



**Prolećni semestar 2017/18
PROJEKTNII ZADATAK**

NSI – Softver za distribuiranje video igara putem interneta

AUTORI

Nemanja Kuzmanović 2851

Ivan Ilić 2636

Sava Jeremić 2733

Profesor:

Prof. dr Ljubomir Lazić

Asistent:

Boro Mijović

Beograd, 2018

Sadržaj

Sadržaj.....	1
1. Pokretanje, planiranje i realizacija projekta.....	4
1.1 Ciljevi i zadaci.....	4
1.2 Sistematski izkaz opsega.....	6
1.2.1 Opšti zahtevi.....	6
1.2.2 Poboljšanja.....	6
1.3 Kontekst sistema.....	7
1.4 Ograničenja sistema.....	8
2. Procene obima i složenosti projekta.....	9
2.1 Podaci korišćeni za procenu obima i troškova.....	9
2.2. Modeli i Tehnike primenjenih procena, rezultati obima i troškova.....	12
2.2.1 Procena veličine bazirana na Linijama Koda.....	12
2.2.2 Procena veličine bazirana na Funkcionalnoj analizi (FP).....	13
2.2.3 Procena veličine bazirana na COCOMO II modelu.....	16
2.2.4 Analiza izvršenih procena.....	21
2.3 Procena resursa projekta.....	21
2.3.1 Ljudstvo.....	21
2.3.2 Analiza troškova.....	29
2.3.3 Minimalni hardverski zahtevi.....	34
2.3.4 Minimalni softverski zahtevi.....	34
3. Procene kvaliteta, održavanja i testiranja softvera.....	35
3.1 “Rule of Thumb” metoda procene.....	35

3.2. Procena održavanja COCOMO II modelom.....	41
4. Menadžment rizika.....	42
4.1 Opseg i namera RMMM aktivnosti.....	42
4.2 Menadžment rizika organizacione uloge.....	43
4.3 Identifikacija i opis rizika.....	45
4.4 Tabela intenziteta rizika.....	48
4.5. Analiza identifikovanih rizika.....	49
5. Raspored aktivnosti projekta.....	50
5.1 Vremenski okvir.....	54
5.2 Strukturalna podela posla (WBS).....	58
6. Organizacija projektnog tima.....	60
6.1 Struktura i odgovornosti članova tima.....	60
7. Mehanizmi praćenja i kontrole.....	62
7.1 Kontrola i menadžment promena.....	62
8. Upravljanje komunikacijom na projektu.....	71
9. Zaključak.....	78
10. Literatura.....	79
11. Prilozi i alati.....	80
10.1 Prilozi.....	80
10.3 Korišćeni softverski alati.....	80

Revizija projekta

Opis	Datum predaje	Komentari
Faza 1	5/30/2018	

1. Pokretanje, planiranje i realizacija projekta

U okviru prve oblasti definisanja projekta biće definisani ciljevi i zadatci projekta, sistemski iskaz opsega, kontekst sistema, kao i ograničenja sistema.

1.1 Ciljevi i zadaci

Sam cilj projekta je izrada sistema, tj. Platforme, koja zapravo predstavlja veb aplikaciju kojom se pristupa putem interneta, i omogućava korisnicima da pretraži video igre, kao i da ih kupe, po najnižoj mogućoj ceni. Pored navedenih stavki korisnici mogu dodati video igre u svoju tzv. Listu želja iz koje ih mogu kasnije kupiti, i dobiti informacije kada su te igre na akciji. Ovaj projekat rešava problem između distributera igara i krajnjeg korisnika, tako što korisnici ne moraju fizički da kupe igre nego mogu da dođu do sadržaja digitalnim putem u jako malom vremenskom periodu.

CILJ PROJEKTA	
<ul style="list-style-type: none"> Izrada sistema koji će poboljšati poslovanje između distributera igara i krajnjeg korisnika svojim funkcionalnostima. 	
DISTRIBUCIJA PROJEKTA	
<ul style="list-style-type: none"> Platforma će biti podignuta na serveru, tako da korisnici mogu pristupiti putem interneta sa svojih uređaja u bilo kom trenutku. 	
USPEŠNOST PROJEKTA	
Zainteresovane strane – Stakeholders	Kriterijum uspešnosti
Korisnik	Očekivanja su da sajt bude odrađen na vreme, u postavljenom roku i bez dodatnih troškova. Sajt kao i interfejs treba biti intuitivan i adaptivan, odnosno lak za upotrebu kao i navigaciju. Treba biti efektivan prilikom kupovanja igrice, da ne dođe do nikakvih problema u vezi novca davanih za kupljene igrice.
Programer	Programeru treba biti zagarantovana isplata u dogovorenom roku u kojem je definisana plata. Ne sme se menjati bilo kakvi raniji dogovoreni uslovi koji su predefinisani. Treba se poštovati svi dogovori, bez ikakvih izmena ili malverzacija.

Tabela 1.1/1 - Ciljevi i zahtevi sistema

1.2 Sistematski izkaz opsega

Sistem treba da skрати vreme koje korisniku treba da izvrši operaciju kupovanja video igre, tako što umesto toga da korisnik fizički ide i da kupi igru, on je putem naše platforme dobio tu istu igru u digitalizovanom formatu. Aplikacija će biti definisana modelom arhitekture MVC (Model-View-Controller), kako bi omogućili lakše upravljanje sistemom, a kasnije lakše održavanje.

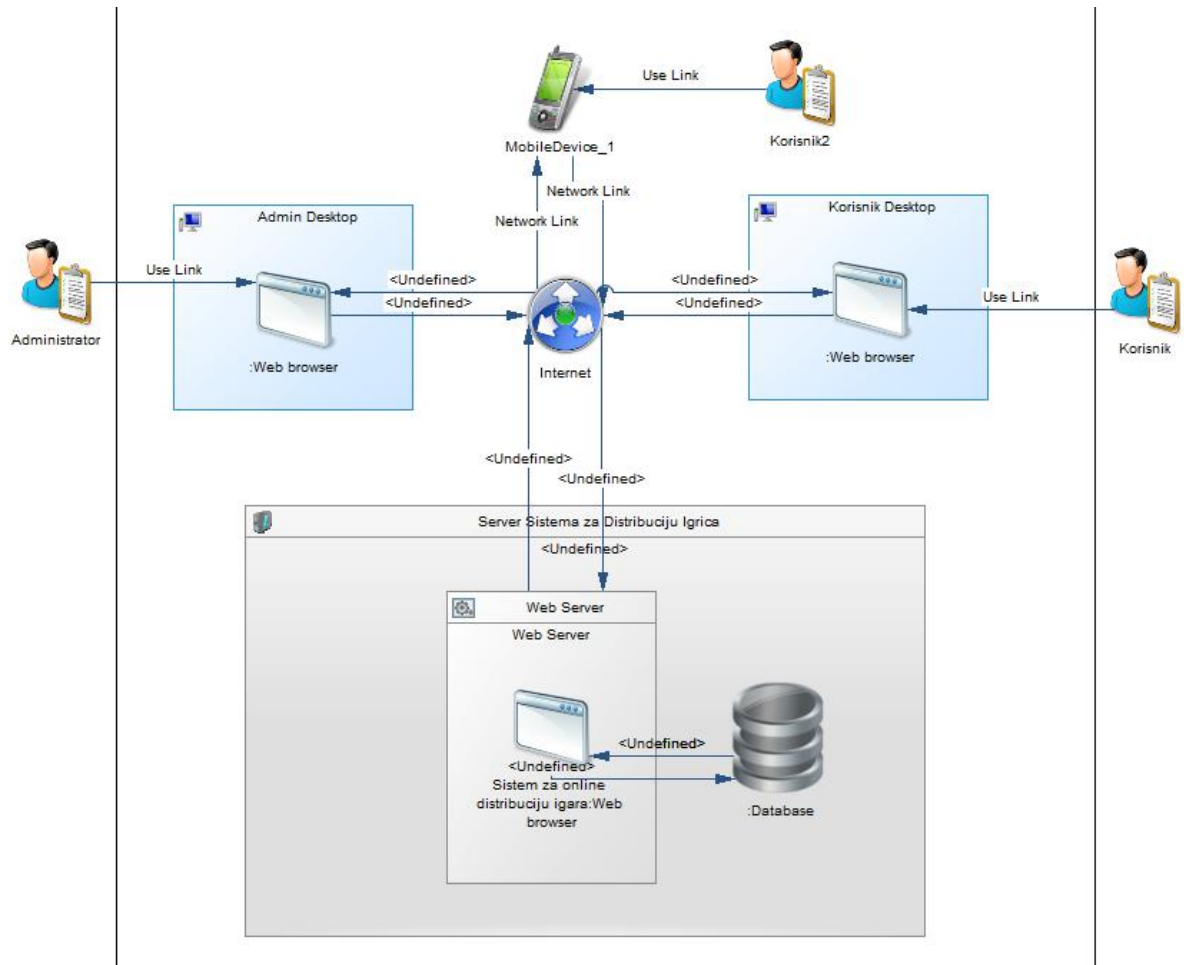
1.2.1 Opšti zahtevi

Sistem opslužuje istovremeno veliki broj korisnika. Korisnici na sistemu imaju mogućnost pregleda svih igara, pregleda konkretne igre kada otvore stranicu sa detaljima o igri, pretragu igara koje se nalaze u bazi podataka sistema, dodavanje igara u svoju listu želja, kupovinu igara za sebe, kupovinu igara za prijatelja, ocenjivanje igre, dodavanja komentara za igru, brisanje igre iz liste želja, izmenu detalja svog profila kojeg drugi korisnici sistema mogu videti, kao i dodavanje drugih korisnika u listu prijatelja. Takođe, korisnici se mogu dopisivati za drugim korisnicima sistema, kako bi obavili razmenu igara (u vidu ``kupovine za prijatelja``). Konačno, korisnici u bilo kom trenutku mogu kontaktirati administratora ukoliko dođe do problema pri kupovine igre (transakcije), podešavanju profila, interakcije sa drugim korisnicima sistema, ukoliko dođe do greške pri korišćenju ili u samom sistemu i slično.

1.2.2 Poboljšanja

Ovaj sistem uvodi poboljšanja u vezi bržeg i lakšeg izvršavanja kupovine igara, tako što korisnici ne moraju fizički da odlaze i kupuje igre, već mogu putem naše platforme u samo nekoliko klikova da dođu do najnovijih, najpovoljnijih, i najpopularnijih naslova. Takođe, platforma ima inovativan, moderan i lak za korišćenje grafički korisnički interfejs, što korisnicima omogućava brže i lakše dolaženje do željenih operacija, a takođe atraktivnim izgledom dovodi do povećanja broja korisnika. Sistem takođe može opsluživati znatno veći broj korisnika od sličnih sistema, pri čemu ne dolazi do degradacije brzine sistema.

1.3 Kontekst sistema



Slika 1.3/1 Dijagram konteksta softvera NSI

Na slici možemo videti dijagram konteksta softvera NSI. Oblast sive boje predstavlja server sistema za distribuciju video igara putem interneta, nalazi se i baza podataka koja je povezana sa serverom i koja je u konstantnoj komunikaciji sa serverom.

Korisnici interaguju sa sistemom putem interneta, mogu mu pristupiti putem internet pretraživača ili putem pametnih uređaja.

Postoji dva tipa korisnika, a to su sami korisnici sistema I administratori.

1.4 Ograničenja sistema

Pošto sistem radi putem interneta, mora se koristiti adekvatna zaštita, jer sistem koristi poverljive podatke korisnika. Sistem zbog toga koristi sigurnu HTTPS konekciju, kao i sigurne SSL sertifikate. Pored toga koristi se AES-enripcija za enkriptovanje kritičnih i ličnih korisničkih podataka, poput broja kartice, adrese i slično. Takođe, sistem treba da prikaže korisnicima ukoliko dođe do greške, čime će ih obavestiti o tome adekvatnom porukom, i navesti ih na adekvatnu akciju, a takođe će i sam sistem preuzeti adekvatne akcije, kako ne bi došlo do narušavanja integriteta sistema prilikom greške, jer to može uticati da se obavi akcija koja nije planirana ili željena, pa maliciozni korisnik je može upotrebiti u neke druge svrhe.

2. Procene obima i složenosti projekta

U okviru procene obima i složenosti projekta će biti prikazani i definisani troškovi sistema, određivanje i predviđanje prosečnog broja linija koda softverskog sistema, na osnovu nekih već definisanih standarda i modela za određivanje ove procene.

2.1 Podaci korišćeni za procenu obima i troškova

Pošto nemamo ranijih iskustava sa sličnim projektima poput ovog, a to je sistem za distribuciju igara putem interneta, oslonićemo se na istraživanja stručnih ljudi koji su već istraživali ove oblasti i definisali obim i troškove sistema. Jedan od stručnjaka koji su istraživali ovu oblast jeste Carpers Jones, i rezultate ovog istraživanja možete videti u sledećim tabelama.

Organizations	Function Points	Lines of Code	Personnel with Access	Packages Used
Internal Revenue Service	150,000	7,500,000	10,000	10
Banks	125,000	6,250,000	90,000	12
Insurance companies	125,000	6,250,000	75,000	15
Credit card companies	125,000	6,250,000	3,000	10
Credit bureaus	120,000	6,000,000	1,500	9
Census Bureau	100,000	5,000,000	1,000	5
State tax boards	90,000	4,500,000	200	5
Airlines	75,000	3,750,000	250	12
Police organizations	75,000	3,750,000	10,000	5
Hospitals	75,000	3,750,000	1,000	5
Web-based stores	75,000	3,750,000	1,500	12
Municipal tax boards	50,000	2,500,000	20	3
Motor vehicle department	50,000	2,500,000	200	3
Physicians offices	30,000	1,500,000	50	6
Dental offices	30,000	1,500,000	50	6
Schools/universities	25,000	1,250,000	125	8
Clubs and associations	20,000	1,000,000	250	3
Retail stores	20,000	1,000,000	100	4
TOTALS	1,360,000	68,000,000	194,245	133

Tabela 2.1/1 - Podaci o veličini sistema[1]

Iz tabele 1 možemo uočiti *Web-based stores* obeleženo crvenom bojom, sistem koji je sličan našem, iz te tabele možemo pročitati da projekti sličnog tipa imaju 75.000 funkcionalnih poena, dok imaju 3.750.000 linija koda.

