UML dijagrama,. Dijagram slučaja upotrebe za Pop frejmворк [1] (koristi se za razvoj igara) može izgledati kao na slici. Pošto je i pop i program i **razvojni okvir** (**framework**), njegova dijagram slučaja upotrebe ima dve vrste aktera : korisnike koji igraju igre Pop i programere koji koriste Pop okvir za razvoj nove igre. Umesto da se puno objašnjava u detalje o tome kako ćemo prikazati naše grafiku, mi jednostavno imamo ' gledaj' slučaj korišćenja da izrazi ideju da igra treba da bude priyatna za gledanje.



Slika 4.4 Dijagram slučaja upotrebe za Pop frejmворк. Izvor: Ian Sommerville, Software Engineering, 10th Global Edition, Pearson Education Publishing, 2015.

' Resize ' slučaj izražava uslov da igra treba da bude nezavisna od rezolucije ekrana. ' Podesi ' slučaj korišćenja izražava činjenicu da se zahteva sposobnost da se mogu da uradi stvari kao što je odabir nivoa igre ili resetovanje igre. ' Proveri progres ' slučaj korišćenja dovodi do zahteva da igra treba da prikaže trenutni rezultat, zdravlje igrača, i tako dalje.

Na programerskoj strani, imamo ' proširi ' i ' test ' slučaj korišćenja. Zahtevi koji slede iz ' proširi ' slučaja su da kod treba da imaju jasne, lako proširljive klase, a da su različiti numerički parametri bi mogu da bude lako pronađeni i laci za promenu. Zahtevi koji proizilaze iz ' test ' slučaja upotrebe može biti da Pop treba da ima metode za takozvane *tehnike testiranja crne kutije* (**black box testing**), kao i mod autorun u kome se igra ' igra sama '.

PRIKUPLJANJE ZAHTEVA

Zahteve treba pažljivo utrditi pre pisanja i jedne linije koda

Softverski zahtevi su više ili manje detaljan spisak želje i potreba za određenu vrstu programa. To je nešto kao pitanje ili porudžbina „Hoćeš li napisati program da uradi to i to koja izgleda tako i tako i radi na sledećim platformama ?“

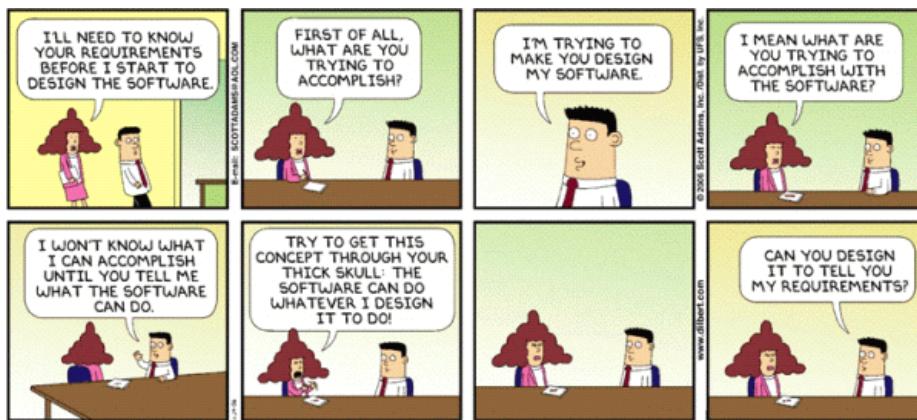
Specifikacija softvera je više ili manje detaljan opis programa. Specifikacija nastaje kao odgovor na pitanja postavljena u softverskim zahtevima. Specifikacija odgovara na zahtev strane govoreći, " Ja mogu napisati program sa sledećim karakteristikama, a njen izgled i ponašanje će biti nešto ovako.“

Kao što je pomenuto, ključna stvar je shvatiti da - kao i mnogi aspekti softverskog inženjerstva - dostizanje konačnog zahteva i specifikacija je iterativni proces. Ovo je proces koji zovemo prikupljanje zahteva (**requirements gathering**).

Neophodno je provesti dovoljno deo vremena na prikupljanju zahteva pre pisanja i jedne linije koda !

Bez prikupljanja dovoljno zahteva rizikujete da trošite mnogo energije na pisanje programa koji vaš klijent ne želi.

Da bi malo više prodiskutovali okupljanje zahteva, zamislimo situaciju gde ste vodeći softver inženjer koji treba da napravi program za nekog kupca. Kupac predlaže zahteve, vi predložite specifikaciju i pokažete je kupcu, kupac menja uslov, menjate specifikaciju, a proces se nastavlja sve dok kupac ne shvati šta on (ili ona) zaista želi, a vi ste shvatili koji od odgovora je zadovoljavajući za kupca. Ovo je primer prikupljanja zahteva.



Slika 4.5 Prikupljanje zahteva treba pažljivo uraditi. Izvor: <https://dilbert.com>

Šta o situaciji u kojoj ste se razvija program baziran na pretpostavkama, bez ikakvih investitora ili korporativnih klijenata koji su uključeni?

Pa, vi stvarno ne bi trebalo da pokuša da razvije program za imaginarnog kupca koji postoji samo u vašoj glavi. Potrebno je da izadete i razgovarajte sa drugim ljudima. Ako ste u razvoju za masovno tržište, vaš ' kupac ' mogu biti potencijalni korisnici. Ako nadograđujete postojeći proizvod, vaši korisnici mogu biti vaši budući kupci. Ako radite sami, dobro je da vaši prijatelji ili porodica budu potencijalni korisnici.

Ako ste rukovodilac kompanije, ključni stakeholder može biti rukovodilac druge kompanije koja je ugovorila da joj napišete poseban program. Ako ste zaposleni nižeg nivoa u preduzeću, vaš najvažniji stakeholder je šef.

PRIKUPLJANJE ZAHTEVA OD STAKEHOLDERA.

Zahteve treba pažljivo utrditi pre pisanja i jedne linije koda. Veoma je loša ideja pisati kod bez ikakvog pismenog plana

Ako ste rukovodilac kompanije, ključni stakeholder može biti rukovodilac druge kompanije koja je ugovorila da joj napišete poseban program. Ako ste zaposleni nižeg nivoa u preduzeću, vaš najvažniji stakeholder je šef. Ako ste student u klasi softverskih projekata, vaši stakeholderi su vaš profesor, i ostali članovi vašeg tima, a sa svakim od korisnika treba da razgovarate.

U svakom slučaju jedan od stakeholder predlaže neke manje ili više nejasne zahteve i na vama je da dođete do specifikaciju programa tako da

(a) svi akteri se slažu da specifikacija zadovoljava zahteve i

(b) vi osećate može da završite program s obzirom na raspoloživo vreme, troškove, i kvalitet.

Uslov (b) znači da u fazi prikupljanja zahteva vam imate na umu trougao ograničenja. Greška je ako obećavate da ćete biti u mogućnosti da uključi ogromnu listu fensi funkcija. Uvek se setite trougao troškova, vremena i kvaliteta. Ukoliko postoje neke apsolutno neophodne karakteristike koje su izvan temena kvaliteta u trouglu, proverite da li ste dobili naknade za troškove i / ili vremenske produženja kao nadoknadu. Ako vaš kupac, ili vaš šef, ili vaše

marketinški odeljenje, neće prihvati realni set funkcija, polako krenite u potragu za novim kupcem ili novim poslom. Previše je stresno raditi na projektu koji je osuđen od početka na neuspeh usled nerealnih procena.

Kao što smo rekli, prikupljanje zahteva treba da rezultuje u nacrt specifikacije softvera ili **SRS** (*Software Requirement Specification*).

Naime, veoma je loša ideja pisati kod bez ikakvog pismenog plana. Ako imate pismeni plan, gledajući u njega možete da izbegnete od odlutajte u pravcu cilja koje nije od presudnog značaja. I kratak pisani plan čini dobru polaznu tačku za razgovor sa drugima. Otuda, zaista treba da napišemo specifikaciju pre početka kodiranja.

NACRT SPECIFIKACIJE SOFTVERA

Pre početka kodiranja treba imati napisanu specifikaciju softvera

Kako bi trebalo da izgleda specifikacija? Razlikujemo dve vrste specifikacije :

Nacrt specifikacije je kratak, privremeni dokument, i

Kompletna specifikacija je detaljnija, čistija, i više formalna.

Idealan puna specifikacija može sastojati od header fajlova za klase vašeg program i kompletno uputstvo (**User's Guide**) program. Realno, obično niste u stanju da korektno napišete kompletну specifikaciju sve dok vaš program skoro nije proradio ! Zato, za početak, zadovoljićemo se nacrtom specifikacije. Bolje išta nego ništa.

Olakšavajuće je ako prihvate da nije realno napisati kompletну specifikaciju pre nego što napišete i jednu liniju koda. U stvarnom svetu, to pomaže da se igrate nekim kodom dok razmišljate i pokušavate nekoliko stvari, dok ne steknete neku predstavu o tome šta su prave mogućnosti.

Ovo je mesto gde se uvodi pojam nacrta specifikacije..

Kao se program razvija, specifikacija dobija sve više i više detalja. Neki aspekti specifikaciji ne može biti razrađen do ste napisali nekoliko verzija koda.

Specifikacija skica treba da opiše četiri osnovne oblasti:

(S1) koncept,

(S2) izgled,

(S3) kontrole i

(S4) ponašanje programa.

(S1). **Koncept** (**Concept**) navodi šta program radi, i opisuje ujedinjavanja teme programa.

(S2) **Izgled** (**Appearance**). Izgled bi trebalo da bude određen sa nekoliko crteža šta se očekujete na ekranu vašeg programa. Na veoma velikom projektu se može ići tako daleko da se urade makete softvera sa radnim modelima koji prikazuju uzorce na ekranima. Ovo se može uraditi bilo kao programsko prikazivanje slika urađenih u alatu za crtanje, ili kao rezultat brzog i pravog prototipa programa. Ali za mali projekat, olovka i crtež na parčetu papira može biti dovoljno.

(S3) **Kontrole** (**Controls**). Kontrole treba navesti rekavši kako će funkcionišati tastatura, miš, i još mnogo važnih opcija u menijima.

(S4) **Ponašanje** (**Behavior**). Opis ponašanja treba navede glavne karakteristike programa. Često je korisno da prođete kroz program i opišete kakvi bi bili odgovori na akcije korisnika u tipičnom scenariju korišćenja. Takođe, opisuje se kako će prosečan korisnik da počne

korišćenje programa, i pomenuti neke očekivane savete za uspešno korišćenje programa.

PRIMER: NACRT SPECIFIKACIJE SOFTVERA

Kompletna specifikacija je detaljniji, konzistentniji, i više formalni dokument

Pretpostavimo da je vaš zahtev je da se napiše kompjuterska igra.

Predložena specifikacija može biti zasnovana, na primer, na koncept za igru koja podseća na Pacman.

(S1) Nacrt koncepta bi takođe trebalo da uključuje ideju sa grafičkom temom koja se razlikuju od grafike PacMan-a.

(S2) Nacrt bi trebalo da uključuje crtež ekrana vaše igre.

(S3) Kontrole su prosto tasteri strelica.

(S4) Što se tiče ponašanja, nacrt bi imao neke detalje o tome koliko će biti neprijatelja, koji je skor za jedenje pilule za snagu, kakav će biti oblik labyrintha, i kako se nivoi igre razlikuju.

Naravno odgovor je kupca da takve specifikacije može da bude, " Ja želim novu vrstu kompjuterske igre, a ne klon postojeće igre. "

A tada ćete morati da pronađe način da vaša Pacman specifikacija bude više originalna, i nastaviti prikupljanja zahteva u sledećem ciklusu.



Slika 4.6 Pac Man. Izvor: <https://en.wikipedia.org/wiki/Pac-Man>

✓ Poglavlje 5

Procesi softverskog inženjerstva

ŠTA JE TO PROCES

Proces je niz postupaka koje transformišu ulaz u izlaz

Proces u datom kontekstu (softverskog inženjeringu) označava niz postupaka koje transformišu ulaz (postojeće stanje) u izlaz (rezultujuće stanje) u skladu sa zakonitostima i utvrđenim pravilima.

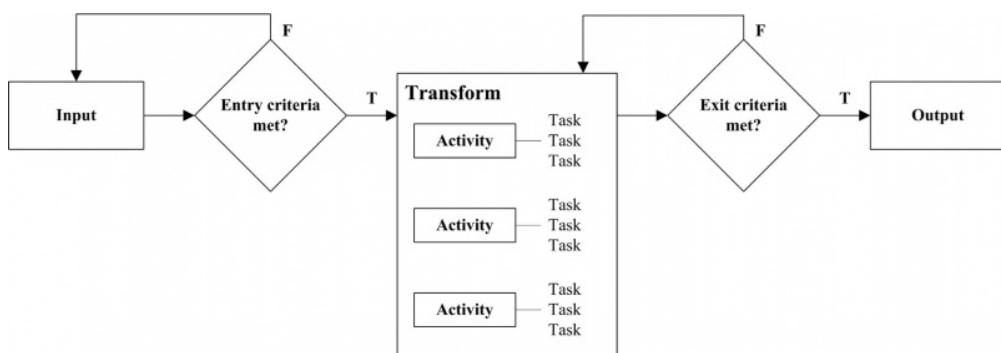


Slika 5.1 Inženjerski proces. Izvor: Autor.

Definicija procesa je slična definiciji računarskog algoritma, s tim što je pojam procesa opštiji i razumljiv ljudima, za razliku od računarskog algoritma koji treba da se izvrši na računaru.

Kao što se vidi na slici, proces može uključivati i proveru da li su zadovoljeni kriterijumi za ulazne i izlazne elemente.

Proces utvrđuje **Ko** radi, **Šta** radi, **Kada** radi i **Kako** radi (na engleskom Who, What, When i How) tokom razvoja softvera.



Slika 5.2 Proces softverskog inženjerstva. Izvor: Ian Sommerville, Software Engineering, 10th Global Edition, Pearson Education Publishing, 2015.

Projekat softvera se sastoji i od koda i od procesa kojim ste razvili kod .

Projekat softvera = kod (softver) + proces razvoja

Veoma je važno je da se formalizuje proces koji koristite pri radu na projektu, tj. razvoju softvera.

To znači da treba da imate skup dokumenata koji opisuju vaš proces , i da dovoljno često tokom projekat pogledate i revidirate dokumenata.

proces razvoja = dokumenti + provere i korekcije

Postoji mnogo načina da se razdvoje različiti aspekti procesa razvoja softvera (što je šire nego što je programiranje , testiranje , i otklanjanje grešaka u kodu) . O tome će puno više biti reči u narednim predavanjima.

PRIMER: MALI PROJEKTI I PRIKUPLJANJE ZAHTEVA

Mali projektat ne traje dovoljno dugo da se primene sve procedure iz arsenala profesionalnog menidžmenta

Mnogi od procesa razvoja softvera su veoma sofisticirani, kompleksni i – primenljivi u velikim organizacijama (Microsoft, IBM, ...). A obično velike firme ne rade mele projekte.

Izazov je uvesti proces ili proceduru razvoja softvera u mali i nezavisni tim koji nema eksperata u svakoj od oblasti, a ni projektat ne traje dovoljno dugo da se primene sve procedure iz arsenala profesionalnog menidžmenta.

Ovde ćemo navesti najendostavniji formalni proces koji je predložen za razvoj računarskih igara u malim timovima [1]. Naime, puno je tzv „Indi“ igara (indie games) razvijenih u malim timovima koje su uspešne na komercijalnom planu, a koje su očigledno rezultat uspešno vođenih projekata.

Predloženi proces se sastoji od četiri stavke:

Proces izlazi (**deliverables**)

- Zahtevi i specifikacija . dokument
- Rokovi i raspored poslova. dokument
- Dizajn . dokument
- Projektna dokumentacija . dokument

Već je bilo reči o zahtevima i specifikacijama, a o ostalim stavkama ćer biti u narednim predavanjima.

S druge strane, možda vam ovo daje ideju kako da bolje organizujete svoje projekte koje već radite.

Recimo, neka je to neka mala igra koju radite sami (ili s još nekim kolegom) koju ćete nakon završetka pokazati prijateljima.

PRIMER: MALI PROJEKTI I STAKEHOLDERI

Mali projektat nema dovoljni budžet da se primene sve procedure iz arsenala profesionalnog menidžmenta

Kao prvo, pribeležite negde šta sve hoćete da uradite, biće vam veoma korisno da se kasnije podsetite. Tim pre ako sem vas postoji još neko na projektu, biće i njemu korisna ta beleška koju ćete zvati „zahtevi“ (ili nekako drugačije), da se podseti i da ne mora vas da pita „šta smo ono hteli?“.

Sledeća beleška neka bude „rokov“. Pribeležite kad treba da bude gotovo to što radite ili bar, kad bi ste voleli da bude gotovo. Korist je da će to da podseti vas i vašeg eventualnog saradnika. A da se stvari zaboravljaju, čak i ako su važne, znate i sami.

Dizajn opišite opet u nekoj formi. Navedite ulaze, izlaze, interfejs, koje ćete algoritme

implementirati.

Sve ovo prethodno opisano može stati na par stranica u sveci.

Ako hoćete malo profesionalnije rezultate mora vam i dokumentacija biti profesionalnija. Na pr. ReadMe fajlovi i uputstvo za korisnika mogu biti neki minimum.

Ono što ostaje za vas može biti sve u jednom dokumentu (npr. Izveštaju o projektu), a koji bi trebalo da sadrži sve ono što je bitno za vas (ne i za korisnika), da bi ste podsetili šta je rađeno ili upoznali nekog ko treba da nastavi dalji posao (npr. novu verziju igre).

Skoro sve firme imaju spisak dokumenata i koraka potrebnih za izradu tih dokumenata.

Ako želite da unapredite rad na svojim projektima i ujedno proverite materiju koju izučavate o menadžmentu projekata, napravite vaš spisak dokumenta.

Probajte da ispoštujete proceduru. Ako vidite da nešto ne ide (ili može bolje) korigujte proceduru i nastavite po njoj.

Tako rade i veliki (IBM, MS,...).

▼ Poglavlje 6

Pokazna vežba: Vaš prvi PM zadatak u industriji

URAĐENI PRIMER: SPECIFIČNOSTI PM INDI STUDIJA ZA VIDEOIGRE

O menadžmentu u industriji igara se puno toga može naći na internetu

Zamislite da ste u sledećoj situaciji.

Vi ste na probnom radu u novoosnovanom studiju za izradu igara, a vaš zadatak je da pomognete direktoru u razradi metoda menadžmenta softverskih projekata i njihovoj primeni na proces razvoja videoigara u studiju.

Vaš prvi zadatak je da se upoznate sa specifičnostima razvoja softvera u malom indi (eng. Indie) timu.

Zaduženi ste da pogledate sledeći video

How To Run A Game Studio, <https://www.youtube.com/watch?v=r806kY1gz3E>

i da napišete kratak Word dokument (do 1 strane) o sadržaju videa i korisnim preporukama za vaš dalji rad.

Zatim izvršite pretraživanje interneta sa ključnim rečima „Project Management Simulation“, sa ciljem da nađete neki softverski alat (po mogućству besplatan) koji bi se mogao koristiti u studiju za obuku vođa timova i njihovih pomoćnika u menadžmentu.

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

ZADATAK ZA SAMOSTALNI RAD (45 MIN)

Napravite analizu prednosti i nedostatka preporuka kako da uvedete metode PM-a u vaš indi tim

Izvršite pretraživanje interneta sa ključnim rečima „Project Management Simulation“, sa ciljem da nađete neki softverski alat (po mogućству besplatan) koji bi se mogao koristiti u studiju za obuku vođa timova i njihovih pomoćnika u menadžmentu

Pripremite kratak izveštaj za svoje kolege iz tima na temu šta ste našli na internetu od korisnih alata i preporuka kako da uvedete metode PM-a u vaš indi tim.