Broj linija se računa na sledeći način:

$$\text{LinesOfCode(LC)} = \text{FunctionPoints(FP)} * 50$$

Broj 50 predstavlja veličinu sa kojom se množe funkcionalni poeni, za određeni programski jezik. Odabir programskog jezika na osnovu kojeg biramo broj, možemo videti sa sledeće tabele

Language	QSM SLOC/FP Data			
	Avg	Median	Low	High
ABAP (SAP) *	28	18	16	60
ASP*	51	54	15	69
Assembler *	119	98	25	320
Brio +	14	14	13	16
C *	97	99	39	333
C++ *	50	53	25	80
C# *	54	59	29	70
COBOL *	61	55	23	297
Cognos Impromptu Scripts +	47	42	30	100
Cross System Products (CSP) +	20	18	10	38
Cool:Gen/IEF *	32	24	10	82
Datastage	71	65	31	157
Excel *	209	191	131	315
Focus *	43	45	45	45
FoxPro	36	35	34	38
HTML *	34	40	14	48
J2EE *	46	49	15	67
Java *	53	53	14	134
JavaScript *	47	53	31	63
JCL *	62	48	25	221
LINC II	29	30	22	38
Lotus Notes *	23	21	19	40
Natural *	40	34	34	53
.NET *	57	60	53	60
Oracle *	37	40	17	60
PACBASE *	35	32	22	60
Perl *	24	15	15	60
PL/I *	64	80	16	80
PL/SQL *	37	35	13	60
Powerbuilder *	26	28	7	40
REXX *	77	80	50	80

Slika 2.1/2 - Prosečan broj linija koda(LOC) po funkcionalnom poenu(FP)[2]

Sa tabele 3 možemo videti da su JAVA i C++ zabeleženi crvenom bojom, kao jezici višeg nivoa čiji prosečan broj linija koda je približan 50 odnosno onom u formuli(u Avg koloni). C++ stručnjaci daju 50 LOC dok za Javu 53, tako da ćemo za razvoj softverskog sistema uzeti one jezike koji imaju jednaku vrednost 50 kao i približnu njoj, a to je 53 - JAVA.

(Burdened cost = \$10,000 per month)			
Function Points	Low Quality	Average Quality	High Quality
10	\$6,875	\$6,250	\$5,938
100	\$88,561	\$78,721	\$74,785
1,000	\$1,039,889	\$920,256	\$846,636
10,000	\$23,925,127	\$23,804,458	\$18,724,012
100,000	\$507,767,782	\$433,989,557	\$381,910,810

Slika 2.1/3 - Troškovi softvera po veličini i kvalitetu[1]

Troškovi softvera visokog kvaliteta kao što se može videti na slici, na osnovu prethodnih podataka o veličini softvera za Šopove bazirane na vebu (WebBased Shops), gde funkcionalni poeni iznose oko 75000, su između 18.724.012,00 dolara i 382.910.810,00 dolara, po istraživanju Carper Jones-a.

Jedina razlika je što je profesor Jones vršio istraživanja po Američkim standardima koja su daleko iznad naših u Republici Srbiji, pa su i cene rada desetostruko veće.

2.2. Modeli i Tehnike primenjenih procena, rezultati obima i troškova

Za početnu fazu projekta je bitno odrediti tačno troškove kao i pravilno proceniti očekivane rezultate.

2.2.1 Procena veličine bazirana na Linijama Koda

Formula po kojoj se računa veličina bazirana na LOC:

$$\text{LOC} = (\text{O} + 4 * \text{E} + \text{P}) / 6$$

Gde je:

- **LOC**(Lines of Code) - Linije koda
- **E**(estimated) - predstavlja procenu
- **O**(ptimistic) - predstavlja optimistično procenjeno vreme
- **P**(esimistic) - predstavlja pesimističko procenjeno vreme

Predviđanje broja linija koda		
Optimistično	Pretpostavljeno	Pesimistično
27.000	33.000	37.000

Tabela 2.2.1/1 - Predviđanje broj linija koda(LOC)

Na osnovu prethodno date tabele vršimo izračunavanje broja linija koda(LOC) korišćenjem prethodno napomenute formule, te dobijamo sledeće:

$$\text{LOC} = (27.000 + 4 * 33.000 + 37.000) / 6 = \mathbf{32.666,67}$$

Sledeće što treba uraditi je izračunavanje funkcionalnih poena(FP), na osnovu izračunatog broja linija koda, tako što ćemo podeliti broj linija koda sa koeficijentom programskog jezika, koji u našem slučaju predstavlja 53 za JAVA programski jezik, iz prethodne navedene Tabele 3.

$$\text{FP} = 32.666,67 / 53 = \mathbf{616.36}$$

2.2.2 Procena veličine bazirana na Funkcionalnoj analizi (FP)

Funkcionalne tačke su mera koja spada u kategoriju mera za krajnjeg korisnika poslovnih funkcija. FP su određivane na više metodičan način nego metod brojanja LOC.

Jedan od najpoznatijih metoda za estimaciju veličine softvera koji se razvija je Metod Funkcionalnih Tačaka - FP je zasnovan na ideji da se veličina softvera bolje izražava preko broja i složenosti funkcija koje softver izvršava nego što to prikazuje broj linija koda.

Prvi rad u kome je objavljen metod funkcionalnih tačaka potiče iz davne 1970, koje je opisao A.J. Albrecht iz IBM za sisteme transakciono – orijentisane. Caper Jones iz Software Productivity Research, Inc., proširio je Albrecht-ovu ideju u dobro i široko ustanovljenu oblast znanja.

Početkom 1986., neprofitna grupa (International Function Point User Group - IFPUG) je bila formirana da širi informacije o ovoj metrici. Početkom 1987. godine, Britanska vlada je usvojila modifikovan metod funkcionalnih tačaka za standardnu metriku produktivnosti razvoja softvera.[3]

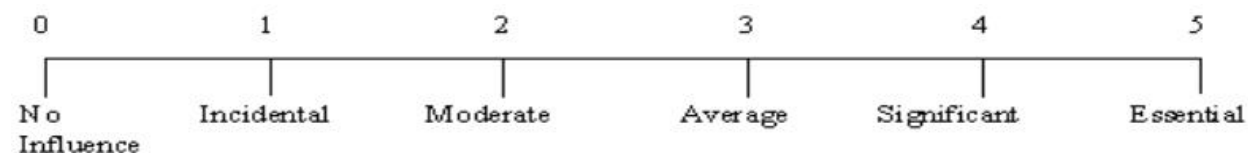
Formula za izračunavanje funkcionalnih poena:

$$FP = UPF \times VAF$$

UPF(Unadjusted Function Points) - neprilagođeni funkcionalni poeni

VAF(Value added Factor) = $0.65 + (0.01 * TDI)$ - vrednost dodatnog faktora

TDI(Total Degree of Influence) - totalni stepen uticaja generalnih sistemskih karakteristika.



Slika 2.2.2/1 - Generalne sistemske karakteristike[2]

Broj linija koda se može odrediti i na osnovu funkcionalnih poena i faktora programskog jezika sledećom formulom:

$$\text{LOC} = \text{FP} * \text{LF}$$

LOC(Lines of Code) - predstavlja broj linija koda.

FP(Functional Points) - predstavlja funkcionalne poene.

LF(Language Factor) - predstavlja faktor programskog jezika i zavisi od konkretnog odabranog programskog jezika.

Za izračunavanje broja funkcionalnih poena korišćena je online aplikacija, odnosno kalkulator, čiji link će se nalaziti u okviru poglavlja [10.3](#).

Domain Characteristic Table

MEASUREMENT PARAMETER	COUNT (value >= 0)	WEIGHTING FACTOR		
		Simple	Average	Complex
Number of User Input	66	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
Number of User Outputs	6	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
Number of User Inquiries	12	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
Number of Files	13	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
Number of External Interfaces	5	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>

Slika 2.2.2/2 - Domenske karakteristike sistema

Complexity Adjustment Table

ITEM	COMPLEXITY ADJUSTMENT QUESTIONS	SCALE					
		No Influence					Essential
		0	1	2	3	4	5
1	Does the system require reliable backup and recovery?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
2	Are data communications required?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
3	Are there distributed processing functions?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
4	Is performance critical?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
5	Will the system run in an existing, heavily utilized operational environment?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
6	Does the system require on-line data entry?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
7	Does the on-line data entry require the input transaction to be built over multiple screens or operations?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
8	Are the master files updated on-line?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
9	Are the inputs, outputs, files or inquiries complex?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
10	Is the internal processing complex?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
11	Is the code to be designed reusable?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
12	Are conversion and installation included in the design?	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
13	Is the system designed for multiple installations in different organizations?	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
14	Is the application designed to facilitate change and ease of use by the user?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>

Slika 2.2.2/3 - Podešavanje uticaja i kompleksnosti faktora

RESULT	
PROJECT FUNCTION POINTS	588.1200000000001

Slika 2.2.2/4 - Rezultat projektnih funkcionalnih poena

Kao rezultat izračunavanja projektnih funkcionalnih poena, na osnovu unetih podatka, aplikacija nam je generisala sledeći rezultat: **588.12** funkcionalnih poena.

Na osnovu funkcionalnih poena možemo izračunati i potencijalni broj linija koda:

$$LOC = 588.12 * 53 = 31.170,36$$

2.2.3 Procena veličine bazirana na COCOMO II modelu

COCOMO(Constructive Cost Model) je konstruktivni model troškova za procenu napora i vremena razvoja softvera na bazi veličine softvera u LOC. COCOMO je prvobitno objavljen u tekstu Ekonomija Softverskog Inženjerstva(Boehm - 1981. godine). Ovaj originalni model se često naziva COCOMO 81. Model je definisan na osnovu analize 63 završena projekata iz različitih oblasti u toku 1970-ih i početkom 1980. Da bi se rešili problemi koji proizilaze iz unapređenja i promena u tehnologijama i razvojnim procesima, USC Centar za sisteme i Softversko inženjerstvo je razvio i objavio COCOMO II. Model je prvobitno objavljen(Boehm 1995. godine), a zatim je objavljen u knjizi(Boehm 2000. godine). Među glavnim nadogradnjama su uvođenje novih funkcionalnih obrazaca koji koriste skala faktore, nove cene za korisnike, kao i novi set vrednosti parametara.

COCOMO II se sastoji od tri pod-modela, a to su: Kompozicija Aplikacije, Rani Dizajn i Post - Arhitektura. Model Kompozicije Aplikacije se koristi, za izračunavanje napora i rasporeda, da bi se razvio sistem koji je integrisan za višekratnu upotrebu komponenata i drugih sredstava, pomoću integrisanog razvojnog alata za projektovanje, izgradnju, integraciju , i testiranje.

Model kompozicije aplikacije ima drugačiji obrazac estimacije od drugih modela. On koristi za unos veličine merene u smislu primene poena aplikacije ili poena objekata (Kaufman i Kumar 1993 , 1994 Bankar) kao i stope produktivnosti da se izračuna napor. Model prevremenog dizajna je korišćen u ranim fazama projekta, kada informacije nisu dovoljno detaljne za preciznije estimacije. Kada su detaljne informacije, model post-arhitekture se naizmenično koristi. Modeli prevremeni dizajn i post-arhitektura koriste linije izvornog koda , kao osnovnu jedinicu veličine, i prate isti obrazac aritmetike.[4]

Opšti oblik obrasca za izračunavanje napora COCOMO 81, Modela Ranog dizajna i Post - Arhitekture mogu biti napisan kao:

$$PM = A * Size^B * \prod_{i=1}^{15} EM_i$$

Slika 2.2.3/1 - Formula COCOMO '81 modela za izračunavanje napora

Parametri formuli predstavljaju sledeće:

PM - estimacija napora osobe izražena u mesecima.

A - multiplikativna konstanta, koja se može kalibrisati pomoću istorijskih podataka.

Size – je veličina tj. estimirana veličina softvera, merena u KSLOC .

B - je eksponencijalna konstanta (COCOMO I) ili skala faktora (COCOMO II).

EM - je multiplikator napora, koji predstavljaju multiplikativne komponente jednačine.

Zato što je COCOMO nezaštićeni model, i kako su detalji dostupni u javnosti, podstiču se istraživači i praktičari na polju softverskog inženjerstva da samostalno estimiraju model. Bilo je mnogo samostalno prijavljenih proširenja (Kemerer 1987, Jeffery i Low 1990, Gulezian 1991, Menzies 2006).

COCOMO model nam omogućava da odredimo vreme trajanja projekta u jedinici čovek-mesec na osnovu procenjenog broja hiljada linija koda projekta.

Model se zasniva na zavisnosti vremena razvoja od broja linija koda i prirode projekta. Prošireni COCOMO vrši estimacije sa nekim dodatnim parametrima kao što su : atributi proizvoda, atributi računara, atributi zaposlenih i atributi projekta. Svaki od tih atributa ima podatribute, a svaki podatribut se ocenjuje na skali od jedan do šest. Proizvod svih vrednosti podatributa daje EAF (effort adjustment factor) koji može varirati od 0.9 do 1.4.

Proširena COCOMO formula onda glasi:

$$E = a_i * (KSLOC)^{b_i} * EAF$$

E - označava trud izražen u čovek-meseci.

KLOC - je procenjen broj linija koda.

EAF - je faktor koji se računa prethodno opisanim postupkom.

Parametri **a_i** i **b_i** zavise od tipa projekta (organic, semi-detached, embedded), a prema atributima za izbor tipa projekta. U našem slučaju je u pitanju semi-detached tip projekta, pa je prema podacima iz Tabele 8 uzeto da je $a_i = 2.4$ i $b_i = 1.05$.

	Ai		Bi	
Tip	Osnovni	Srednji	Osnovni	Srednji
Organic	2.4	3.2	1.05	1.05
Semi-Detached	3.0	3.0	1.12	1.12
Embedded	3.6	2.8	1.20	1.20

Tabela 2.2.3/1 - Rezultat projektnih funkcionalnih poena

Kalkulaciju po COCOMO II modelu će se izvršiti preko online alata, odnosno kalkulatora, čiji link možete pronaći u okviru poglavlja [10.3](#).

USC CSSE

COCOMO II - Constructive Cost Model

Model(s): COCOMO
Monte Carlo Risk: Off
Auto Calculate: Off

Software Size Sizing Method: Source Lines of Code

[SLOC](#)

	% Design Modified	% Code Modified	% Integration Required	Assessment and Assimilation (0% - 8%)	Software Understanding (0% - 50%)	Unfamiliarity (0-1)
New	33000					
Reused	58000	0	0	10	1	
Modified	7000	5	5	3	0	50

Software Scale Drivers

Precedentedness	Very Low	Architecture / Risk Resolution	Nominal	Process Maturity	Nominal
Development Flexibility	High	Team Cohesion	Very High		

Software Cost Drivers

Product		Personnel		Platform		Project	
Required Software Reliability	High	Analyst Capability	Nominal	Time Constraint	Nominal	Use of Software Tools	High
Data Base Size	High	Programmer Capability	High	Storage Constraint	Nominal	Multisite Development	Nominal
Product Complexity	Nominal	Personnel Continuity	High	Platform Volatility	Low	Required Development Schedule	Nominal
Developed for Reusability	Nominal	Application Experience	High				
Documentation Match to Lifecycle Needs	Low	Platform Experience	Very High				
		Language and Toolset Experience	Very High				

Maintenance: Off

Software Labor Rates

Cost per Person-Month (Dollars): 1500

Calculate

Slika 2.2.3/2 - COCOMO II kalkulator

Informacije koje su unete u COCOMO II kalkulatoru su unete na osnovu Carper Jones-ovih podataka I mesečnom trošku po osobi u iznosu od 1500.

Po slici 3 možemo videti sledeće podatke:

SLOC: Očekuje se 33.000 novih linija koda(*New*), dok se ponovo iskorišćenih očekuje 58.000(*Reused*), od kojih je 7000 izmenjeno(*Modified*).

% Design Modified: U okviru modifikacije dizajna se očekuje 5% izmene.