Izveštaj bazirajte na pretraživanje interneta sa ključnim rečima „Project Management Simulation“, i rezultatima koje ste našli na internetu.

U izveštaju (obima oko 1 strane) navedite linkove i vaš kratak komentar šta taj link sadrži i zašto će vam koristiti.

Izveštaj treba da sadrži naslov, vaše ime i sadržaj koji odgovara zadatku koji ste dobili.

Mejtom ga pošaljite asistentu, stavljajući nastavnika u Cc:

Ako imate kolege koji bi mogli biti zainteresovani za vaš izveštaj, pošaljite im ga i zatražite mišljenje.

Ukoliko ste zadovoljni onim što vam odgovore (a i ako niste) pošaljite te odgovore-komentare asistentu.

URAĐENI PRIMER: SIMULTRAIN I PM GAME (45 MIN)

Ozbiljan alat i edukativna igra za podučavanje upravljanju projektima

Posetite sajt SimulTrain alata.

Ovo je ozbiljno rešenje, koncipirano kao edukativna igra, za podučavanje o upravljanju projektima. U pitanju je online alat koji omogućava ljudima da unaprede veštine timskog rada, organizacije i liderске veštine, tako što simulira realno okruženje za rad na određenim tipovima projekata.

Poseduje interesantan korisnički interfejs i adekvatne kontrole za simulaciju upravljanja projektom. Namenjen je kako trenerima, tako i početnicima u sticanju menadžerskih veština. Ovaj alat nije besplatan, ali na sajtu postoji video o korišćenju i može se isprobati demo okruženje.

The Project Management Game: <http://thatpmgame.com/>

Ovo je igra, vrlo jednostavnog korisničkog interfejsa, ali veoma interesantna, realistična i drži pažnju. Igra korisniku na početku daje raspoložive ljudske resurse, nasumično izabrane, sa opisom njihovih mogućnosti i efektivnosti u radu. Dat je i predviđen budžet na projektu, aktivnosti i dužina trajanja aktivnosti. Na igraču je da pažljivo odluči kom članu tima će dodeliti koji zadatak, srazmerno njegovim mogućnostima, tako da se projekat završi na vreme i u okviru datog budžeta. Kada se igra pokrene, igra simulira ono što je igrač prosledio kao ulaze i na kraju izveštava igrača o uspešnosti njegovog planiranja projekta.

Igra je besplatna, dostupna je online, i svaki put kada se pokrene da drugačije resurse. Dobra je za vežbanje razmišljanja i sticanje veština koje su potrebne za dobru procenu.

2 Zadaci za samostalni rad

Isprobajte igre za simulaciju upravljanja projekatima

1. Posetite sajt SimulTrain alata. Proucite video o korišćenju i demo koji je dostupan za korišćenje. Iznesite svoje mišljenje i zaključke o ovoj edukacionoj igri.

2. Posetite veb adresu PM game igre. Pokrenite igru, rasporedite resurse i proverite rezultate. Dajte priliku svakom kolegi da uradi isto. Izvedite zaključke o uspešnom rešavanju igre.

3. Predložite kakav biste vi alat ili edukativnu igru napravili kao pomoć za sticanje veština o upravljanju projektima. Iznesite viziju takvog softverskog rešenja. Napišite je u izveštaju (obima oko 1 strane). Izveštaj treba da sadrži naslov, vaše ime i broj indeksa i sadržaj koji odgovara zadatku koji ste dobili.

Mejtom ga pošaljite asistentu, stavljajući nastavnika u Cc:

▼ Poglavlje 7

Zaključak

ZAKLJUČAK

Rezime lekcije

Menadžment softverskih projekata ima zajedničkog sa ostalim projektima ali i specifičnosti. Izložen je najpre opšti tok razvoja, primenljiv na sve projekte, a koji počinje sa formiranjem tima koji definiše ciljeve i obim projekta, plan itd. i koji vodi projekat do završetka.

Nakon toga je razmatran projekat razvoja softvera sa aspekta onog šta treba uraditi da bi se dobio proizvod softver.

Osnovne etape razvoja softvera su

- zahtevi za softver
- dizajn softvera
- Implementacija
- testiranje

Postoje različite metodologije razvoja softvera koje imaju i više etapa, ali sve one sadrže ovu navedenu, jer ne postoji kraća procedura (a da je efikasna).

Izložen je najjednostavniji proces menadžmenta primenljiv na veoma male projekte razvoja softvera, a koji se sastoji u beleženju najbitnijeg:

- Zahtevi za softver
- rokovi i raspored poslova
- opis dizajna
- lista potrebnih dokumenata

Pri razvoju treba poštovati proceduru. Po potrebi, korigujte proceduru i nastavite po njoj.

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

REFERENCE

Korišćena literatura i linkovi

[1] Rudy Rucker, Software Engineering and Computer Games, Addison-Wesley, 2002

[2] Ravi Varma Thumati, Software Development Life Cycle Models April 22, 2009,
<https://rthumati.wordpress.com/2009/04/page/20/>

[3] Simplified project management http://www.consulting.ky/simplified_project_management.php

[4] Business simulation game <https://edu-simulation.com/promo/?adwords>

[5] The Project Management Game <http://thatpmgame.com/>



SE325 - UPRAVLJANJE PROJEKTIMA RAZVOJA SOFTVERA

Osnove upravljanja projektima

Lekcija 03

PRIRUČNIK ZA STUDENTE

SE325 - UPRAVLJANJE PROJEKTIMA RAZVOJA SOFTVERA

Lekcija 03

OSNOVE UPRAVLJANJA PROJEKTIMA

- ✓ Osnove upravljanja projektima
- ✓ Poglavlje 1: Osnove menadžmenta projekata
- ✓ Poglavlje 2: Pojam projekta
- ✓ Poglavlje 3: Pojam menadžment projekta
- ✓ Poglavlje 4: Ograničenja projekta
- ✓ Poglavlje 5: Upravljanje ograničenjima projekta
- ✓ Poglavlje 6: Pokazna vežba – Metodologije PM
- ✓ Poglavlje 7: Domaći 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

Što je projekat veći i složeniji to je potrebno više znanja i primene tehnika upravljanja projektima

Na prethodnom izlaganju smo rezimirali intuitivna shvatanja pojma projekta i upoznali sa nekim novim faktima koji će vam pomoći u daljem savladavanju materije upravljanja softverskim projektima, koja se primenjuje u poslovnom, odnosno profesionalnom svetu.

Naime, rekli smo da je projekat neki poduhvat u kome se za određeno vreme vrši rad i troše sredstva da bi se postigao željeni cilj.

Za uspešnost projekta, tj. ostvarivanje cilja projekta je veoma važna organizacija rada i trošenja sredstava, a što se naziva upravljanje projektima ili menadžment projekata.

Što je projekat veći i složeniji, to je teže upravljati njime, pa je, sem za veoma male projekte potrebno da vođa projekta (ili menadžer projekta) voda znanjima i tehnikama upravljanja projektima.

Kako upravljati projektom je pitanje kojim su ljudi se puno bavili kroz istoriju (s obzirom na dugu tradiciju složenih projekata) i rešavali ga na različite načine, zavisno od tehnologije, politike, kulture i drugog.

Najpre ćemo pomenuti osnovne prilaze (ili učenja) koja se danas koriste u profesionalnoj praksi pri definisanju konkretnog načina upravljanja u svakoj organizaciji, a koje ćemo uglavnom koristiti ka osnovu za izlaganje materije ovog predmeta.

Nakon toga ćemo objasniti šta je (u skladu s navedenim prilazima) menadžment projekta.

Nakon toga će biti izložena jedna od osnovnih zakonitosti, tzv. trougao menadžmenta projekta, koja ima najveći uticaj na uspešnost ili neuspešnost projekta.

Na kraju, biće dat osvrt na postupak kako pri osmišljavanju projekta uskladiti trougao menadžmenta projekta.

▼ Poglavlje 1

Osnove menadžmenta projekata

OSNOVNE METODOLOGIJE MENADŽMENTA PROJEKATA

Najpoznatije metodologije menadžmenta projekata su PMBOK i PRINCE

U svetu postoji više metoda upravljanja projektima, od kojih su za opšte tipove projekta najpoznatiji PMBOK i PRINCE, dok, kad su u pitanju softverski projekti dodatne SWEBOK 2013 daje pregled postojećih tehnika menadžmenta specifičnih za razvoj softvera.

PRINCE (Projects in Controlled Environments) je metod upravljanja projektima koji je razvio UK government agency Office of Government Commerce(OGC).

PRINCE koristi engleska vlada, pa je samim tim i najviše zastupljen u Engleskoj.

PMBOK Guide (A Guide to the Project Management Body of Knowledge) je razvio Project Management Institute (PMI) USA. Ovaj metod se najčešće koristi ne samo u Americi nego i u svetu. U ovom izlaganju ćemo iskoristiti PMBOK da bi smo izložili faze projekta.



Slika 1.1 Najpoznatije metodologije su PMBOK i PRINCE. Izvor: www.amazon.com

SPECIFIČNE METODOLOGIJE MENADŽMENTA SOFTVERSKIH PROJEKATA

Specifične metodologije menadžmenta softverskih projekata opisuje SWEBOK

Metodologija koje je specifična za razvoj softvera, a koja predstavlja dodatak na tehnike PMBOK ili PRINCE (a ne zamenu), je opisana u **SWEBOK** (Software Engineering Body of Knowledge), a takođe i u referenci [3].

Ova metodologija se deli na tzv. tradicionalne metode i agilne (Agile project management), o čemu će na kasnijim predavanjima biti više reči.