% Code Modified: U okviru modifikacije koda se očekuje 5% izmene.

% Integration Required: 10% ponovo iskorišćenog koda se mora integrisati, kao i 3% modifikovanog koda.

Assessment and Assimilation: 1% ponovo iskorišćenog koda mora proći proces asimilacije i asesmenta.

Software Understanding: Razumljivost softvera je 50% kao i maksimalni mogući stepen za ovu vrednost.

Unfamiliarity: 0 jer ne postoji nedoumica i nepoznanica softvera.

Ostali faktori koji se mogu videti na slici su rangirani od Very Low - Very High.

Unos jeste subjektivan i zavisi isključivo od projektnog menadžera, njegovog znanja, poznavanja radnog tima, kao i iskustva.

Results

Software Development (Elaboration and Construction)

Effort = 64.8 Person-months

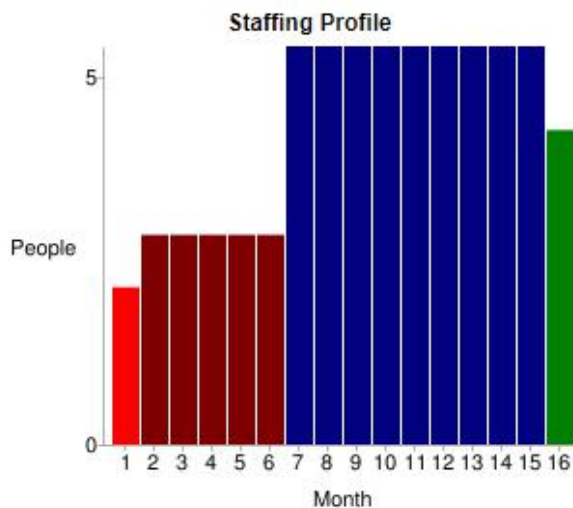
Schedule = 14.5 Months

Cost = \$97224

Total Equivalent Size = 35628 SLOC

Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	3.9	1.8	2.1	\$5833
Elaboration	15.6	5.5	2.9	\$23334
Construction	49.3	9.1	5.4	\$73891
Transition	7.8	1.8	4.3	\$11667



Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.5	1.9	4.9	1.1
Environment/CM	0.4	1.2	2.5	0.4
Requirements	1.5	2.8	3.9	0.3
Design	0.7	5.6	7.9	0.3
Implementation	0.3	2.0	16.7	1.5
Assessment	0.3	1.6	11.8	1.9
Deployment	0.1	0.5	1.5	2.3

Slika 2.2.3/3 - COCOMO II rezultati

Na slici 4 se može videti iskalkulisane vrednosti prethodno unetih parametara.

Napor (**Effort**)= 64.8 čovek-meseci

Rok izvršenja(**Schedule**) = 14.5 meseci

Cena izrade(**Cost**) = \$97.224

Veličina softvera u linijama koda(**Total Equivalent Size**) = 35.628 SLOC

Možemo videti histogram razvoja softvera na slici 4 podeljen u 4 faze: Početak(Inception), Razrada(Elaboration), Izgradnja(Construction) i Tranzicija(Transition), takođe možemo uočiti sa iste slike da faze izgradnje(plave boje naznačeno) i razvoja(zelene boje naznačeno) zauzimaju najveći deo napora.

Do broja funkcionalnih poena se takođe može doći tako što podelimo broj linija koda sa 53(JAVA programski jezik):

$$FP = 35.628 / 53 = 672.227$$

2.2.4 Analiza izvršenih procena

	Analiza		
	Po broju linija koda	Po funkcionalnim poenima	COCOMO II
LOC	32.667,67	31.170,36	35.628,00
FP	616,36	588,12	672,227

Tabela 2.2.4/1 - Analiza izvršenih procena

Možemo videti da nemamo ekstremnih odsupanja između vrednosti.

Izračunaćemo srednju vrednosti za ove tri procene, i nju ćemo uzeti kao glavnu procenu za naš sistem:

$$\text{LOC} = (32.667,67 + 31.170,36 + 35628) / 3 = \mathbf{33155,34}$$

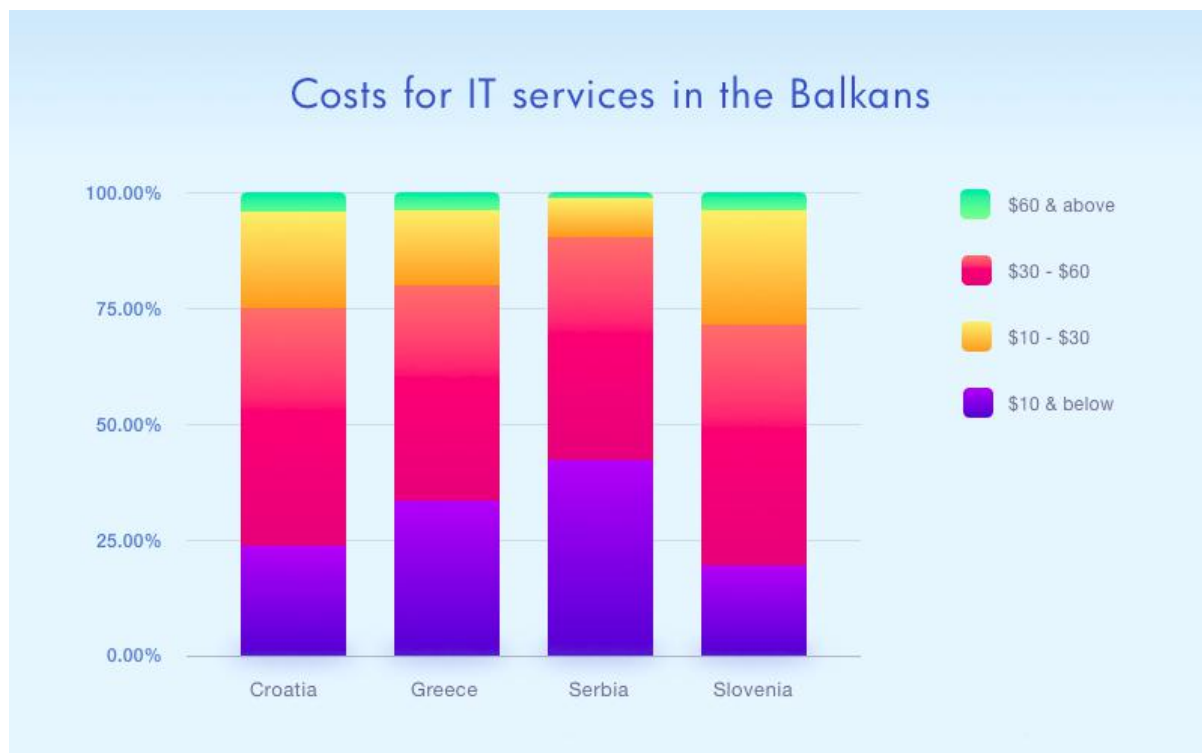
$$\text{FP} = (616,36 + 588,12 + 672,227) / 3 = \mathbf{625,57}$$

2.3 Procena resursa projekta

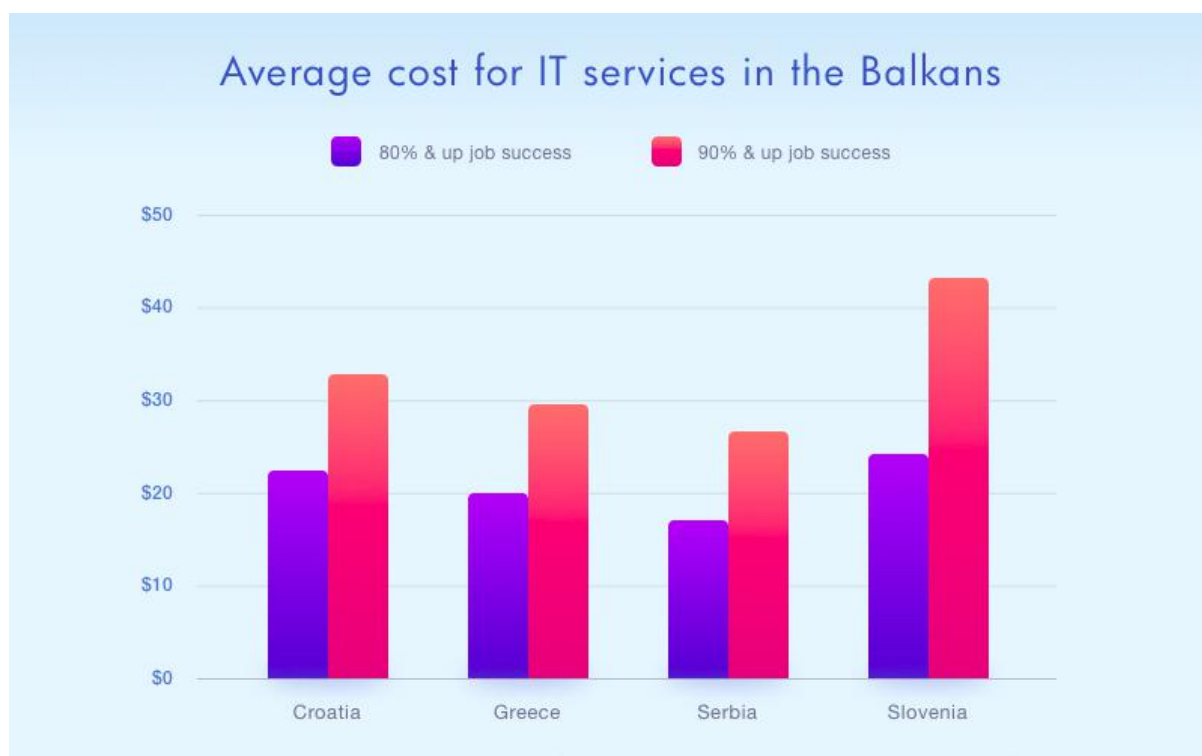
Procena resursa projekata podrazumeva definisanje ljudi (broj ljudi, obučenosti, raspoređivanje..), Minimalne hardverske kao i softverske zahteve

2.3.1 Ljudstvo

Najveći trošak svakako predstavlja ljudstvo (zaposleni), kada se radi o softverskom projektu, stoga, da bi izračunali prosečnu cenu zaposlenog po satu, moramo istraživanje bazirati na našem regionu odnosno Republici Srbiji, te za analizu uzimamo prosečnu cenu sata na osnovu statističkih podataka Srbije, koji se nalaze na sledećoj slici.



Slika 2.3.1/1 – Cena koštanja IT usluga na Balkanu[4]



Slika 2.3.1/2 – Prosečna cena koštanja IT usluga na Balkanu[4]

Na osnovu istraživanja popunjena je sledeća tabela za cene koštanja resursa. Postoje tri tipa resursa: radni (work), materijal (material) i trošak (cost). Na slici je predstavljena tabela troškova softvera, gde su sivom bojom obeleži radni, plavom bojom materijalni resursi, dok su narandžastom bojom označeni resursi troškova.

Svaki resurs služi određenom procesu ili podprocesu. Neki resursi su upotrebljivi više puta. Jedan od njih je i NetBeans IDE razvojno okruženje, BitRix24, PowerDesigner.

Resource Name ▼	Type ▼	Initials ▼	Max. ▼	Std. Rate ▼	Ovt. ▼	Cost/Use ▼	Accrue ▼	Base ▼
Project Manager	Work	P	100%	\$30.00/hr	\$35.00/hr	\$0.00	Prorated	Standard
Management	Work	M	100%	\$22.00/hr	\$23.00/hr	\$0.00	Prorated	Standard
Analyst	Work	A	100%	\$22.00/hr	\$23.00/hr	\$0.00	Prorated	Standard
Back-End Developer	Work	B	100%	\$30.00/hr	\$35.00/hr	\$0.00	Prorated	Standard
Designer	Work	D	100%	\$15.00/hr	\$18.00/hr	\$0.00	Prorated	Standard
Front-End Developer	Work	F	100%	\$25.00/hr	\$30.00/hr	\$0.00	Prorated	Standard
Tester	Work	T	100%	\$20.00/hr	\$23.00/hr	\$0.00	Prorated	Standard
Technical Support	Work	T	100%	\$18.00/hr	\$20.00/hr	\$0.00	Prorated	Standard
Deployment Team	Work	D	100%	\$20.00/hr	\$21.50/hr	\$0.00	Prorated	Standard
Server	Material	S		\$2,000.00		\$0.00	Prorated	
Hardver	Material	H		\$300.00		\$0.00	Prorated	
Power Designer	Material	P		\$0.00		\$50.00	Prorated	
Selenium	Material	S		\$0.00		\$0.00	Prorated	
Bitrix24	Material	B		\$0.00		\$13.00	Prorated	
NetBeans	Material	N		\$0.00		\$3.00	Prorated	
MS Project	Material	M		\$0.00		\$10.00	Prorated	
External expert	Cost	E					Prorated	

Slika 2.3.1/3 – Resursi razvoja softvera

Na slikama ispod su raspoređeni resursi.


	Tas Mc	Task Name	Duratio	Start	Finish	Pre	Resource Names	Cost
		▲ Razvoj NSI sistema	154 days	Mon 26-03-18	Thu 25-10-18			\$46,852.00
		▲ Planiranje projekta	6 days	Mon 26-03-18	Mon 02-04-18		Bitrix24[1], External expert[\$350.00], MS Project[1]	\$2,199.00
		Procena veličine projekta	1 day	Mon 26-03-18	Mon 26-03-18		Project Manager, Management	\$416.00
		Procena sredstava za razvoj projekta	2 days	Tue 27-03-18	Wed 28-03-18	3	Project Manager, Management	\$832.00
		Planiranje razvoja softvera	2 days	Thu 29-03-18	Fri 30-03-18	4	Management	\$352.00
		Procena finansija	1 day	Mon 02-04-18	Mon 02-04-18	5	Management	\$176.00
		▲ Zahtevi projekta	10 days	Tue 03-04-18	Mon 16-04-18		Bitrix24[1], MS Project[1], Power Designer[1]	\$2,377.00
		Prikupljanje zahteva	4 days	Tue 03-04-18	Fri 06-04-18	6	Analyst	\$704.00
		Definisanje zahteva projekta	2 days	Mon 09-04-18	Tue 10-04-18	8	Analyst	\$352.00
		Analiza definisanih zahteva	2 days	Wed 11-04-18	Thu 12-04-18	9	Analyst[50%], Project Manager[50%]	\$416.00
		Funkcionalni i nefunkcionalni zahtevi	4 days	Wed 11-04-18	Mon 16-04-18	9	Analyst[50%], Project Manager[50%]	\$832.00
		▲ Dizajniranje softvera sistema	13 days	Tue 17-04-18	Thu 03-05-18		Bitrix24[1], External expert[\$350.00], NetBeans[1], Power Designer[1]	\$6,672.00
		Pregled specifikacije	1 day	Tue 17-04-18	Tue 17-04-18	11	Analyst, Project Manager	\$416.00
		Planiranje i dizajniranje arhitekture sistema	7 days	Wed 18-04-18	Thu 26-04-18	13	Back-End Developer, Project Manager	\$3,360.00
		Dizajn grafičkog korisničkog interfejsa	4 days	Fri 27-04-18	Wed 02-05-18	14	Designer, Front-End Developer	\$1,280.00
		Dizajniranje modela I baze podataka	5 days	Fri 27-04-18	Thu 03-05-18	14	Back-End Developer	\$1,200.00

Slika 2.3.1/4 – Resursi prema procesima razvoja softvera - 1


	Implementacija softvera	10 days	Fri 04-05-18	Thu 17-05-18		Hardver[1], NetBeans[1], Power Designer[1], Server[1]	\$7,233.00
	Obrada zahteva sistema	4 days	Fri 04-05-18	Wed 09-05-18	16,15	Back-End Developer, Front-End Developer	\$1,760.00
	Pregled zahteva i dizajna sistema	2 days	Thu 10-05-18	Fri 11-05-18	18	Back-End Developer, Front-End Developer	\$880.00
	Planiranje i raspoređivanje timova	2 days	Mon 14-05-18	Tue 15-05-18	19	Back-End Developer, Front-End Developer, Project Manager	\$1,360.00
	Planiranje i organizacija razvoja softvera	2 days	Wed 16-05-18	Thu 17-05-18	20	Back-End Developer, Front-End Developer	\$880.00
	Razvoj softvera	55 days	Fri 18-05-18	Thu 02-08-18		Bitrix24[1], NetBeans[1], Power Designer[1], Serv	\$12,866.00
	Implementacija baze podataka	5 days	Fri 18-05-18	Thu 24-05-18	21	Back-End Developer	\$1,200.00
	Razvoj prototipa	10 days	Fri 25-05-18	Thu 07-06-18	23	Designer	\$1,200.00
	Analiza prototipa i prikupljanje grešaka	5 days	Fri 08-06-18	Thu 14-06-18	24	Designer	\$600.00
	Back-end programiranje	20 days	Fri 15-06-18	Thu 12-07-18	25	Back-End Developer	\$4,800.00
	Izrada klijentske aplikacije	15 days	Fri 13-07-18	Thu 02-08-18	26	Front-End Developer	\$3,000.00
	Testiranje	20 days	Fri 03-08-18	Thu 30-08-18		Hardver[1], MS Project[1], Server[1]	\$5,510.00
	Alfa testiranje	10 days	Fri 03-08-18	Thu 16-08-18	27	Tester	\$1,600.00
	Beta testiranje	10 days	Fri 17-08-18	Thu 30-08-18	29	Tester	\$1,600.00
	Dokumentacija	40 days	Fri 31-08-18	Thu 25-10-18		Bitrix24[1], MS Project[1], NetBeans[1], Power De	\$6,200.00
	Izrada dokumentacij	35 days	Fri 31-08-18	Thu 18-10-18	30	Technical Support	\$5,040.00
	Izrada korisničkog uputstva	5 days	Fri 19-10-18	Thu 25-10-18	32	Front-End Developer[50%], Analyst[50%]	\$940.00
	Pregled konačne dokumentacije	1 day	Fri 19-10-18	Fri 19-10-18	32	Technical Support	\$144.00
	Puštanje u rad	3 days	Mon 22-10-18	Wed 24-10-18	30	Hardver[1], NetBeans[1], Server[1]	\$3,455.00
	Isporuka softvera	3 days	Mon 22-10-18	Wed 24-10-18	34	Project Manager, Technical Support	\$1,152.00
	Održavanje	1 day	Thu 25-10-18	Thu 25-10-18	35	Back-End Developer, Front-End Developer[50%]	\$340.00

Slika 2.3.1/5 – Resursi prema procesima razvoja softvera - 2

Svaki resurs ima određenu iskorišćenost po kojoj se meri, to mogu biti vreme, broj iskorišćenja i slično. Sada će se na sledeće 3 slike prikazati iskorišćenost resursa za razvoj.

		Resource Name	Work
1		Project Manager <i>Procena veličine projekta</i> <i>Procena sredstava za razvoj projekta</i> <i>Analiza definisanih zahteva</i> <i>Funkcionalni I nefunkcionalni zahtevi</i> <i>Pregled specifikacije zahteva</i> <i>Planiranje I dizajniranje arhitekture sistema</i> <i>Planiranje I raspoređivanje timova</i> <i>Isporuka softvera</i>	152 hrs 8 hrs 16 hrs 8 hrs 16 hrs 8 hrs 56 hrs 16 hrs 24 hrs
2		Management <i>Procena veličine projekta</i> <i>Procena sredstava za razvoj projekta</i> <i>Planiranje razvoja softvera</i> <i>Procena finansija</i>	48 hrs 8 hrs 16 hrs 16 hrs 8 hrs
3		Analyst <i>Prikupljanje zahteva</i> <i>Definisanje zahteva projekta</i> <i>Analiza definisanih zahteva</i> <i>Funkcionalni I nefunkcionalni zahtevi</i> <i>Pregled specifikacije zahteva</i> <i>Izrada korisničkog uputstva</i>	100 hrs 32 hrs 16 hrs 8 hrs 16 hrs 8 hrs 20 hrs
4		Back-End Developer <i>Planiranje I dizajniranje arhitekture sistema</i> <i>Dizajniranje modela I baze podataka</i> <i>Obrada zahteva sistema</i> <i>Pregled zahteva I dizajna sistema</i> <i>Planiranje I raspoređivanje timova</i> <i>Planiranje I organizacija razvoja softvera</i> <i>Implementacija baze podataka</i> <i>Back-end programiranje</i> <i>Održavanje</i>	384 hrs 56 hrs 40 hrs 32 hrs 16 hrs 16 hrs 16 hrs 40 hrs 160 hrs 8 hrs
5		Designer <i>Dizajn grafičkog korisničkog interfejsa</i> <i>Razvoj prototipa</i> <i>Analiza prototipa I prikupljanje grešaka</i>	152 hrs 32 hrs 80 hrs 40 hrs
6		Front-End Developer <i>Dizajn grafičkog korisničkog interfejsa</i> <i>Obrada zahteva sistema</i>	256 hrs 32 hrs 32 hrs

Slika 2.3.1/6 – Iskorišćenosti resursa - 1

	 Resource Name	Work
	<i>Pregled zahteva i dizajna sistema</i>	16 hrs
	<i>Planiranje i raspoređivanje timova</i>	16 hrs
	<i>Planiranje i organizacija razvoja softvera</i>	16 hrs
	<i>Izrada klijentske aplikacije</i>	120 hrs
	<i>Izrada korisničkog uputstva</i>	20 hrs
	<i>Održavanje</i>	4 hrs
7	▲ Tester	160 hrs
	<i>Alfa testiranje</i>	80 hrs
	<i>Beta testiranje</i>	80 hrs
8	▲ Technical Support	312 hrs
	<i>Izrada dokumentacije</i>	280 hrs
	<i>Pregled konačne dokumentacije</i>	8 hrs
	<i>Isporuka softvera</i>	24 hrs
9	Deployment Team	0 hrs
10	▲ Server	4
	<i>Implementacija softvera</i>	1
	<i>Razvoj softvera</i>	1
	<i>Testiranje</i>	1
	<i>Puštanje u rad</i>	1
11	▲ Hardver	3
	<i>Implementacija softvera</i>	1
	<i>Testiranje</i>	1
	<i>Puštanje u rad</i>	1
12	▲ Power Designer	6
	<i>Planiranje projekta</i>	1
	<i>Zahtevi projekta</i>	1
	<i>Dizajniranje softvera sistema</i>	1
	<i>Implementacija softvera</i>	1
	<i>Razvoj softvera</i>	1
	<i>Dokumentacija</i>	1
13	Selenium	0
14	▲ Bitrix24	5
	<i>Planiranje projekta</i>	1
	<i>Zahtevi projekta</i>	1
	<i>Dizajniranje softvera sistema</i>	1
	<i>Razvoj softvera</i>	1
	<i>Dokumentacija</i>	1
15	▲ NetBeans	5

Slika 2.3.1/7 – Iskorišćenosti resursa - 2

15	▀ NetBeans	5
	<i>Dizajniranje softvera sistema</i>	1
	<i>Implementacija softvera</i>	1
	<i>Razvoj softvera</i>	1
	<i>Dokumentacija</i>	1
	<i>Puštanje u rad</i>	1
16	▀ MS Project	4
	<i>Planiranje projekta</i>	1
	<i>Zahtevi projekta</i>	1
	<i>Testiranje</i>	1
	<i>Dokumentacija</i>	1
17	▀ External expert	
	<i>Planiranje projekta</i>	
	<i>Dizajniranje softvera sistema</i>	

Slika 2.3.1/8 – Iskorišćenosti resursa – 3

2.3.2 Analiza troškova

Analizu troškova moguće je i poželjno sprovesti tri puta tokom razvoja softvera. Prvi put na početku razvoja, kao što je to ovde slučaj i u tom slučaju vrši se analiza procene troškova.

Drugi put u sredini razvoja, jer smo tom analizom u mogućnosti da pratimo kako se kreće naš razvoj softvera, da li prelazimo predviđen budžet, koje tačke smo precenili, a koje smo potcenili. Ova analiza je veoma bitna kako bismo u toku razvoja softvera vršili određene promene koje će smanjiti troškove, smanjiti vreme isporuke ili povećati kvalitet softveru. Ova osobina zove se agilnost i veoma ju je teško postići.

Treća analiza vrši se na kraju razvoja softvera kada posedujemo sve podatke iz tek završenog procesa razvoja. Ova analiza je od ključne važnosti za unapređenje sledeće procese razvoja softvera.

COST OVERVIEW

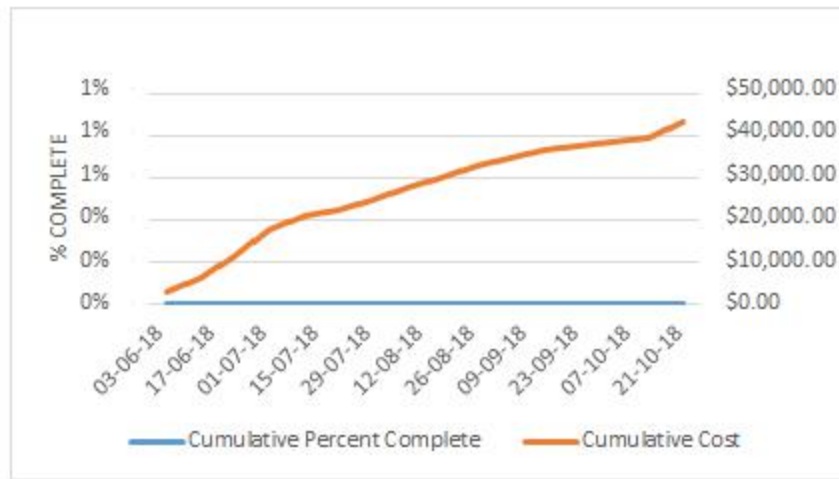
MON 26-03-18 THU 25-10-18



Slika 2.3.2/1 – Pregled troškova sistema

PROGRESS VERSUS COST

Progress made versus the cost spent over time. If % Complete line below the cumulative cost line, your project may be over budget.



Slika 2.3.2/2 – Napredak troškova

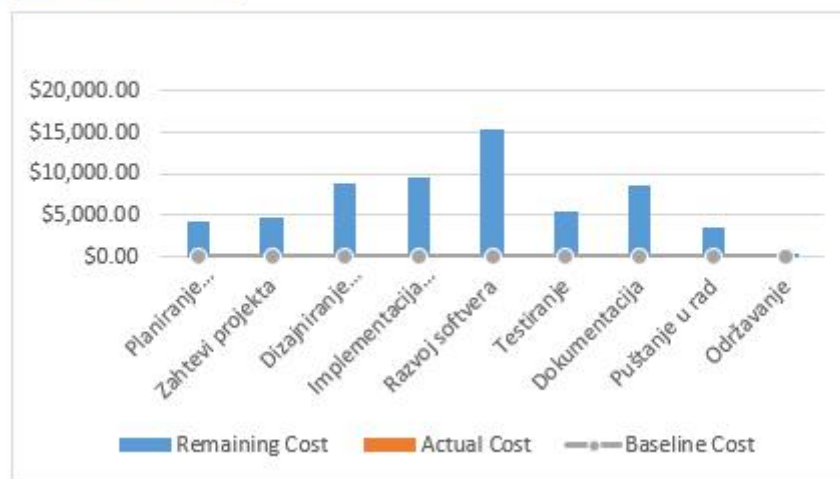
Slika iznad prikazuje kumulativnu krivu troškova.

Plava linija predstavlja krivu troškova u realnom vremenu.

COST STATUS

Cost status for all top-level tasks. Is your baseline zero?

[Try setting as baseline](#)



Slika 2.3.2/3 – Status troškova

Slika 2.3.1/11 predstavlja tabelu statusa troškova što se odnosi na njihovu procenjenu vrednost. Vrednost koja je potrošena i vrednost koja preostaje.

COST STATUS

Cost status for top level tasks.

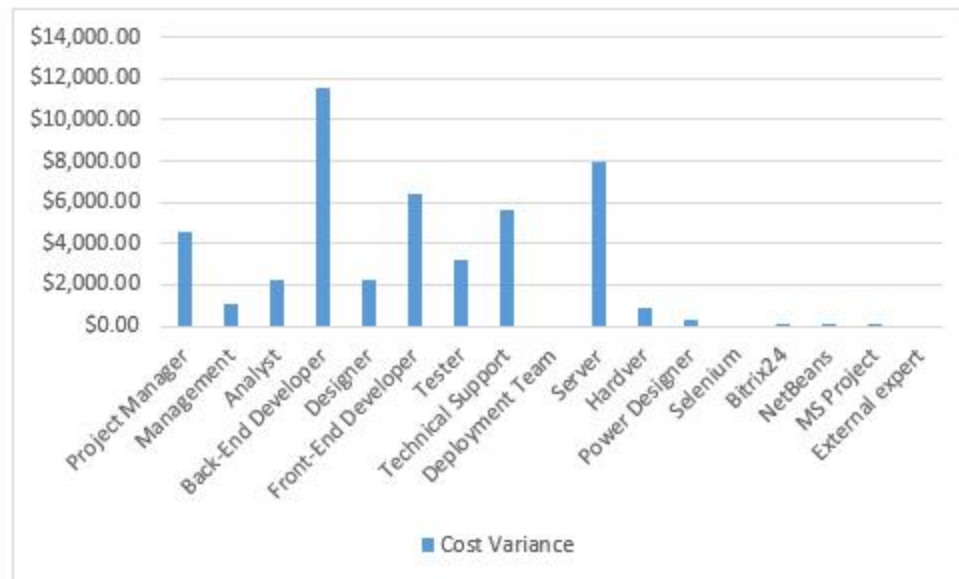
Name	Actual Cost	Remaining Cost	Baseline Cost	Cost	Cost Variance
Planiranje projekta	\$0.00	\$4,299.00	\$0.00	\$4,299.00	\$4,299.00
Zahtevi projekta	\$0.00	\$4,827.00	\$0.00	\$4,827.00	\$4,827.00
Dizajniranje softvera sistema	\$0.00	\$8,772.00	\$0.00	\$8,772.00	\$8,772.00
Implementacija softvera	\$0.00	\$9,683.00	\$0.00	\$9,683.00	\$9,683.00
Razvoj softvera	\$0.00	\$15,316.00	\$0.00	\$15,316.00	\$15,316.00
Testiranje	\$0.00	\$5,510.00	\$0.00	\$5,510.00	\$5,510.00
Dokumentacija	\$0.00	\$8,650.00	\$0.00	\$8,650.00	\$8,650.00
Puštanje u rad	\$0.00	\$3,455.00	\$0.00	\$3,455.00	\$3,455.00
Održavanje	\$0.00	\$340.00	\$0.00	\$340.00	\$340.00

Slika 2.3.2/4 – Status troškova – II

Ova slika predstavlja tabelarni prikaz statusa troškova.