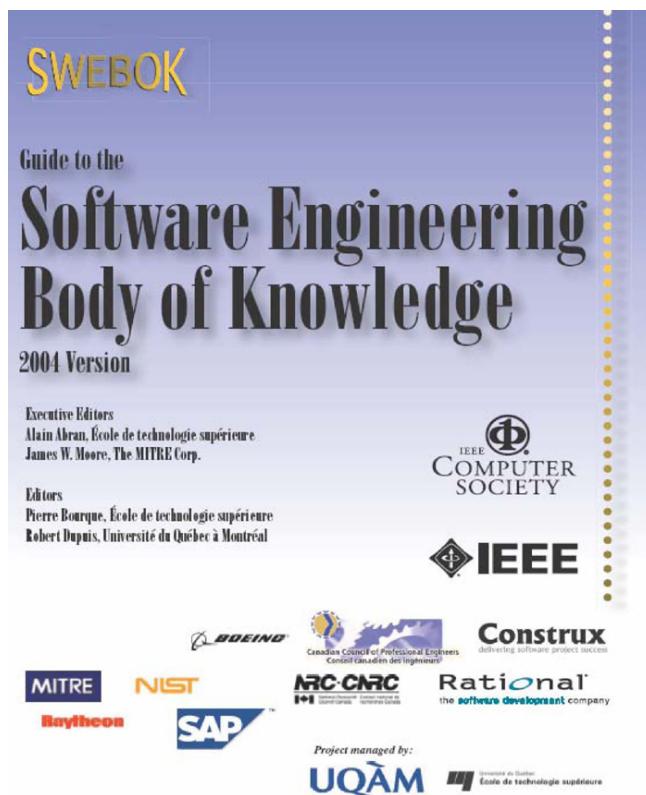
No, bez obzira na prilaz (tj. metodologiju menadžmenta) može se reći da je opšte usvojeno sledeće.

Uspeti kao menadžer projekta zahteva da završite svoje projekte na vreme, u okviru raspoloživog budžeta, i obezbedite da su vaši klijenti zadovoljni sa onim što isporuči.

To zvuči veoma prosto, ali za koliko projekti ste čuli (ili ste radili na njima) da su završeni kasno, da koštaju previše, ili ne zadovoljava potrebe svojih naručioца?

U ovom izlaganju ćemo se fokusirati na osnove upravljanja projektima, bez obzira na to kakav softver se razvija i kakvi alati i metodologija se koriste. Dakle, posmatraćemo one tehnike i svojstva upravljanja softverskim projektima koje su zajedničke za sve projekte.

U izlaganju osnovne pojmove upravljanja projektima baziraćemo se na PMBOK metodologiji.



Slika 1.2 Specifičnosti menadžmenta softverskih projekata daje SWEBOK. Izvor:
<https://www.computer.org/education/bodies-of-knowledge/software-engineering>

▼ Poglavlje 2

Pojam projekta

ŠTA JE PROJEKAT

Projekat je privremeni poduhvat preduzet da stvori jedinstven rezultat, proizvod ili uslugu.

Priručnik za [korpus znanja](#) upravljanja projektima (Guide to the Project Management Body of Knowledge 3. izdanje, objavljeno od strane Instituta za upravljanje projektima, 2013) - naziva se PMBOK i izgovara " pimbok " – definiše projekt kao

" Projekat je privremeni poduhvat preduzet da stvori jedinstven rezultat, proizvod ili uslugu. "

Prodiskutujmo ovu definiciju da vidimo šta je projekt i šta nije.

Prvo, **projekat je privremen**. Trajanje projekta može biti samo nedelju dana ili možda godinama, ali svaki projekt ima rok, krajnji datum do koga treba da bude završen. Vi možda ne znate kada krajnji datum u trenutku kada počinje projekt, ali znate da je negde u budućnosti i da će sigurno biti poznat kroz neko vreme.

Projekti nisu isti kao tekući poslovi, iako imaju mnogo toga zajedničkog. U tekući poslovi, kao što ime sugerije, se rade na neodređeno vreme, bez postavljenog datum završetka. Primer: većina aktivnosti računovodstvenih i odeljenja ljudskih resursa. Ljudi koji rade tekuće poslove mogu takođe upravljati projektima, na primer, direktor odeljenja ljudskih resursa za velike organizacije mogu planirati učešće na sajmu. Ipak, projekti se razlikuju od tekućih poslove jer se zna rok, kao što je datum sajma.

Dalje, **projekat je poduhvat**. Resursi, kao što su ljudi i opreme, treba da urade posao. Poduhvat sprovodi tim ili organizacija, i zato su projekti namerni, planirani događaji. Uspešni projekti se ne dešavaju spontano, prethodi im pripremaju i planiranja.

Konačno, **svaki projekt stvara kao rezultat jedinstven rezultat, proizvod ili uslugu**. Ovo se naziva predmet isporuke ([deliverable](#)) projekta i razlog zašto je projekt preduzet.

- Naučni i istraživački projekti imaju cilj postizanje novih saznanja (npr. Kako se može lečiti bolest koja se smatra neizlečivom, kako programirati računar da rešava probleme koje još uvek moraju da rade ljudi – veštačka inteligencija).
- Industrijski projekt najčešće ima cilj razvoj novog proizvoda (npr. Novi softver za inženjerska projektovanja, novi endžin za razvoj računarskih igara).
- Vladini ili investicioni projekti imaju cilj uvođenje novih usluga (npr. Poboljšanje kvalifikovanosti populacije, poboljšanje zdravstvenog stanja u nekom regionu ili čitavoj zemlji, izgradnja vodovoda itd).

Možda ste već shvatiti da je mnogo poslova koji se rade u svetu su projekti.

PROJEKTI I TEKUĆI POSLOVI

Tekući poslovi su stalni i ponovljivi dok su projekti privremeni i jedinstveni

Generalno organizacije i preduzeća ulažu rad da bi postigle postavljene ciljeve. Taj rad se može podeliti na tekuće poslove i projekte.

Osnovna razlika tekućih poslova i projekata je da su tekući poslovi stalni i ponovljivi dok su projekti privremeni i jedinstveni.

Međutim, i projekti i tekući poslovi su:

- Forme ljudskog rada
- Vrše se u ograničenim resursima (oprema i finansije)
- Planirani su, izvršavaju se i upravljeni su.

Rafinerija koja proizvodi benzin ne proizvodi jedinstven proizvod. Ideja je, u datom slučaju, je da se proizvede standardizovani proizvod, (obično ne želite da kupite benzin na jednoj pumpi koja se znatno razlikuje od benzina na drugoj pumpi). S druge strane, komercijalni avioni su jedinstveni proizvodi. Iako svi avioni Boeing 777 mogu da izgledaju isto za većinu od nas, svaki je, u stvari, veoma prilagođeno za potrebe svog kupca.

Pogledajte sledeći video na datu temu <https://www.coursera.org/lecture/project-management-basics/1-1-what-is-a-project-xRdoW>



Slika 2.1 Projekti I tekući poslovi. Izvor: Autor.

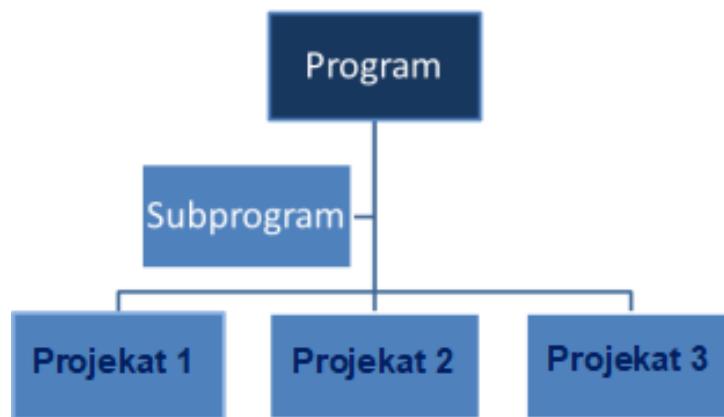
PROJEKTI, PROGRAMI I PORTFOLIO

Program je grupa projekata a portfolio je grupa programa

Pošto ćete često čuti u praksi upotrebu termina projekat, program, portfolio i o upravljanju njima, pri čemu su neki aspekti zajednički a neki se razlikuju, pogledajmo šta je šta i kakve su relacije među njima.

Projekat smo upravo objasnili.

Program je grupa projekata, a može sadržati i potprogram. Potprogram je deo programa, tj. sadrži neke od projekata koje sadrži i program.



Slika 2.2 program je skup projekata. Izvor: Autor.

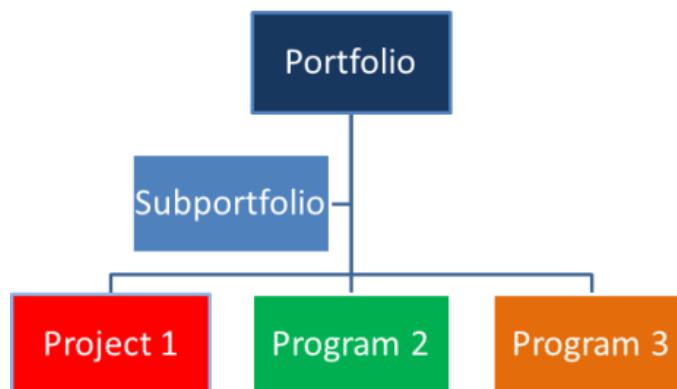
Sad je jasno zašto su neke karakteristike zajedničke a neke specifične. Projekti se razlikuju među sobom (po cilju, rokovima, resursima i drugo), ali svi imaju iste karakteristike koje važe za program, koji služi za koordinaciju povezanih projekata.

Kao primer, razvoj novog aviona je program koji se sastoji od razvoja motora, strukture aviona, hidrauličnog sistema, električnog sistema i drugog, od čega je svaki za sebe poseban projekt. Jasno je da su ovi projekti povezani i treba da budu koordinisani.

Portfolio je grupa programa, a može sadržati i potportfolio kao i individualne projekte. Potportfolio je deo portfolija, tj. sadrži neke od programa koje sadrži i portfolio.

Ključna reč za portfolio (tj. karakteristika za sve programe) je "strategijski cilj". Svi programi unutar portfolija treba da imaju isti strategijski cilj.

Kao primer, ako je razvoj jednog aviona program, tada je razvoj poljoprivrednih aviona portfolio. Na nivou vazduhoplovne organizacije mogu postojati portfolio lаких aviona, portfolio transportnih aviona, portfolio vojnih aviona itd.



Slika 2.3 Portfolio je je grupa programa. Izvor: Autor.

✓ Poglavlje 3

Pojam menadžment projekta

ŠTA JE TO MENADŽMENT PROJEKTA

Upravljanje projektom je usmeravanju poslova na projektu da bi se zadovoljili zahtevi postavljenih pred projekat

Upravljanje projektima (project management) je priznata profesija od oko 1950, ali project management u nekom obliku je morao da postoji otkada su ljudi počeli da rade složeni posao. Kada su građene velike piramide u Gizi u Egiptu, neko negde je bio zadužen za praćenje resursa, izrade rasporeda aktivnosti i specifikacije na neki način.

Upravljanje projektom je primena znanja, veština, alata i tehnika u usmeravanju aktivnosti (poslova) na projektu da bi se zadovoljili zahtevi koji su postavljeni pred projekat.

Ova primena znanja, veština, alata i tehnika se vrši kroz više faza ili upravljačkih procesa .

- Iniciranje projekta (*initiating*)
- Planiranje projekta (*planning*)
- Realizacija projekta (*executing*)
- Praćenje, nadgledanje toka projekta (*monitoring, controlling*)
i upravljanje - donošenje odluka
- Završavanje projekta (*closing*)



Slika 3.1 Upravljanje poslovima na projektu je menadžment. Izvor: Autor.

UPRAVLJAČKI PROCESI PROJEKTA

Upravljanje projektom se vrši kroz više faza ili upravljačkih procesa i u više oblasti



Slika 3.2 Upravljački procesi projekta. Izvor: Autor.

Tokom upravljanja projektom se vrše sledeće aktivnosti (kao deo navedenih procesa)

- Utvrđivanje zahteva za proizvod
- Postavljanje jasnih i ostvarljivih ciljeva
- Usklađivanje zahteva za kvalitetom proizvoda sa obimom poslova, troškova i raspoloživog vremena, ili preciznije: Obima projekta (**scope**), Kvaliteta (**quality**), Rokova (**schedule**), Budžeta (para) (**budget**), Resursa (ljudi, opreme) (**resources**) i Rizik (**risks**),
- Usklađivanje specifikacija, planova i načina rada sa očekivanjima subjekata zainteresovanih za projekat (**stakeholders**).

Možda vam nisu svi od uvedenih pojmove jasni, no u sledećim poglavljima i predavanjima će biti detaljnije objašnjeni i prodiskutovani.

ZAŠTO JE MENADŽMENT PROJEKTA VAŽAN

Ne retko se projekti vode "iz glave" sa veoma oskudnim beleškama

Kod nas (pa i u svetu) nije retkost da se poslovi sklapaju u kafani, a da to što je dogovoreno zapiše na parčetu papira ("notes scribbled down on the back of the napkin at lunch") ili uopšte i ne zapiše da bi se još više uprostila stvar.

Nakon toga se izdaje zadatak timu za realizaciju "uradite to i to" ("just do it."), i izvršioci se lačaju posla.

Na kraju rezultati projekta nisu zadovoljavajući za naručioca i on neće da plati.

Umesto da uzme novi posao, firma mora da prepravlja ono što je već uradila, i da se nada da će imati dovoljno para za plate do zavšetka i naplate.

Na neki način, ako ljudi već tako rade nije bez osnova. Uvođenje dodatnih poslova (npr. pisanje dokumentacije, razrada planova, itd.) dodatno komplikuje projekat i poskupljuje ga. No, to svakako nije ono što se naziva dobrom organizacijom (rukovođenjem) projektom a za organizatora projekta svakako nije mesto gde treba početi sa štednjom.

Nažalost, mnogi projekti u svetu, i to u profesionalnom svetu se rade ovakvom "metodologijom".

Pogledajte sledeći video In Chaos We Trust <https://www.youtube.com/watch?v=xh7P1WNV0ME>.

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

CHAOS REPORT

Samo jedna trećina IT projekata u USA je zavšena na vreme, u okviru budžeta i sa zahtevanim karakteristikama

U Americi je izdat poznati *CHAOS Report* čiji je autor Standish Group iz Bostona, čiji su rezultati u najmanju ruku neočekivani.

Naime, u Americi se svake godine organizuje oko 175000 IT projekata sa ukupnom vrednošću od oko 250 milijardi dolara.

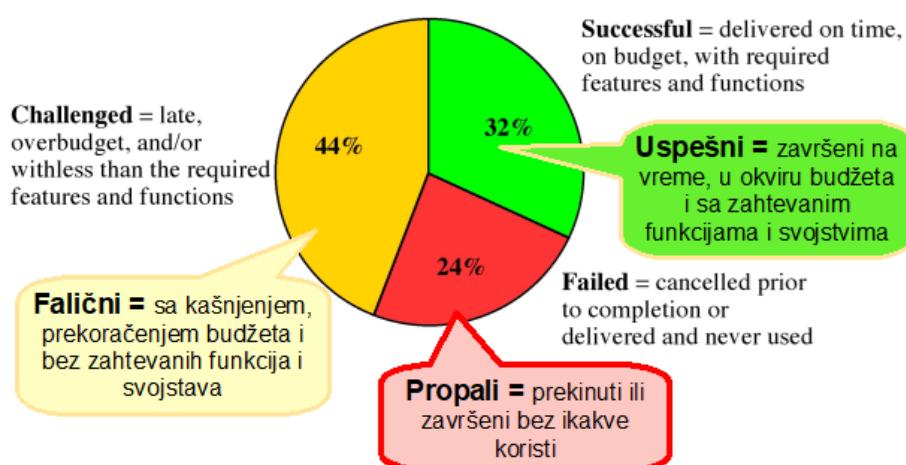
Rezultati CHAOS Report-a su sledeći:

- Oko 32% svih projekata je završeno na vreme, bez prekoračenja planiranog budžeta i sa zadovoljavajućim karakteristikama.
- oko 44% projekta je ili kasnilo, ili je prekoračilo budžet ili je završeno sa manjim funkcionalnostima i kvalitetom u odnosu na ono što je zahtevano.
- oko 24 % je obustavljeno

Otuda, jedan od glavnih ciljeva menadžmenta projekta je smanjenje rizika čime se neposredno povećava verovatnost da će projekat biti uspešno realizovan

No, dobro i profesionalno upravljanje projektima ne može potpuno eliminisati probleme na projektu, pojavu nepredviđenih događaja i izbeći svaki rizik. Pravi pristup ovom pitanju je predvideti šta se može predvideti i postići zacrtani cilj što jednostavnije i lakše.

Ipak, veliki broj projekata se može efikasnije uraditi boljom organizacijom što opravdava primenu sofisticiranih znanja i tehnika menadžmenta



Slika 3.3 Uspešnijim vođenjem projekata se mogu ostvariti velike uštede. Izvor: Autor.

▼ Poglavlje 4

Ograničenja projekta

OGRANIČENJA PROJEKTA

Rok je jedno od ograničenja projekta

Da li ste ikada radili na projektu koji je imao rok ? (Možda je bolje pitati da li ste ikada radili na projektu koji nije imao rok). Konačno vreme je jedno ograničenje jednog projekta sa kojima smo svi verovatno najbolje upoznati. Ako radite na projektu sada, pitajte članove svog tima da navedu datum završetka projekta. Oni možda ne znaju budžet projekta ili obim posla u detalje, ali šanse su oni svi znaju rok projekta.

Slede primeri vremenskih ograničenja :

- Gradite kuću i treba da završite krov pre kišne sezone.
 - Montirate veliki displej za štand na sajmu koji počinje u dva meseca.
- Za mnoge projekte koji stvaraju proizvod ili pripremaju događaj, vreme je najvažnija prepreka za upravljanje.

Često se misli o trošku jednostavno kao o novcu, ali troškovi projekta imaju šire značenje : troškovi uključuju sve resurse potrebne za realizaciju projekta. Troškovi uključuju ljude i opremu koji rade posao, koje materijale oni koriste, kao i sve druge događaje i pitanja koja zahtevaju novac ili nečiju pažnju u projektu.

Slede primeri ograničenja troškova:

- Vi ste potpisali ugovor sa fiksnom cenom za isporuku softvera za praćenje inventara klijentu. Ako vaši troškovi premašuju dogovorenu cenu, vaš klijent može da vas razume, ali verovatno neće biti spremni da ponovo pregovara o ugovoru.
- Predsednik vaše organizacije traži da sprovede projekat istraživanja kupaca koristeći samo osoblje i opremu u svom odeljenju.
- Primili ste \$ 5,000 da uradite javnu instalaciju. Nemate drugih sredstava Za gotovo sve projekte, cena je na krajnje ograničenje. Veoma malo projekata može da ide preko budžeta, uz eventualne korektivne akcije.
Umesto troškova se često koristi budžet.

TROUGAO PROJEKTA

Projekti treba da budu realizovani na vreme, u predviđenom obimu i u okviru predviđenog budžeta

Svet u kome živimo je ograničen, ili bar jeste ono što znamo vidimo i čujemo. Otuda i projekat ima svoja ograničenja. Jedno ograničenje je već pomenuto pri definiciji projekta, a to je vreme trajanja. Štaviše, vreme izvršavanja projekta je unapred zadato i često se umesto termina vreme koristi rok.

Druga ograničenja na projektu su takođe pomenuta (**Obima projekta, Kvalitet, rokovi, budžet, resursi i rizik** tj. na engleskom **scope, quality, schedule, budget, resources, risks**).

No pre nego što predemo na detaljnu analizu svih faktora, krenimo od odomaćene izreke u engleskom da projekti treba da budu realizovani "on time, on scope, on budget" tj. u zadatom roku, u okviru predviđenog obima i predviđenog budžeta. Ova izreka ukazuje ne samo da su ovo najbitniji faktori (iako je pogrešno misliti da ostali nisu bitni), već i da postoji veza između njih.

Možete da vizuelno predstavite rad na projektima na mnogo načina, ali je omiljena metoda je ono što se ponekad naziva trougao projekta ili trougao trostrukih ograničenja.