RESOURCE COST VARIANCE

Cost variance for all the work resources.



Slika 2.3.2/5 – Vratiranje troškova

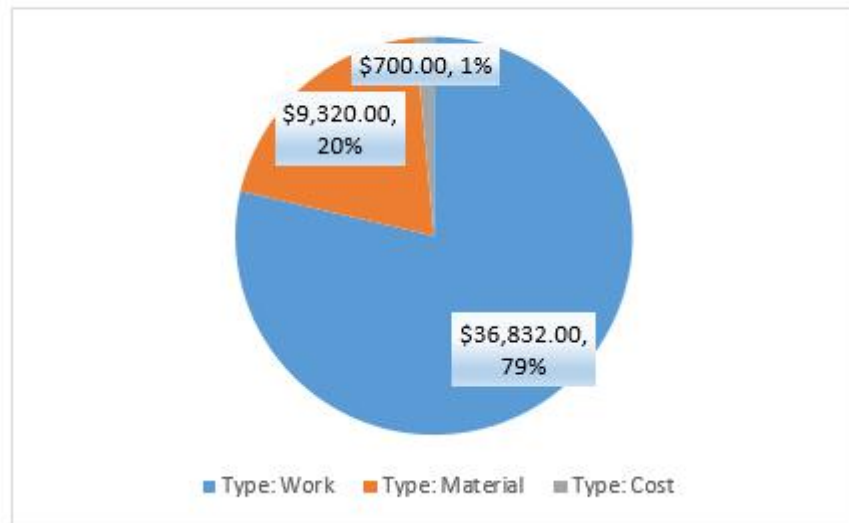
Name	Cost	Baseline Cost	Cost Variance
Project Manager	\$4,560.00	\$0.00	\$4,560.00
Management	\$1,056.00	\$0.00	\$1,056.00
Analyst	\$2,200.00	\$0.00	\$2,200.00
Back-End Developer	\$11,520.00	\$0.00	\$11,520.00
Designer	\$2,280.00	\$0.00	\$2,280.00
Front-End Developer	\$6,400.00	\$0.00	\$6,400.00
Tester	\$3,200.00	\$0.00	\$3,200.00
Technical Support	\$5,616.00	\$0.00	\$5,616.00

Slika 2.3.3/6 – Vratiranje troškova – II

Slike iznad prikazuju tabele statusa troškova koje se odnose na njihovu vrednost koja je potrošena i vrednost koja preostaje.

COST DISTRIBUTION

How costs are spread out amongst tasks based on their status.



Slika 2.3.2/7 – Pita distribucije troškova resursa prema tipu

Slika 2.3.2/7 predstavlja raspodelu troškova resursa prema tipu. Kao što smo naveli na početku postoje tri tipa troškova: radni, materijal i trošak.

Uočavamo da 79% troškova resursa otpada na radne troškove (zaposleni) a to je \$36,832.00, 20% otpada na materijalne resurse (servere, okruženja itd.) a to je \$9,320.00 , dok samo 1% otpada na resurse troška u šta spadaju treninzi i savetovanja, a trošak za ovaj resurs iznosi 700.00\$.

Ukupni predviđeni troškovi razvoja softvera NSI iznose \$46,852.00

2.3.3 Minimalni hardverski zahtevi

Sistem minimalno mora da poseduje sledeće uređaje:

- Računar
- Pametni uređaj
- Uređaj koji u sebi ima veb pretraživač i pristup internetu

2.3.4 Minimalni softverski zahtevi

Sistem ne zavisi ni od kakvog softvera, nego radi isključivo samostalno kao veb platforma, gde se za komunikaciju koriste sigurnosni HTTPS protokoli radi zaštite korisnika, kao i sigurnosni sertifikati, jer korisnici koriste svoje lične podatke, kao i kreditne kartice na sistemu.

Jedini softver koji je potreban za pristup ka sistemu je Veb pretraživač, bilo kog proizvođača, jer je sistem tako napravljen da je prilagođen svim poznatim i aktuelnim pretraživačima.

3. Procene kvaliteta, održavanja i testiranja softvera

U ovom poglavlju biće govora o proceni kvaliteta, održavanju kao i testiranju softvera. Obradiće se metoda procene “Rule of thumb”, dvanaest pravila Carper Jones-a, kao i procena održavanja COCOMO II modelom.

3.1 “Rule of Thumb” metoda procene

Za procenu softvera se najčešće koristi tehnika “Rule of Thumb”, u značenju vodič ili princip, koji je zasnovan na praksi, radije nego na teoriji. Ovu tehniku je utvrdio Capers Jones na osnovu metrika velikog broja projekata, koje je on istraživao.

Capers Jones je naveo dvanaest pravila na osnovu kojih se procenjuje softver. Kako bismo primenili tehniku “rule of thumb”, treba predhodno imati broj LOC ili izračunat broj funkcionalnih poena, koristiće se predviđanja zasnovana na broju linija koda, ali se ipak pokazao tačniji, u većini slučajeva, pristup procene na osnovu FP.

Table 1: Rules of Thumb Based on LOC Metrics for Procedural Languages
(Assumes 1 work month = 132 work hours)

Size of Program in LOC	Coding LOC per Month	Coding Effort (Months)	Testing Effort Percent	Noncode Effort Percent	Total Effort (Months)	Net LOC per Month
1	2500	0.0004	10.00%	10.00%	0.0005	2083
10	2250	0.0044	20.00%	20.00%	0.0062	1607
100	2000	0.0500	40.00%	40.00%	0.0900	1111
1,000	1750	0.5714	50.00%	60.00%	1.2000	833
10,000	1500	6.6667	75.00%	80.00%	17.0000	588
100,000	1200	83.3333	100.00%	100.00%	250.0000	400
1,000,000	1000	1000.0000	125.00%	150.00%	3750.0000	267

Slika 3.1/1 – Procena napora na mesečnom nivou[5]

Table 2: Rules of Thumb Based on LOC Metrics for Procedural Languages
(Assumes 1 work month = 132 work hours)

Size of Program in LOC	Coding LOC per Hour	Coding Effort (Hours)	Testing Effort Percent	Noncode Effort Percent	Total Effort (Hours)	Net LOC per Hour
1	18.94	0.05	10.00%	10.00%	0.06	15.78
10	17.05	0.59	20.00%	20.00%	0.82	12.18
100	15.15	6.60	40.00%	40.00%	11.88	8.42
1,000	13.26	75.43	50.00%	80.00%	173.49	5.76
10,000	11.36	880.00	75.00%	100.00%	2,420.00	4.13
100,000	9.09	11,000.00	100.00%	150.00%	38,500.00	2.60
1,000,000	7.58	132,000.00	125.00%	150.00%	495,000.00	2.02

Slika 3.1/2 – Procena napora na nivou radnog časa[5]

Na slikama 3.1/1 i 3.1/2 se nalaze tabele koje su ustanovljene na osnovu dobre prakse i koriste se pri procenama na osnovu broja LOC. Ove tabele su podeljene na dve, odnosno posebno na mesečnom i na časovnom nivou jer se za veće projekte koristi tabela 3.1/1 odnosno na mesečnom nivou kako bi se preciznije procenio napor, dok se za manje projekte teže izračunava procena na mesešnom nivou, tako da se za njih koristi na osnovu radnog časa.

Kao što vidimo na napomenutim slikama, vidimo da su tabele popunjene tako da pretpostavljen 1 radni mesec je jednak 132 radna časa ('Assumes 1 work month = 132 work hours'). Ovaj broj nije striktan, radi se o nekom izračunatom proseku, neke kompanije imaju manje od 132 radna časa, kao recimo 120, neke kompanije imaju više kao na primer 160 radna časa, kompanije vrše procenu svojih radnih časova na osnovu ove tabele, ne prilagođavaju ih direktno sa tabelom.

12 Pravila Carper Jones-a

Pravilo 1 - Estimacija veličine izvornog koda

Estimacija veličine izvornog odnosno programskog koda, odnosno linija koda - LOC vrši se na isti način kao što smo radili u tabeli 2.2.4/1. To se računa množenjem broja FP sa koeficijentom programskog jezika u kom je softver rađen, mi smo tu odabrali JAVA programski jezik (Slika 2.1/2). Pretpostavljamo da naš sistem koristi tri programska jezika, raspodelu ovoga možemo videti u tabeli ispod.

Na osnovu prethodnog proračuna (tabela 2.2.4/1) ukupan broj linija kod nam je 33155,34, tako da u slučaju 3 programska jezika ćemo podeliti to na osnovu procenata svakog pojedinačno, gde će zbir svih činiti ukupan broj LOC.

Programski jezik	% udela u softveru	Koeficijent	LOC
JAVA	85%	53	28182,04
JavaScript	5%	47	1657,77
SQL	10%	21	3315,53
Ukupan Broj linija koda - Lines of Code(LOC)			33155,34

Tabela 3.1/1 – Pravilo 1 - Ukupan broj LOC podeljen na osnovu jezika

Pravilo 2 – Estimacija dokumentacije

Estimacija veličine dokumentacije se izražava u broju potrebnih stranica dokumenta.

Dokumentacija je ključni deo projekta, bez njega softver ne može predstavljati softver i postaće eventualno neupotrebljiv i neodrživ.

Procena se vrši na osnovu sledeće formule:

$$\text{Dokumentacija} = \text{FP}^{1.15} = 625,57^{1.15} = \mathbf{1643,3 \text{ stranica}}$$

Pravilo 3- Estimacija izmene korisničkih zahteva

Estimacija korisničkih zahteva u proseku iznosi oko 2% mesečno, što je ustanovljeno na osnovu tehnike "*rule of thumb*", tako da možemo proceniti potencijalnu veličinu softvera koja uključuje i korisničke izmene. Kako bi se izbegla neslaganja potrebno je definisati u ugovoru i promeniti troškove ovog problema na vreme.

Broj funkcionalnih poena našeg sistema iznosi 625,57, možemo izračunati koliko će se na mesešnom nivou dodavati novih funkcionalnih poena:

$$FP * 2\% = 625,57 * 2\% = \mathbf{12,51 \text{ novih funkcionalnih poena}}$$

Procenjeno vreme razvoja softvera koju smo ustanovili na osnovu COCOMO II modela iznosi 14.5 meseci (Slika 2.2.3/3) od kojih se 9,1 meseci sastoji vreme razvoja i testiranja softvera, što se može videti na istoj slici 2.2.3/3 u okviru *Aquisition Phase Distribution* tabele, reda pod nazivom *Consturction*.

Na osnovu ove procene, možemo utvrditi da će softver imati **novih funkcionalnih poena**:

$$12,51 * 9,1 = \mathbf{113,84}$$

Iz ovoga zaključujemo da će veličina softvera najverovatnije iznositi $625,57 + 113,84 = \mathbf{739,41}$ funkcionalna poena nego prethodno izračunatih 625,57.

Pravilo 4 - Estimacija broja slučajeva testiranja

Estimacija broja slučajeva korišćenja, a i test slučajeva, je jedna od najbitnijih faza razvoja softvera, na koju treba obratiti posebnu pažnju. Procena se vrši sledećom formulom:

$$\text{Broj test slučajeva} = FP^{1.2} = 625,57^{1.2} = \mathbf{2267,42 \text{ test slučaja}}$$

Dakle, primenom 4. pravili smo ustanovili da nam je potrebno oko 2267 slučajeva korišćenja kako bismo potpuno testirali softver, uključujući sve vrste testiranja.

Pravilo 5 - Estimacija mogućnosti dolaska do greške

Estimacija broja potencijalnih grešaka predstavlja sumu svih tipova grešaka u koje spadaju

- greške u zahtevima
- dizajnu
- kodiranu
- dokumentaciji
- greške nastale nakon ispravljanja greške

Potencijalni broj grešaka utvrđujemo tako što stepenujemo broj funkcionalnih poena sa 1,25:

$$\text{Potencijalni broj grešaka} = FP^{1.25} = 625,57^{1.25} = \mathbf{3128,56}$$

Potencijalni broj grešaka za naš sistem iznosi oko 3128.

Pravilo 6 - Estimacija efikasnosti otklanjanja greške

Estimacija efikasnosti otklanjanja greške se vrši procena broja grešaka koje će biti otklonjene. Svaki put kada dođe do neke inspekcije, testiranja ili revizije će se otkloniti barem 30% grešaka koje postoje u datom trenutku.

Pravilo 7 - Estimacija efikasnosti organizovanog otklanjanja grešaka

Estimacija efikasnosti organizovanog otklanjanja grešaka uključuje kolektivnu inspekciju testiranja i otklanjanja grešaka, što obično iznosi to 70% otklanjanja grešaka u tom trenutku postojećih grešaka.

Ovaj način je efikasniji od prethodnog načina otklanjanja grešaka, ali je zato i skuplji i zahtevniji. Druga stavka ovog pravila glasi da svaka inspekcija koda će pronaći i otkloniti 65% u tom trenutku postojećih grešaka.

Pravilo 8 - Estimacija *post release* efikasnosti otklanjanja

Estimacija post release efikasnosti otklanjanja greške odnosno nakon puštanja sistema u rad kaže da programeri za održavanje mogu popraviti 8 bagova, odnosno grešaka po radničkom mesecu. Otklanjanje grešaka nakon puštanja je bilo prisutno u softverskoj industriji više od 30 godina. Ovo pravilo takođe kaže da dobro definisani procesi i alati dovode do poboljšanih kvaliteta odnosno rezultata ove efikasnosti.

Pravilo 9 - Estimacija trajanja realizacije projekta

Pravilo 9 procenjuje potencijalno vreme koje je potrebno kako bi se realizovao softver . Ovo računamo tako što broj funkcionalnih poena stepenujemo sa brojem 0,4.

Ukupno vreme = $FP^{0.4} = 625,57^{0.4} = \mathbf{13,13 \text{ kalendarskih meseci}}$

Broj meseci potrebnih za realizaciju, odnosno razvijanje sistema jeste oko 13 meseci.

Pravilo 10 - Estimacija potrebnih ljudi za realizaciju projekta

Ovo pravilo koristimo kako bismo procenili neophodan broj osoblja za razvoj softvera, što u startu znači da će ovo pravilo, odnosno broj varirati.

Estimacija potrebnih ljudi za realizaciju projekta iznosi tako što se podeli broj funkcionalnih poena sa 150.

Broj ljudi = $FP / 150 = 625,57 / 150 = \mathbf{4,17 - odnosno 4 čoveka}$

Pravilo 11 - Estimacija ljudi potrebnih za održavanje softvera

Estimacija potrebnih ljudi za održavanje projekta iznosi tako što se podeli broj funkcionalnih poena sa 750.

Broj ljudi = $FP / 750 = 625,57 / 750 = \mathbf{0,83 - odnosno 1 čoveka}$

Za naš softver je potrebno 1 čovek za održavanje sistema. Sada ćemo primeniti još jednu estimaciju ovog pravila, a to je izračunavanje životni vek korišćenja softvera, odnosno vreme koliko će se softver smatrati ažurnim i funkcionalnim za korišćenje i ne zastarelim. Procenu računamo na sledeći način:

Broj godina = $FP^{0.25} = 625,57^{0.25} = \mathbf{5.01 \text{ godina korišćenja}}$

Pravilo 12 - Estimacija napora

Estimacija napora razvoja softvera se računa tako što pomnožimo broj meseci potrebnih za razvoj sa brojem osoblja kako bismo predvideli broj osoblja-meseci napora. Ovo pravilo kao što vidimo uključuje zapravo 9.(meseci) i 10.(osoblje) pravilo.

Napor = Broj meseci * Broj osoblja = $13,13 * 4,17 = \mathbf{54.75 \text{ čovek-meseci}}$

3.2. Procena održavanja COCOMO II modelom

Isti alat koji smo koristili za procenu COCOMO II iskoristili smo i za procenu održavanja.

Maintenance

Annual Change Size (ESLOC) Maintenance Duration (Years)

Software Understanding (0%-50%) Unfamiliarity (0-1)

Software Labor Rates

Cost per Person-Month (Dollars)

Slika 3.2/1 COCOMO II model za procenu održavanja

Maintenance

Annual Maintenance Effort = 0.0 Person-Months

Annual Maintenance Cost = \$0

Total Maintenance Cost = \$0

Slika 3.2/2 COCOMO 2 rezultat

ESLOC predstavlja broj efektivnih linija koda i izračunava se po sledećoj formuli:

$$S_e = S_{new} + 0.75S_{mod} + 0.2S_{reused}$$

S_e is the number of effective source lines of code (ESLOC).

Na primeru NSI sistema to izgleda ovako:

$$ESLOC = 33000 + 0,75 * 7.000 + 0,2 * 58.000 = \mathbf{49.850}$$

Sve podatke za S_{new} , S_{mod} i S_{reused} možete videti na slici 2.2.3.6/ (onde gde je prikazan cocomo prvi put) gde S_{new} predstavlja broj novih linija koda, S_{mod} broj izmenjenih a S_{reused} broj ponovno iskorišćenih linija koda.

4. Menadžment rizika

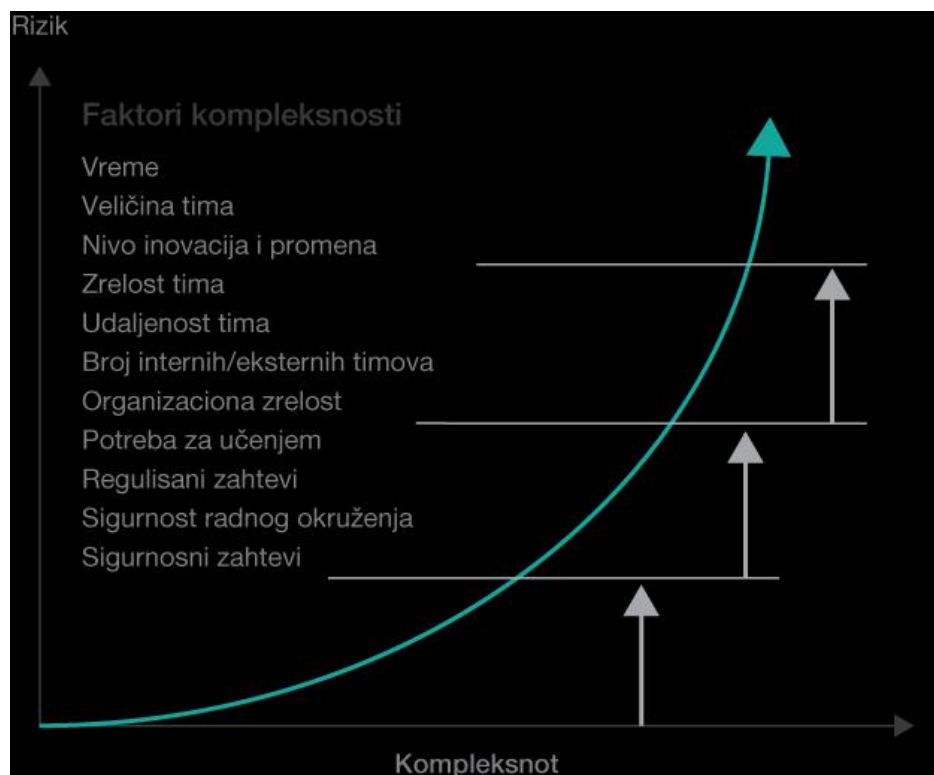
Planiranje rizika uključuje aktivnosti neophodne da se svaki rizik dovede pod kontrolu. Strategije i planovi za svaki rizik su sačinjeni kako bi postojala mogućnost da se izbegnu ili bar smanje posledice ostvarenja nekog rizika.

4.1 Opseg i namera RMMM aktivnosti

Rizik projekta je događaj ili uslov koji, ako se realizuje, ima pozitivan ili negativan efekt na jedan ili više ciljeva projekta, kao što su obim, raspored, troškovi i kvalitet. Može imati jedan ili više uzroka i ako se dogodi jedan ili više uzroka.

Upravljanje rizicima predstavlja kontinualan proces koji se sprovodi tokom čitavog životnog ciklusa projekta. Uopštena metodologija za upravljanje rizikom obuhvata sledece podprocese:

- Identifikacija rizika
- Analiza i procena rizika
- Planiranje izbegavanja rizika i reakcija na rizik
- Kontrola primene reakcija na rizik



Slika 4.1/1– Uticaj kompleksnosti na rizičnost projekta

4.2 Menadžment rizika organizacione uloge

Redni broj	Pozicija	Opis
1.	Viši menadžment	Viši menadžment mora da osigura da se neophodni resursi efikasno primenjuju da bi se razvile mogućnosti koje su potrebne za postizanje cilja. Moraju takođe proceniti i uključiti rezultate aktivnosti procene rizika u process donošenja konačne odluke. Efikasan program upravljanja rizicima procenjuje rizike misije vezane za IT zahteva podrsku i uključenost višeg menadžmenta.
2.	Šef tehnološkog sektora(CIO)	Šef tehnološkog sektora, odnosno CIO, je odgovoran za planiranje, budžetiranje i performanse IT agencije, uključujući komponente za bezbednost informacija. Odluke donešene u ovim oblastima bi trebalo da budu bazirane na efikasnom programu za upravljanja rizicima.
3.	Sistemske i informacione menadžeri	Sistemske i informacione menadžeri su odgovorni za obezbeđivanje da su pravilne kontrole primenjene na oblasti integriteta, poverljivosti i dostupnosti IT sistema i podataka čiji su oni vlasnici.
4.	Biznis i funkcionalni menadžeri	Menadžeri odgovorni za poslovne operacije i IT nabavke moraju da imaju aktivnu ulogu u procesu upravljanja rizika. Ovi menadžeri su osobe sa autoritetom i odgovornosti za donošenje odluka o trgovini neophodnim za ostvarivanje cilja. Njihovo učešće u procesu upravljanja rizicima omogućava postizanje odgovarajuće sigurnosti za IT sisteme, koji će, ukoliko se pravilno upravlja, obezbediti efikasnost ostvarivanja cilja sa minimalnim troškovima sredstava.
5.	ISSO	IT menadžeri bezbednosnih programa i službenici za bezbednost računara su odgovorni za sigurnosne programe svojih organizacija uključujući i upravljanje rizicima. Dakle, oni imaju mnogo veću ulogu u uvođenju odgovarajuće strukturane metodologije za identifikaciju, procenu, i umanjuje rizike za IT sisteme koje ih podržavaju.

6.	IT praktičari bezbednosti	IT praktičari bezbednosti su odgovorni za odgovarajuću implementaciju bezbednosnih zahteva u svojim IT sistemima. Kako se promene pojavljuju u postojećem IT sistemu, IT praktičari bezbednosti moraju podržati ili koristiti proces upravljanja rizika da identifikuju i procene nove potencijalne rizike i da sprovedu nove bezbednosne kontrole po potrebi radi zaštite svojih IT sistema.
7.	Instruktori za bezbednosnu svest	Organizaciono osoblje su korisnici IT sistema. Upotreba IT sistema i podataka u skladu sa organizacionim polisama, smernicama i pravilima ponašanja su od ključnog značaja zbog ublažavanja rizika i zaštite IT resursa organizacije. Da bi se rizik IT sistema umanjio, neophodno je da se sistemskim i aplikacionim korisnicima obezbede vežbe za povećanje svesti o bezbednosti. Stoga, instruktori za IT bezbednost ili profesionalci u ovoj oblasti moraju da razumeju proces upravljanja rizicima da bi mogli da razviju adekvatan material za obuku i uključiti procenu rizika u programe obuke za edukaciju krajnjih korisnika.

Tabela 4.2/2 Uloge i odgovornosti u RMMM[6]

U tabeli 4.2.2 prikazane su uloge i odgovornosti zaposlenih u okviru procesa upravljanja rizicima (RMMM).

4.3 Identifikacija i opis rizika

Identifikacija rizika je proces prepoznavanja potencijalnih rizika koji mogu nastupiti na određenom projektu i njihovog detaljnog opisivanja. Učesnici ovog procesa su najčešće projektni menadžer, članovi projektnog tima, stručnjaci za upravljanje rizicima, tim za upravljanje rizikom, krajnji korisnici, stakelholderi i drugi

Redni broj	Identifikacija izvora rizika	Verovatnoća rizika	Uticaj gubitaka usled rizika(C)	Izloženost rizika(RE)
1.	Loše dizajnirana arhitektura – Arhitektura koja je loše dizajnirana i njene greške se kasno u toku razvoja projekta otkrivaju.	0,05	80.000	4.000
2.	Opseg projekta – loše definisan projekta, cilj, plan... Ne shvatanje šta je klijent želeo i šta programer može da omogućiti.	0,30	10.000	3.000
3.	Odlazak zaposlenog - Zaposleni može iz različitih razloga da odustane od rada ili može biti sprečen.	0,09	6.500	585
4.	Testiranje - Ukoliko se testiranja ne vrše po planu ili ukoliko je veliki broj propalih testova, povećava se vreme rada na ispravka vezanim za njih.	0,40	10.000	4.000

Tabela 4.3/1 – Identifikacija i analiza rizika

Kao što je prikazano na tabeli 4.3/1 može se primetiti da najveću mogućnost rizika ima rizik pod rednim brojem 4. Dok rizik pod rednim brojem 1. Ima najveći uticaj gubitka usled nastanka rizika(80.000 jedinica u nekoj valuti). Rizici pod brojem 4. I 1. Imaju najveću izloženost rizika(po 4.000 jedinica). Samim tim su ovi rizici na prvom mestu.

Formula za izračunavanje izloženosti rizika glasi:

$$Re = P * I$$

Re - izloženost rizika

I - uticaj rizika(dinar,euro,jen,dolar)

P - verovatnoća rizika

Kako bismo odgovorili na planiran rizik treba definisati šta treba preduzeti usled nastanka tog rizika i kako ublažiti uticaj ili dolazak do istog.

Redni broj	Identifikacija izvora rizika	Akcija prevencije rizika
1.	Loše dizajnirana arhitektura – Arhitektura koja je loše dizajnirana i njene greške se kasno u toku razvoja projekta otkrivaju.	Dizajniranje arhitekture softvera će biti obrađivana od strane stručnjaka iz ove oblasti.
2.	Opseg projekta – loše definisan projekta, cilj, plan... Ne shvatanje šta je klijent želeo i šta programer može da omogućiti.	Napraviti specifikaciju i definisati cilj. Omogućiti svakom zaposlenom da ima plan i da sanjam može da razgovara o projektu da li je radnik u stanju da ostvari zahteve ili ne.
3.	Odlazak zaposlenog - Zaposleni može iz različitih razloga da odustane od rada ili može biti sprečen.	Najmanje još jedan zaposleni pored onog koji je zadužen za taj deo projekta treba da bude upućen za isti.
4.	Testiranje - Ukoliko se testiranja ne vrše po planu ili ukoliko je veliki broj propalih testova, povećava se vreme rada na ispravka vezanim za njih.	Praćenje rada testiranja i njene uspešnosti, promena rada ukoliko se primeti odstupanje.

Tabela 4.3/2 – Akcije prevencije rizika

Iz tabele 4.3/2 navodimo akcije prevencije rizika odnosno šta ćemo preduzeti kako do rizika ne bi došlo. Naravno ove akcije ne garantuju da se rizik neće pojaviti već smanjuju njegovu verovatnoću nastanka kao i gubitke koji bi se mogli ostvariti ukoliko se rizik ostvari.

Redni broj	Rizik	Akcija smanjenja gubitaka usled rizika	Akcija oporavka usled rizika
1.	Loše dizajnirana arhitektura – kasno otkrivanje i potreba za reinženjeringom ili odustajanje od projekta.	Reiženjering arhitekture i modifikacija low-level dizajna.	Edukacija arhitekta, obavezna primena paterna.
2.	Opseg projekta – loše definisan projekta, cilj, plan... Ne shvatanje šta je klijent želeo i šta programer može da omogućiti.	Popraviti specifikacije i sprovesti ih u akciju.	Uvođenje stručnog kadra koji će napraviti specifikacije projekta i koja će navesti tačno šta kako ko gde i kad treba da uradi.
3.	Odlazak zaposlenog - Zaposleni može iz različitih razloga da odustane od rada ili može biti sprečen.	Ako dođe do odlaska nekog od zaposlenih treba imati zamenu, što bi značilo da budu otvoreni za primanje zaposlenih na određeno vreme	Angažovanje freelancera ili nekih pouzdanih programera na kratke staze dok se ne reši pitanje zaposlenog za stalno ukoliko nismo u mogućnosti da zaposlimo nekoga odmah.
4.	Testiranje - Ukoliko se testiranja ne vrše po planu ili ukoliko je veliki broj propalih testova, povećava se vreme rada na ispravka vezanim za njih.	Ukoliko dolazi do kašnjenja uvođenje dodatnih vremenskih termina ili ukoliko je to moguće novih ljudskih resursa (u zavisnosti od vrste testiranja)	Povratak do poslednjih uspešnog testa i testiranje odatle.

Tabela 4.3/3 – Akcije smanjenja gubitaka i oporavka usled rizika

Ukoliko rizik nije uspešno zaustavljen treba razmotriti opcije nakon njegovog delovanja. Dakle treba definisati smanjenje gubitaka i oporavak posle rizika kao rezultat načinjene štete. Budući da se razvija ozbiljan softver ove stavke je neophodno definisati. Ako su ove akcije dobro napisane i kasnije iskorišćene njihove instrukcije projekat se može spasiti od propadanja i kompanija može uštedeti mnogo novca.