Ova tema ima mnogo varijacija, ali osnovni koncept je da svaki projekat ima neki element vremenskog ograničenja, ima neku vrstu budžeta, a zahteva neku količinu posla da se završi (drugim rečima, ona ima definisanu obim ili **scope** na engleskom).. Razmotrimo ova ograničenja jednu po jednu.



Slika 4.1 Trougao projekta. Izvor: Autor.

VREME I TROŠKOVI (BUDŽET) PROJEKTA

Slede primeri vremenskih ograničenja i ograničenja traškova.

Da li ste ikada radili na projektu koji je imao rok ? (Možda je bolje pitati da li ste ikada radili na projektu koji nije imao rok). Konačno vreme je jedno ograničenje jednog projekta sa kojima smo svi verovatno najbolje upoznati. Ako radite na projektu sada, pitajte članove svog tima

da navedu datum završetka projekta. Oni možda ne znaju budžet projekta ili obim posla u detalje, ali šanse su oni svi znaju rok projekta.

Slede primeri vremenskih ograničenja :

- Gradite kuću i treba da završite krov pre kišne sezone.
 - Montirate veliki displej za štand na sajmu koji počinje u dva meseca.
- Za mnoge projekte koji stvaraju proizvod ili pripremaju događaj, vreme je najvažnija prepreka za upravljanje.

Često se misli o trošku jednostavno kao o novcu, ali troškovi projekta imaju šire značenje : troškovi uključuju sve resurse potrebne za realizaciju projekta. Troškovi uključuju ljude i opremu koji rade posao, koje materijale oni koriste, kao i sve druge događaje i pitanja koja zahtevaju novac ili nečiju pažnju u projektu.

Slede primeri ograničenja troškova:

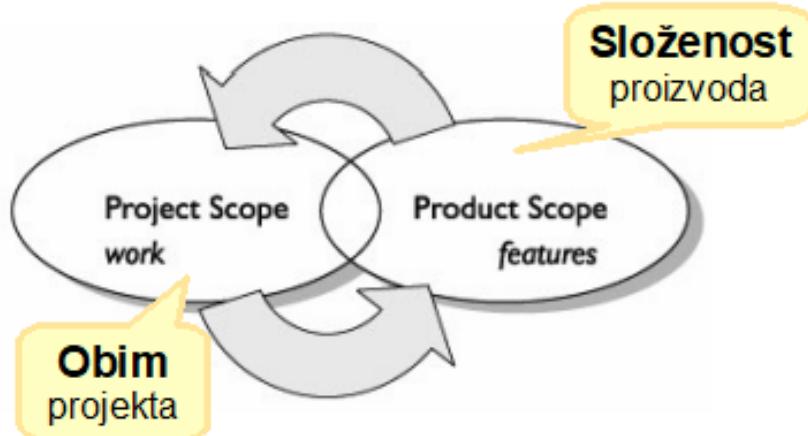
- Vi ste potpisali ugovor sa fiksnom cenom za isporuku softvera za praćenje inventara klijentu. Ako vaši troškovi premašuju dogovorenu cenu, vaš klijent može da vas razume, ali verovatno neće biti spremni da ponovo pregovara o ugovoru.
- Predsednik vaše organizacije traži da sprovede projekat istraživanja kupaca koristeći samo osoblje i opremu u svom odeljenju.
- Primili ste \$ 5,000 da uradite javnu instalaciju. Nemate drugih sredstava Za gotovo sve projekte, cena je na krajnje ograničenje. Veoma malo projekata može da ide preko budžeta, uz eventualne korektivne akcije.
Uместо troškova se često koristi budžet.

OBIM (SCOPE) PROJEKTA

Obim može značiti količinu posla ili projekta ili složenost proizvoda

Treba razmatrati dva aspekta obima : svojstva („obim“) proizvoda i obim projekta (**product scope and project scope**). Svaki uspešan projekat daje jedinstven proizvod: konkretni element ili usluga. Kupci obično imaju određena očekivanja o karakteristikama i funkcijama proizvoda.

Svojstva proizvoda (product scope) opisuje željeni kvalitet, karakteristike i funkcije proizvoda, često do najsitnijeg detalja. Dokumenti koji opisuju ove informacije se ponekad nazivaju specifikacije proizvoda. Servis ili događaj takođe obično imaju neke očekivane karakteristike. Svi imamo očekivanja o tome šta ćemo uraditi, ili videti na žurci, koncert, ili sportskog događaja.



Slika 4.2 Obim posla i složenost proizvoda. Izvor: Autor.

Obim projekta (**project scope**), s druge strane, opisuje rad koji treba da isporuči proizvod ili uslugu sa željenim svojstvima proizvoda. Obim projekta se obično meri u zadacima (poslovima) i fazama.

Slede primeri obima kao ograničenja projekta:

Vaša organizacija je dobila ugovor da razvije automobilsku proizvod koji ima tačne uslove - na primer, fizičke dimenzije merene na 0,01 mm. Ovo svojstvo proizvoda (**product scope**) je ograničenje koje će uticati na obim projektnih planova.

Gradite zgrade na parceli koja ima ograničenje visine od 15 metara.

Možete da koristite samo interne usluge da razvijete deo vašeg proizvoda, a te usluge prati metodologiju razvoja proizvoda koji se razlikuje od one koju ste planirali.

Svojstva proizvoda (**product scope**) i obim projekta (**project scope**) su usko povezani. Rukovodilac projekta koji dobro upravlja okviru projekta takođe moraju shvatiti svojstva proizvoda ili mora znati kako da komuniciraju sa onima koji rade.

Jasno je da su pojmovi „troškovi“ i „obim“ povezani, u smislu da to što treba da se uradi (obim) košta toliko i toliko (troškovi).

▼ Poglavlje 5

Upravljanje ograničenjima projekta

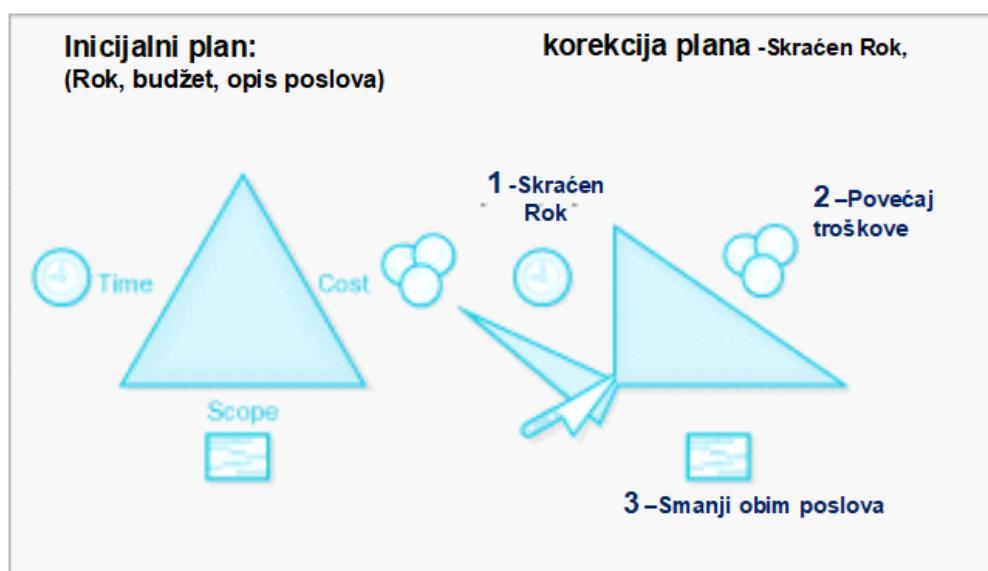
UPRAVLJANJE OGRANIČENJIMA PROJEKTA (VREME, TROŠKOVI, OBIM)

Vreme, troškovi i obim projekta su zavisne veličine. Ako se skraćuje rok rastu troškovi i obim posla

Upravljanje projektima postaje najzanimljivije kada morate balansirati ograničenja svog projekata tj. vreme, troškove i obim. Trougao projekta ilustruje proces balansiranja ograničenja, jer su tri strane trougla povezane. Promena jedne strane trougla utiče bar na jednu od druge dve strane.

Slede primeri ravnoteže ograničenja.

Ako se trajanje (vreme) iz vašeg rasporeda aktivnosti projekta smanjuje, možda ćete morati da se poveća budžet (COST), jer morate zaposliti više resursa da rade isti posao za manje vremena



Slika 5.1 Balans ograničenja u trouglu projekta ako se smanjuje trajanje . Izvor: Autor.

- Ako ne možete da povećate budžet, možda ćete morati da smanji obim jer resursi koje imate ne mogu da završi sve planirane posla za manje vremena.
- Ako morate da smanjite trajanje nekog projekta, uverite se da ukupan kvalitet projekta nije slučajno opao.

Na primer, ispitivanje i kontrolu kvaliteta često stavlja poslednji u projektu razvoja softvera, a ako se trajanje projekta smanjuje u kasnoj fazi projekta, upravo poslovi kontrole kvaliteta

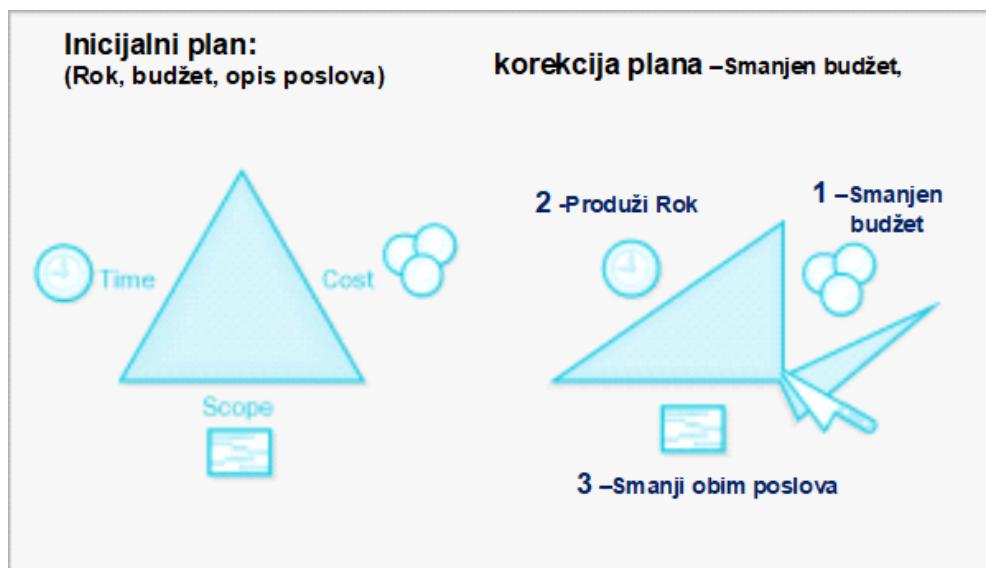
mogu biti oni koji sa smanjuju ili potpuno izbacuju. Posledice mogu biti veoma neprijatne. Morate vagati prednosti smanjenja trajanja projekta protiv potencijalnog mana isporuke proizvoda slabijeg kvaliteta.