4.4 Tabela intenziteta rizika

Verovatnoća	Posledice				
	Beznačajno (6.000 <)	Malo (6.000-25.000)	Umereno (25.000-80.000)	Veliko (80.000-150.000)	Katastrofalno (>150.000)
Skoro izvesno >90%	5400	13.950	47.250	103.500	135.000
Verovatno (60% - 90%)	4.500	11.625	39.375	86.250	112.500
Umereno (25% - 60%)	2.550	6.600	22.500	48.875	63.750
Neizvesno (10% - 25%)	1.050	2.712	9.187	20.125	26.250
Retko 10% <	600	1.550	5.250	11.500	15.000

Tabela 4.4/1 – Klasifikacija rizika

Prihvaljivo		Legenda
Umereno		
Zabrinjavajuće		

Na osnovu prethodnih tabela sastavljena je tabela intenziteta rizika (tabela 4.4/1) na kojoj se vidi u kojoj grupi rizika pripadaju naši rizici.

4.5. Analiza identifikovanih rizika

Redni broj	Rizik	Klasa rizika
1.	Loše dizajnirana arhitektura – kasno otkrivanje i potreba za reinženjeringom ili odustajanje od projekta.	
2.	Opseg projekta – loše definisan projekta, cilj, plan... Ne shvatanje šta je klijent želeo i šta programer može da omogućiti.	
3.	Odlazak zaposlenog - Zaposleni može iz različitih razloga da odustane od rada ili može biti sprečen.	
4.	Testiranje - Ukoliko se testiranja ne vrše po planu ili ukoliko je veliki broj propalih testova, povećava se vreme rada na ispravka vezanim za njih.	

Tabela 4.5/1 – Podela klasa rizika softvera NSI

Na tabeli 4.5/1 od 4 identifikovana rizika dva spadaju u klasu umerenih rizika, što znači da i ako se ovi rizici ostvare neće imati uticaj na poslovanje kompanije a veoma male uticaje na razvoj samog softvera. Zelenom bojom su označena takođe dva identifikovana rizika što znači da ukoliko dođe do ostvarenja ovih rizika troškovi firme neće biti veliki i problem je izuzetno lako rešiv.










Kao što smo već definisali u prethodnim tabelama, dobar projektni menadžer treba da identifikuje rizik i njegov izvor, da ima spremnu akciju preventovanja i ukoliko prevencija ne uspe da ima akcije smanjenja gubitaka i akcije oporavka ukoliko želi da neometano nastavi process razvoja softvera i isporuči kvalitetan softver.

5. Raspored aktivnosti projekta

Raspored aktivnosti projekta je neophodan jer prikazuje koliko vremena će projekat trajati (duration u MS Project-u), koliko rada je potrebno utrošiti na njemu (work u MS Project-u) i koje resurse upotrebiti. Sve ovo je potrebno i neophodno. Ali ako nemamo grafički sistem ili neki grafički alat da sve to vizuelno predstavimo, možemo biti zatrpani informacijama, iz kojih ne možemo isplivati i koje nam ne pomažu da lakše upravljamo vremenom na projektu.

	Tas Mc	Task Name	Duratio	Start	Finish	Pre	Resource Names
1	★	▲ Razvoj NSI sistema	154 days	Mon 26-03-18	Thu 25-10-18		
2	→	▲ Planiranje projekta	6 days	Mon 26-03-18	Mon 02-04-18		
3	→	Procena veličine projekta	1 day	Mon 26-03-18	Mon 26-03-18		Project Manager
4	→	Procena sredstava za razvoj projekta	2 days	Tue 27-03-18	Wed 28-03-18	3	Project Manager
5	→	Planiranje razvoja softvera	2 days	Thu 29-03-18	Fri 30-03-18	4	Project Manager
6	→	Procena finansija	1 day	Mon 02-04-18	Mon 02-04-18	5	Project Manager
7	→	▲ Zahtevi projekta	10 days	Tue 03-04-18	Mon 16-04-18		
8	→	Prikupljanje zahteva	4 days	Tue 03-04-18	Fri 06-04-18	6	Analyst
9	→	Definisanje zahteva projekta	2 days	Mon 09-04-18	Tue 10-04-18	8	Analyst
10	→	Analiza definisanih zahteva	2 days	Wed 11-04-18	Thu 12-04-18	9	Analyst, Project Manager
11	→	Funkcionalni I nefunkcionalni zahtevi	4 days	Wed 11-04-18	Mon 16-04-18	9	Analyst, Project Manager
12	→	▲ Dizajniranje softvera sistema	13 days	Tue 17-04-18	Thu 03-05-18		
13	→	Pregled specifikacije	1 day	Tue 17-04-18	Tue 17-04-18	11	Analyst, Project Manager
14	→	Planiranje I dizajniranje arhitekture sistema	7 days	Wed 18-04-18	Thu 26-04-18	13	Back-End Developer, Project Manager
15	→	Dizajn grafičkog korisničkog interfejsa	4 days	Fri 27-04-18	Wed 02-05-18	14	Designer, Front-End Developer
16	→	Dizajniranje modela I baze podataka	5 days	Fri 27-04-18	Thu 03-05-18	14	Back-End Developer
17	→	▲ Implementacija softvera	10 days	Fri 04-05-18	Thu 17-05-18		
18	→	Obrada zahteva sistema	4 days	Fri 04-05-18	Wed 09-05-18	16,15	Back-End Developer, Front-End Developer
19	→	Pregled zahteva I dizajna sistema	2 days	Thu 10-05-18	Fri 11-05-18	18	Back-End Developer, Front-End Developer
20	→	Planiranje I raspoređivanje	2 days	Mon 14-05-18	Tue 15-05-18	19	Back-End Developer, Front-End Developer,

Slika 5/1 – Tabela procesa i podprocesa razvoja softvera - 1

21		Planiranje i organizacija razvoja softvera	2 days	Wed 16-05-18	Thu 17-05-18	20	Back-End Developer, Front-End Developer
22		▀ Razvoj softvera	55 days	Fri 18-05-18	Thu 02-08-18		
23		Implementacija baze podataka	5 days	Fri 18-05-18	Thu 24-05-18	21	Back-End Developer
24		Razvoj prototipa	10 days	Fri 25-05-18	Thu 07-06-18	23	Designer
25		Analiza prototipa i prikupljanje grešaka	5 days	Fri 08-06-18	Thu 14-06-18	24	Designer
26		Back-end programiranje	20 days	Fri 15-06-18	Thu 12-07-18	25	Back-End Developer
27		Izrada klijentske aplikacije	15 days	Fri 13-07-18	Thu 02-08-18	26	Front-End Developer
28		▀ Testiranje	20 days	Fri 03-08-18	Thu 30-08-18		
29		Alfa testiranje	10 days	Fri 03-08-18	Thu 16-08-18	27	Tester
30		Beta testiranje	10 days	Fri 17-08-18	Thu 30-08-18	29	Tester
31		▀ Dokumentacija	40 days	Fri 31-08-18	Thu 25-10-18		
32		Izrada dokumentacij	35 days	Fri 31-08-18	Thu 18-10-18	30	Technical Support
33		Izrada korisničkog uputstva	5 days	Fri 19-10-18	Thu 25-10-18	32	Technical Support
34		Pregled konačne dokumentacije	1 day	Fri 19-10-18	Fri 19-10-18	32	Technical Support
35		▀ Puštanje u rad	3 days	Mon 22-10-18	Wed 24-10-18	30	
36		Isporuka softvera	3 days	Mon 22-10-18	Wed 24-10-18	34	Project Manager, Technical Su
37		Održavanje	1 day	Thu 25-10-18	Thu 25-10-18	35	Back-End Developer, Front-Er

Slika 5/1 – Tabela procesa i podprocesa razvoja softvera - 2

Slika 5/2 nam prikazuje tabelu iz koje se jasno vidi koji se podprocesi nalaze u okviru koje faze projekta, takođe i koji tip zaposlenog radi na kojim podprocesima.

Tipovi zaposlenih
Project Manager
Management
Analyst
Back-End Developer
Designer
Front-End Developer
Tester
Technical Support

Slika 5/3 – Tabela tipova zaposlenih koji rade na projektu NIS

Svaki od zaposlenih ima odgovarajući tip posla na razvoju softvera za koji je odgovoran.

Na nekim procesima, kao što možemo videti i sa tabele (5/1 i 5/2), rade nekoliko tipova zaposlenih zajedno ili uporedno.

5.1 Vremenski okvir

Pod vremenskim okvirom definišemo vremenski interval svakog procesa I podprocesa, jer dobro upravljanje vremenom kao i dobra raspodela poslova mogu dovesti do znatne uštede novca kompaniji.

MODERN RESOLUTION FOR ALL PROJECTS					
	2011	2012	2013	2014	2015
SUCCESSFUL	29%	27%	31%	28%	29%
CHALLENGED	49%	56%	50%	55%	52%
FAILED	22%	17%	19%	17%	19%

The Modern Resolution (OnTime, OnBudget, with a satisfactory result) of all software projects from FY2011-2015 within the new CHAOS database. Please note that for the rest of this report CHAOS Resolution will refer to the Modern Resolution definition not the Traditional Resolution definition.

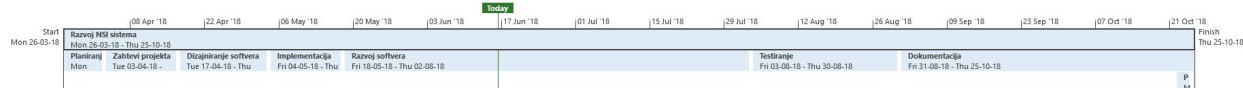
Slika 5.1/1 – Uspešnost softverskih projekata[7]

Kao što se može videti po tabeli za 2015. Godinu, znamo da svega 29% projekata bude u vremenskim I budžetskim rokovima, dok 52% probije budžet ili vremenski rok. 19% projekata propadne usled lošeg menadžmenta.

Task Name	Duration	Start	Finish
Razvoj NSI sistema	154 days	Mon 26-03-18	Thu 25-10-18
Planiranje projekta	6 days	Mon 26-03-18	Mon 02-04-18
Procena veličine projekta	1 day	Mon 26-03-18	Mon 26-03-18
Procena sredstava za razvoj projekta	2 days	Tue 27-03-18	Wed 28-03-18
Planiranje razvoja softvera	2 days	Thu 29-03-18	Fri 30-03-18
Procena finansija	1 day	Mon 02-04-18	Mon 02-04-18
Zahtevi projekta	10 days	Tue 03-04-18	Mon 16-04-18
Prikupljanje zahteva	4 days	Tue 03-04-18	Fri 06-04-18
Definisanje zahteva projekta	2 days	Mon 09-04-18	Tue 10-04-18
Analiza definisanih zahteva	2 days	Wed 11-04-18	Thu 12-04-18
Funkcionalni I nefunkcionalni zahtevi	4 days	Wed 11-04-18	Mon 16-04-18
Dizajniranje softvera sistema	13 days	Tue 17-04-18	Thu 03-05-18
Pregled specifikacije zahteva	1 day	Tue 17-04-18	Tue 17-04-18
Planiranje I dizajniranje arhitekture sistema	7 days	Wed 18-04-18	Thu 26-04-18
Dizajn grafičkog korisničkog interfejsa	4 days	Fri 27-04-18	Wed 02-05-18
Dizajniranje modela I baze podataka	5 days	Fri 27-04-18	Thu 03-05-18
Implementacija softvera	10 days	Fri 04-05-18	Thu 17-05-18
Obrada zahteva sistema	4 days	Fri 04-05-18	Wed 09-05-18
Pregled zahteva I dizajna sistema	2 days	Thu 10-05-18	Fri 11-05-18
Planiranje I raspoređivanje timova	2 days	Mon 14-05-18	Tue 15-05-18
Planiranje I organizacija razvoja softvera	2 days	Wed 16-05-18	Thu 17-05-18
Razvoj softvera	55 days	Fri 18-05-18	Thu 02-08-18
Implementacija baze podataka	5 days	Fri 18-05-18	Thu 24-05-18
Razvoj prototipa	10 days	Fri 25-05-18	Thu 07-06-18
Analiza prototipa I prikupljanje grešaka	5 days	Fri 08-06-18	Thu 14-06-18
Back-end programiranje	20 days	Fri 15-06-18	Thu 12-07-18
Izrada klijentske aplikacije	15 days	Fri 13-07-18	Thu 02-08-18
Testiranje	20 days	Fri 03-08-18	Thu 30-08-18
Alfa testiranje	10 days	Fri 03-08-18	Thu 16-08-18
Beta testiranje	10 days	Fri 17-08-18	Thu 30-08-18
Dokumentacija	40 days	Fri 31-08-18	Thu 25-10-18
Izrada dokumentacije	35 days	Fri 31-08-18	Thu 18-10-18
Izrada korisničkog uputstva	5 days	Fri 19-10-18	Thu 25-10-18
Pregled konačne dokumentacije	1 day	Fri 19-10-18	Fri 19-10-18
Puštanje u rad	3 days	Mon 22-10-18	Wed 24-10-18
Isporuka softvera	3 days	Mon 22-10-18	Wed 24-10-18
Održavanje	1 day	Thu 25-10-18	Thu 25-10-18

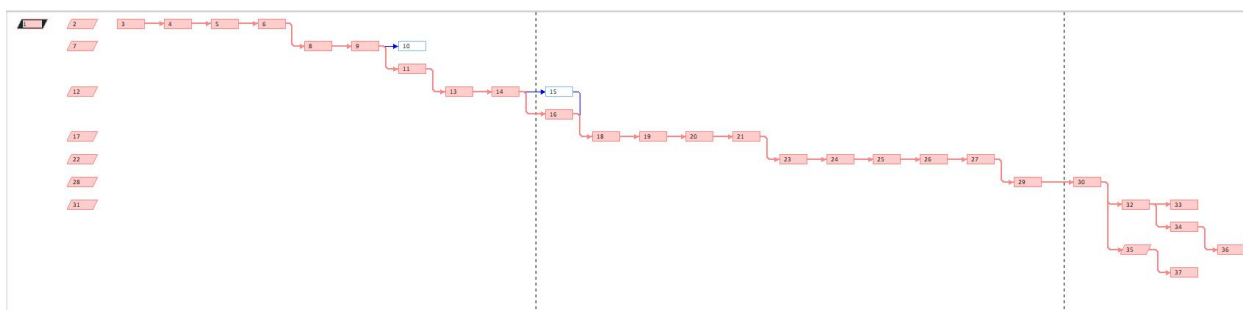
Slika 5.1/2 – Tabela vremenskog rasporeda po procesima i podprocesima

Predstavlja listu rokova za izvršenje zadataka I sadrži vreme trajanja zadatka, kao I datum početka I kraja izvršenja istog.



Slika 5.1/3 – Dijagram vremenske linije (Timeline dijagram)

Na osnovu datih tabela kreiraju se mrežni I Gantov dijagram. Mrežni dijagram predstavlja putanju procesa kroz razvoj softvera (kao I kritični put). Crvenom bojom se odvajaju najkritičnije tačke u razvoju softvera (Kritični put).