BALANS OGRANIČENJA U TROUGLU PROJEKTA - BUDŽET

Ako smanjujete troškove projekat će duže trajati

• Ako se budžet (troškovi) vašeg projekta smanjuje, možda će biti potrebno više vremena, jer ne možete da platite za onoliko resursa koliko je potrebno ili za resurse sa potrebnom efikasnošću.

Ako ne možete povećati vreme, možda ćete morati da smanjite obim projekta, jer je manje sredstava ne možete da završite sve planirane radove u raspoloživom vremenu.



Slika 5.2 Balans ograničenje u trouglu projekta ako se smanjuje budžet. Izvor: Autor.

. Ako morate da smanjite budžet nekog projekta, možete da pogledate nivo materijalnih resursa za koje je budžet predviđen.

Na primer, da li ste planirali da snimite 35 mm film a da u iste svrhe može da posluži jeftiniji digitalni video?

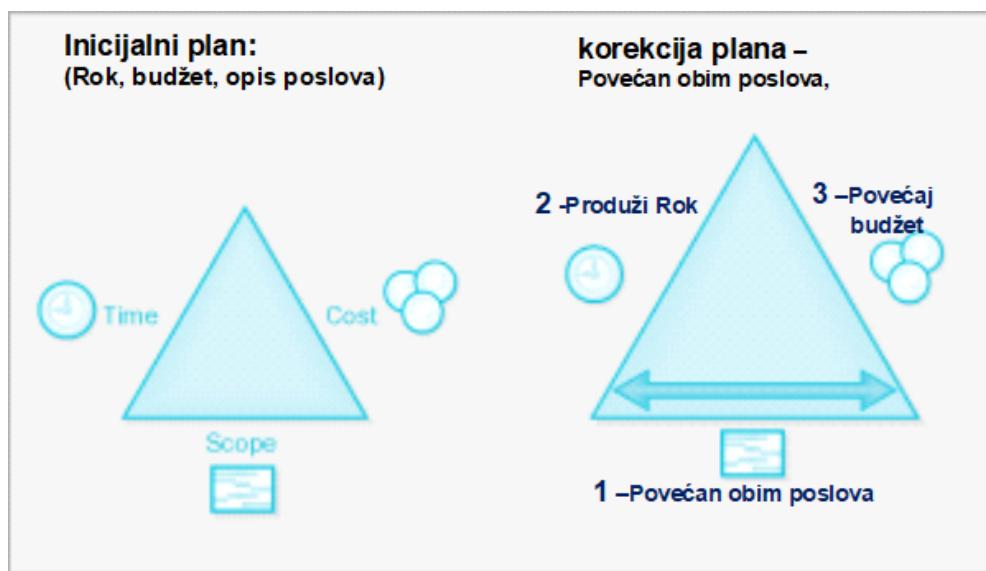
Niži - nivo materijala nije nužno niži kvalitet materijala. Dokle god je nivo materijala je pogodan za planiranu upotrebu, on i dalje može da bude visokog kvaliteta. Kao drugi primer, brza hrana i ekskluzivni restorani su dva razreda restorana hrane, ali možete u svakom naći primere visokog i niskog kvaliteta.

Takođe bi trebalo da pogledate troškova ljudskih resursa i opreme koju ste planirali da koristite. Da li možete da zaposlite manje iskusne ljude da za manje novca da izvrše jednostavnije zadatke?

BALANS OGRANIČENJA U TROUGLU PROJEKTA - OBIM

Ako se povećava obim projekta, možda će biti potrebno više vremena ili resursa

Smanjenje troškova projekta može dovesti do siromašnijeg kvaliteta proizvoda. Kao menadžer projekta, morate uzeti u obzir (ili, verovatnije, da komunicirate sa donosiocima odluka) o prednostima u odnosu na rizike smanjenja troškova.



Slika 5.3 Balans ograničenja u trouglu projekta sa povećanim obimom poslova. Izvor: Autor.

Ako se povećava obim projekta, možda će biti potrebno više vremena ili resursa (COST) da završite dodatne poslove.

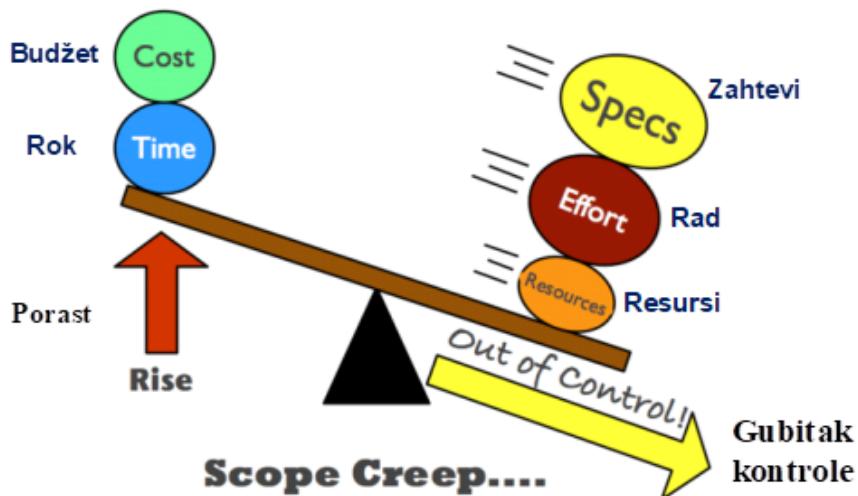
Kada se u okviru projekta desi povećanja obima nakon što je projekat počeo, to se zove "puzanje" obima (Scope Creep / vidi sledeći slajd).

„KLIZANJE“ OBIMA (SCOPE CREEP)

Dodavanjem novih zahteva se povećava vreme, obim posla i troškovi projekta

Promena obima projekta na pola projekta nije nužno loša stvar, na primer, sredina u kojoj će vaš proizvod raditi se možda promenila ili je su uslovi rada jasniji u odnosu na početak projekta.

Promena obima projekta je loša stvar samo ako manager projekta ne prepozna i planira nove zahteve koji se postavljaju - tj. ako druga ograničenja (troškovi, vreme) nisu odgovarajuće ispitani i, ako je potrebno, prilagođeni.



Slika 5.4 Dodavanje novih zahteva preti da ugrozi uspešnost projekta. Izvor: Autor.

Vreme, cena (budžet) projekta, i obim su tri osnovne elemente svakog projekta. Da bi uspeli kao menadžer projekta, trebalo bi da znate dosta o tome kako sva tri ograničenja važe za vaše projekte.

Ovde je naša konačna reč o modelu trougla projekta. Kao i sve jednostavne modele složenih predmeta, ovaj model je koristan alat za učenje, ali ne uvek odraz stvarnog sveta.

Ako se pravi projekti uvek izvode kako sugeriše trougao projekta, možda ćete videti projekte završene kasno ali po planiranoj ceni ili sa očekivanim svojstvima. Ili, projekti mogu biti završeni na vreme i sa očekivanom obimu, ali po višoj ceni.

Drugim rečima, možete očekivati da vidite da bar jedan element trougla projekta izgleda kao što je planirano.

Ali tužna istina je da mnogi projekti, čak i sa rigoroznim nadzorom upravljanja projektima, dostavljaju kasno, preko budžeta, i sa daleko manje od očekivanog obima funkcionalnosti.

Možda ste i sami učestvovali u takvim projektima.

Nije tajna, project management je jednostavno težak.

Uspeh u upravljanju projektima zahteva retku kombinaciju veština i znanja o rasporedu praksi i alata, kao i veštine u oblasti ili industriji u kojoj se projekat izvršava.

▼ Poglavlje 6

Pokazna vežba – Metodologije PM

URAĐENI PRIMER: BIZNIS SIMULACIONE IGRE ZA MENADŽMENT (40 MIN)

Obuka i trening projekt menadžera se može vršiti primenom simulacionih igara

Zamislite da ste u sledećoj situaciji.

Vi ste na probnom radu u novoosnovanom studiju za izradu igara, a vaš zadatak je da pomognete direktoru u razradi metoda menadžmenta softverskih projekata i njihovoj primeni na proces razvoja video igara u studiju.

Pošto ste uspešno rešili prethodne zadatke koje ste dobili, sada treba da ocenite svrshishodnost uvođenja treninga za zaposlene primenom biznis simulacione igre za menadžment.

Najpre pogledajte sledeći video

Business Simulation Games for Managers & Employees, <https://www.youtube.com/watch?v=4-iVwK41RRA>

i napišite izveštaj (kratak Word dokument do 1 strane) o sadržaju videa i korisnim preporukama za vaš dalji rad.

Izvršite pretraživanje interneta sa ključnim rečima „Business Simulation Games“, sa ciljem da nađete neki softverski alat (po mogućству besplatan) koji bi se mogao koristiti u studiju za obuku vođa timova i njihovih pomoćnika u menadžmentu.

Zadatak za samostalni rad

Napravite analizu prednosti i nedostatka korisnih alata i preporuka s interneta

Pripremite kratak izveštaj za svoje kolege iz tima na temu šta ste našli na internetu od korisnih alata i preporuka kako da uvedete metode PM-a u vaš indi tim.

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

Izveštaj bazirajte na pretraživanje interneta sa ključnim rečima „Business Simulation Games“, i rezultatima koje ste našli na internetu.

U izveštaju (obima oko 1 strane) navedite linkove i vaš kratak komentar šta taj link sadrži i zašto će vam koristiti.

Izveštaj treba da sadrži naslov, vaše ime i sadržaj koji odgovara zadatku koji ste dobili.

Međutim ga pošaljite asistentu, stavljajući nastavnika u Cc:

Ako imate kolege koji bi mogli biti zainteresovani za vaš izveštaj, pošaljite im ga i zatražite mišljenje. Ukoliko ste zadovoljni onim što vam odgovore (a i ako niste) pošaljite te odgovore-komentare asistentu.

URAĐENI PRIMER: TRADICIONALNE VS. AGILNE METODOLOGIJE (25 MIN)

Upravljanje projektima razvoja softvera primenom specifične metodologije

Na predmetima usko vezanim za softversko inženjerstvo, već ste učili o metodologijama razvoja softvera. Učili ste već o tome kako se pišu zahtevi i upravlja njima na tradicionalnim projektima i dali osvrt na vođenje za zahteva na agilnim projektima. Znate koje su prakse kada se testira tradicionalni projekat, a koje revolucionarne prakse su uvele agilne metodologije kada je reč o automatizovanom i kvalitetnijem procesu testiranja.

Najbolje je kada je metodologija upravljanja softverskim projektom usaglašena sa tehničkim praksama koje članovi tima primenjuju u razvoju svog softverskog proizvoda. Zašto?

Iako postoje primeri kada timovi prave dobre rezultate mešavinom različitih pristupa, uzimajući dobre prakse iz svakog od njih, jedino su odlični, uigrani i manji timovi u stanju da kreiraju takav model za sebe. Svaki tim koji je nov u primeni neke metodologije ili je nov u razvoju softvera, preporučuje se da prati osnovne principe jedne metodologije. Kasnije jeste dobra odluka istraživati druge prakse, unapređivati sopstvenu metodologiju i prilagođavati je svojim potrebama. Kada nešto učimo ili radimo po prvi put, potrebne su jasne smernice. Kada se izveštimo i spremni smo za nova saznanja, promene i usavršavanje; onda možemo sami da kreiramo put i inoviramo sopstvena pravila.

Dobar deo metodologija je upravo i nastao kombinovanjem pravila, a onda se za ta pravila pročulo i postala su pravila dobre prakse. Na primer, sve agilne metodologije imaju dosta toga zajedničkog, razlike su u finesama i pojedinim tehničkim praksama.

Uzmimo za primer Scrum i Kanban. I jedna i druga metodologija koriste tablu sa zadacima, ali imaju drugačije cikluse izvršenja zadataka. Na kraju je nastao Scrumban kao kombinacija dobrih strana jednog i drugog pristupa.

S druge strane, možemo da kažemo da su agilne metodologije nastale zahvaljujući tradicionalnim, odnosno na njihovim temeljima. Primeričete da i kod jednih i kod drugih imamo gotovo iste faze razvoja, s tim što su drugačije raspoređene ili imaju mogućnost da se dešavaju iterativno. Agilne metodologije jesu nastale upravo kada se javila potreba da se nešto unapredi u tradicionalnom načinu rada. Korak po korak, konstantno se menja sve ono što ima nedostatke, težeći poboljšanju i savršenstvu. To ciljano savršenstvo zapravo nikada neće biti postignuto, ali važno je da postoji progres.

Isto tako je važno da imamo osnove, temelje, tj. dobre prakse u upravljanju projektima. Mnoge od njih nam daje PMBOK i za mnoge tipove ozbiljnih projekta i dalje je nezamislivo da se ne prate standardi koje propisuje ova riznica znanja za upravljanje projektima.

.....

ZADATAK ZA SAMOSTALNI RAD (20 MIN)

Napravite analizu PMBOK-a i agilnih metodologija

1. Pretražite internet i pronadite najnoviju verziju PMBOK-a. Preuzmite je i prelistajte.
2. Koje oblasti znanja ima PMBOK? U narednim lekcijama ćemo ih obrađivati. Sada, dajte

kratak pregled svakog poglavlja i podelite zaključke sa kolegama i asistentom. Napišite izveštaj (obima oko 1 strane), navedite linkove i vaš kratak komentar šta taj link sadrži i zašto će vam koristiti.

3. Dajte podelu tradicionalnih i agilnih metodologija, onaku kako ste učili do sada.
4. Koje agilne metodologije su vam poznate, u smislu kako su koncipirane da se primenjuju, a koje nisu? Zatražite od asistenta objašnjenje za one koje ne razumete.
5. O kojim metodologijama biste voleli da naučite nešto novo? Koje imate u planu da primenjujete u svom daljem radu? Koje biste voleli da primenjujete na svom poslu? Zbog čega?
6. Uzmite za primer jednu agilnu metodologiju koja vam je interesantna. Pronađite priručnik za učenje te metodologije. Istražite da li postoji nešto poput Body of Knowledge za tu metodologiju. Napišite izveštaj (obima oko 1 strane) sa novim saznanjima, navedite linkove za preuzimanje korisnog sadržaja i vaš kratak komentar šta taj link sadrži i zašto će vam koristiti.

Izveštaj treba da sadrži naslov, vaše ime i broj indeksa i sadržaj koji odgovara zadatku koji ste dobili.

Mejlom ga pošaljite asistentu, stavljajući nastavnika u Cc:

-
7. Na osnovu čega timovi i organizacije zapravo biraju kakvu će metodologiju primenjivati u upravljanju i razvoju svojih projekata? Koji faktori utiču na odlučivanje? Svesni ste da jedan skup pravila ne važi u svakoj situaciji.

Pročitajte tekst sa sledećeg linka:

<https://www.cio.com/article/3156998/agile-project-management-a-beginners-guide.html>

.....

✓ Poglavlje 7

Domaći zadatak

DOMAĆI ZADATAK #1

Domaći zadatak #1 okvirno se radi 4.5h

Domaći zadatak se radi u dogovoru sa predmetnim asistentom.

Predaja domaćeg zadatka:

Tradicionalni studenti:

Domaći zadatak treba dostaviti najkasnije nedelju dana nakon predavanja za 100% poena.
Nakon toga poeni se umanjuju za 50%.

Internet studenti:

Domaći zadatak treba dostaviti najkasnije 10 dana pred polaganja ispita. Domaći zadaci se brane!

Domaći zadatak poslati na:

Beograd:

nebojsa.gavrilovic@metropolitan.ac.rs

Niš i Internet studenti:

marina.damjanovic@metropolitan.ac.rs

▼ Poglavlje 8

Zaključak

ZAKLJUČAK

Rezime lekcije

U ovom izlaganju su navedene osnovne reference za profesionalno izučavanje menadžmenta projekta:

- PMBOK Guide (A Guide to the Project Management Body of Knowledge)

- PRINCE (Projects in Controlled Environments),

kao i za izučavanje tehnika menadžmenta specifičnih za softversko inženjerstvo

- SWEBOK (Software Engineering Body of Knowledge)

Objašnjeno je šta je projekat i šta menadžment projekta prema PMBOK:

" Projekat je privremeni poduhvat preduzet da stvori jedinstven rezultat, proizvod ili uslugu. "

"Upravljanje projektom (menadžment projekta) je primena znanja, veština, alata i tehnika u usmeravanju aktivnosti (poslova) na projektu da bi se zadovoljili zahtevi koji su postavljeni pred projekat."

Menadžment projekta se vrši kroz faze (takođe se zovu i upravljački procesi):

- Iniciranje projekta (initiating)

- Planiranje projekta (planning)

- Realizacija projekta (executing)

- Praćenje, nadgledanje toka projekta (monitoring, controling) i upravljanje - donošenje odluka

- Završavanje projekta (closing)

.....

Menadžment projekata je važan jer je:

- svaki treći projekat uspešan

- svaki treći projekat je završen uz gubitke

- svaki treći projekat je propao (nije završen)

U engleskom postoji izreka da je uspešan onaj projekat koji je završen "on time, on scope, on budget".

Osnovni parametri projekta su

- Vreme realizacije,

- obim poslova,

- raspoloživa sredstva

i ove veličine su povezane. Mogu se birati dve od njih, ali je treća uslovljena tim izborom (inače projekat nije uspešan).

U engleskom se za obim koristi reč "scope", i može imati dva različita značenja:

- project scope označava obim poslova (ili vrednost tih poslova)

- product scope označava obim funkcionalnosti i svojstava koje proizvod treba da ima, tj. meri se složenošću i kvalitetom proizvoda.

REFERENCE

Korišćena literatura i linkovi

- [1] P. Bourque and R.E. Fairley, eds., Guide to the Software Engineering Body of Knowledge, Chapter 7 Software Engineering Management Version 3.0, IEEE Computer Society, 2014; www.swebok.org. <https://www.computer.org/education/bodies-of-knowledge/software-engineering>
- [2] Project Management Institute Standards Committee, A Guide to the Project Management Body of Knowledge (PMBOK), Project Management Institute, 2000.
- [3] (PMI00) Project Management Institute Standards Committee, A Guide to the Project Management Body of Knowledge (PMBOK), Project Management Institute, 2000.
- [4] PRINCE2, <http://en.wikipedia.org/wiki/PRINCE2>
- [5] Guide to the Software Engineering Body of Knowledge (SWEBO), CHAPTER 8 SOFTWARE ENGINEERING MANAGEMENT <https://www.computer.org/education/bodies-of-knowledge/software-engineering>
- [6] http://en.wikipedia.org/wiki/Project_management
- [7] Moira Alexander, Agile project management: 12 key principles, 4 big hurdles <https://www.cio.com/article/3156998/agile-project-management-a-beginners-guide.html>