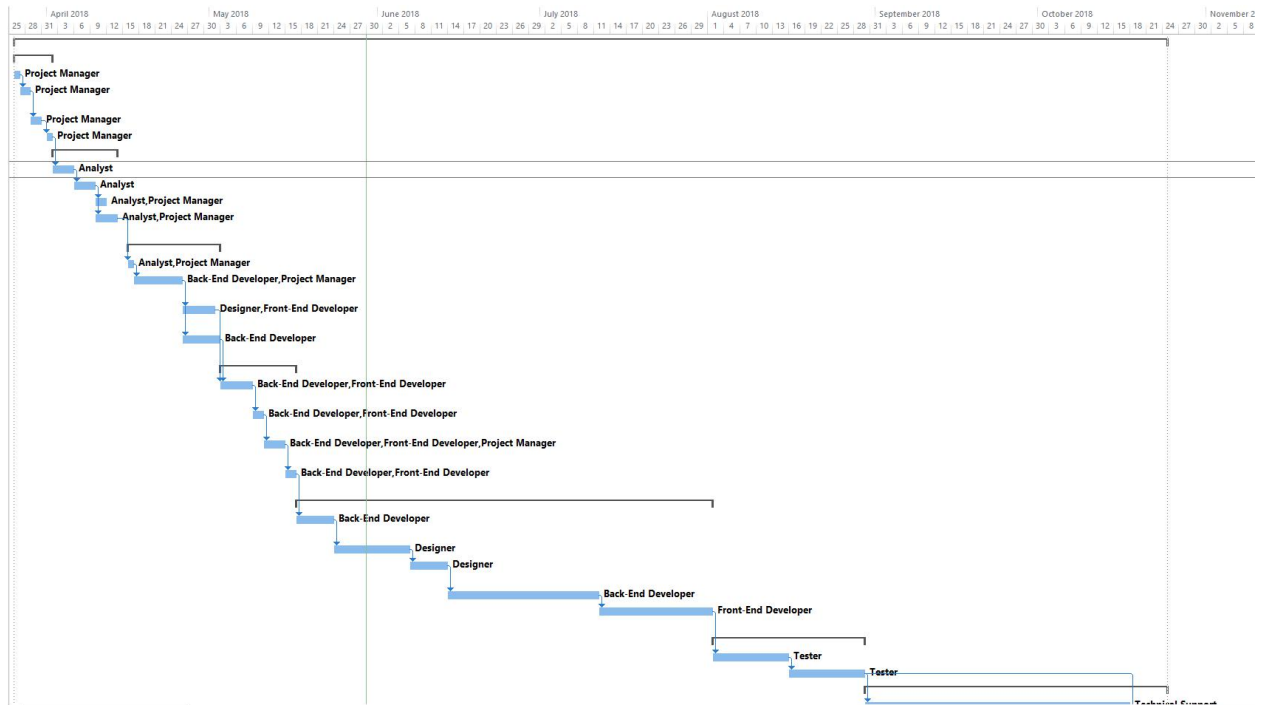
Slika 5.1/4 – Mrežni dijagram

Na osnovu Gantovog dijagrama (Slika 5.1/5) možemo zaključiti koji je vremenski interval projekta, procesa, koji su procesi, ko obavlja procese, koji je tip veze procesa...

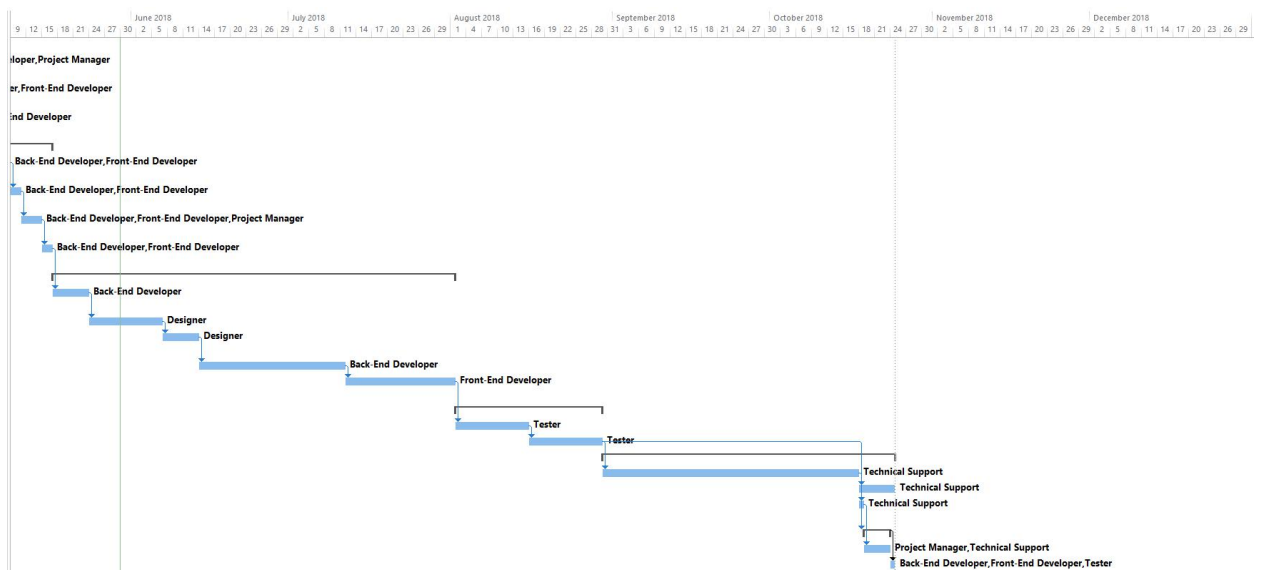
U Gantovom dijagramu postoje četiri tipa veze procesa:

1. StartToStart(SS) – oba počinju istovremeno
2. StartToFinish(SF) – kada prva aktivnost počne druga se završava
3. FinishToFinish(FF) – kraj prve aktivnosti označava I kraj druge
4. FinishToStart(FS) – prva aktivnost se završava a druga počinje

Najčešće korišćena veza u gantovom dijagramu za ovaj sistem je FS (FinishToStart), kada se prva aktivnost završi počinje druga.



Slika 5.1/5 – Gantt dijagram procesa - 1



Slika 5.1/5 – Gantt dijagram procesa - 2

5.2 Strukturalna podela posla (WBS)

WBS predstavlja hijerarhijsku dekompoziciju tj. Razlaganje ukupnog obima posla koje obavlja tim, kako bi se obavili projektni ciljevi i postigli adekvatni rezultati.

Počnemo tako što WBS dekomponujemo na manje zadatke ili taskove.

Ukoliko je WBS dobro osmišljen on nam može dosta pomoći pri proceni troškova, utvrđivanju rizika, raspoređivanju...

Neki od prednosti korišćenja WBSa su:

- Razbija rad u manje celine kojima se lakše rukuje
- Pruža osnovu za procenu troškova I raspodele ljudskih i drugih resursa
- Obezbeđuje vizuelni prikaz svih delova projekta



Slika 5.2/1 – WBS za sistem NSI

Sa slike se može videti da je WBS predstavljen u obliku stable. To stablo koje ima ukupan obim posla, hijerarhijski dekomponujemo na manje zadatke ili taskove, koji su dalje raščlanjeni na zadatke koji se izvršavaju u okviru tog zadatka. Alat koj se koristio za kreiranje WBS dijagrama možete pronaći u literaturi u poglavlju [10.3](#).

NSI sistem se sastoji iz 9 zadataka:

- Planiranje projekta (Definisati obim projekta, sredstva za razvoj, definisati plan razvoja, budžet)
- Zahtevi projekta (prikupiti korisničke zahteve, definisati funkcionalne I nefunkcionalne zahteve projekta)
- Dizajniranje softvera (Dizajniranje baze podataka, frontend, backend delova aplikacije, arhitekturu, modele podataka, tok podataka kroz aplikaciju)
- Implementacija softvera (pregledati specifikaciju zahteva, dizajn Sistema, identifikovati module Sistema, rasporediti timove)
- Razvoj softvera (Na osnovu prethodnih zadataka izvršiti kodiranje aplikacije, baze podataka, tokova podataka kroz aplikaciju, izvršiti debugovanje, testiranje)
- Testiranje (Kreiranje test slučajeva I testiranje)
- Dokumentacija (Dokumentovati projekat, korisnička uputstva, pregledati čitavu dokumentaciju)
- Puštanje u rad (Pripremiti I obaviti puštanje softvera u rad)
- Održavanje (Nastaviti podršku I održavanje softvera do daljnjeg)

6. Organizacija projektnog tima

U okviru 6. poglavlja, odnosno organizacije projektnog tima će biti navedena struktura kao i sve odgovornosti svih članova projektnog tima. Ovaj deo je od ključne važnosti, treba definisati dobru strukturu projektnog tima, dobro podeliti svima uloge, kao i odgovornosti koje dolaze sa konkretnim ulogama i u kojem timu će raditi, među članovima se posebno ističe projektni menadžer tima.

6.1 Struktura i odgovornosti članova tima

Član	Uloga člana	Odgovornost člana	Član tima
Nemanja Kuzmanović	Menadžer razvoja	Pruža dokumentaciju i informacije potrebne za razvoj projekta. Dodeljuje zadatke razvojnom timu.	Menadžment tim Razvojni tim
Ivan Ilić	Menadžer testiranja	Dokumentuje test plan, kao i informacije potrebne za testiranje. Vrši procenu efikasnosti testiranja. Dodeljuje zadatke test timu.	Menadžment tim QA
Sava Jeremić	Menadžer dizajna	Obezbeđuje dokumentaciju za potrebe razvoja softvera. Obaveštava menadžment projekta o rezultatima razvoja. Dodeljuje zadatke	Menadžment tim Razvojni tim

		developerima.	
Petar Perić	Tester	Testira	QA
Marko Marković	Tester	Testira	QA
Nikola Nikolić	Front-end Developer	Kodira front-end	Razvojni tim
Ana Anić	Front-end Developer	Kodira front-end	Razvojni tim
Boban Bobanović	Back-end Developer	Kodira back-end	Razvojni tim
Petar Petrović	Back-end Developer	Kodira back-end	Razvojni tim
Bojan Petrović	Grafički dizajner	Kreira grafički prikaz	Razvojni tim
Isidora Đokić	Analitičar	Bavi se analitikom i statistikom	QA
Kosta Kostić	Tehnička podrška	Pružna tehničke usluge	QA
Jovana Jovanović	Tehnička podrška	Pružna tehničke usluge	Isporuka
Ljubomir Lazić	Stručni konsultant	Pružna stručne savete iz svih oblasti projekta.	Menadžment

Tabela 6.1/1 - Definisane i podela uloga u procesu upravljanja promenama

7. Mehanizmi praćenja i kontrole

U okviru mehanizama praćenja i kontrole će biti opisana kontrola i menadžment promena, biće navedene i identifikovane sve uloge u procesu upravljanja promenama.

7.1 Kontrola i menadžment promena

Sledećim listingom ćemo identifikovati i podeliti uloge u procesu upravljanja promenama, kao i opis svake uloge i ko je zadužen za nju:

1. Direktor I zamenik upravljanja promenama - Ljubomir Lazić

Direktor upravljanja promenama treba da odobri i upravlja promenama sa minimalnim rizikom. Direktor takođe će sazvati CAB kako bi prodiskutovali o visokorizičnim promenama i zadužen je za donošenje odluke o tome da li da sprovede ili da odbije promene.

2. Inicijatori promena - Klijent

Svako može biti inicijator promene unutar organizacije, s tim što se mora razmotriti da li će ovo obuhvatiti baš sve korisnike, odnosno osigurati tako da samo relevantni ljudi mogu inicirati promenu. Ukoliko inženjer baze podataka inicira promenu, koji se smatra relevantnim, ta promena će se sprovedi ostalim inženjerima kao i direktoru za odlučivanje o tome da li će se ta promena sprovedi.

3. Savetodavni odbor za promene - CAB - Ljubomir Lazić, Sava Jeremić, Nemanja Kuzmanović, Ivan Ilić, Nikola Nikolić, Boban Bobanović

Savetodavni odbor za promene je grupa ljudi odnosno CAB tim, koji zajedno sa menadžerom promena deluju u svojstvu savetnika, oni raspoređuju promene po važnosti i riziku. Sastavljeni su od pojedinaca koji se smatraju relevantnim za donošenje odluka, sa ljudima od kojih je svako pojedinačno zadužen za svoj domen znanja.

4. Savetodavni odbor hitnih slučajeva za promene CAB/EC - Ljubomir Lazić, Sava Jeremić, Nemanja Kuzmanović, Ivan Ilić

To je grupa ljudi koja ima ugolu da odlučuje zajedno sa direktorom pri donošenju odluka koje su vezane za hitne promene. Pojedinci ove grupe su visoko rangirani u kompaniji.

5. Kreatori promena - Sava Jeremić

Kreatori promena koriste odgovarajuće tehnologije, stručnjaci koji će osigurati sav neophodan hardver, sotver, kao i lincence koje su na raspolaganju kako bi što pre kreirali promenu pre slanja na testiranje. Ovo može biti bilo koje osoblje iz IT sektora, kao što su front-end, back-end developer, dizajner baze podataka i slično. Oni obično preporučuju načine na koje treba da se testira softver nad kome je izvršena promena.

6. Tester promena - Ivan Ilić, Test tim

Tester promena, odnosno test inženjer, je zadužen da testira sve novo sprovedene promene koje su primenjene. Za ovo je preporučljivo da se ne koristi ista osoba koja je bila kreator promena, zbog boljeg i preciznijeg testiranja.

7. Implementator promena - Nemanja Kuzmanović

Implementator promena je stručnjak koji je zadužen da realizuje svoju kreiranu promenu, uz to je potrebno i da se kreira dokumentacija u obliku forme za promene.

8. Recenzent promena - Ivan Ilić, Sava Jeremić

Recezent promena predstavljaju grupu ljudi koje zajedno sa menadzerom promena vrše recenziju realizovanih promena koje su sprovedene u sistem, i zatvaraju zahtev za promenu, odnosno zatvaraju čitav slučaj.

Bitno je naglasiti da je inicijator za izmene klijent, odnosno korisnik softvera, jer će upravo oni slati zahteve za uvođenje novih funkcionalnosti, ili izmene već postojećih, kako koriste naš sistem će moći nama uvek javiti sve. Direktor i CAB su zaduženi za procenjivanje rizika promene, i oni odlučuju da li će doći uopšte do promene.

Rečnik pojmova

Pojam	Značenje
Promena	Dodavanje, modifikacija ili uklanjanje odobrenih, podržanih ili bazičnih hardvera, mreža, softvera, aplikacija, okruženja, sistema, izrađene dokumentacije.
Savetodavni odbor za promene CAB	Grupa ljudi koji mogu dati stručne savete vezane za menadžment promena i kako implementirati promene.
Kontrola promena	Procedura osiguranja da su sve promene kontrolisane, uključujući i podnošenje, analizu, odluku, odobrenje i implementaciju kao i post implementaciju promena.
Istorija promena	Provereni podatak koji je sačuvan, na primer, šta je urađeno, šta će kada biti urađeno, ko će ga uraditi i zašto?
Menadžment promena	Proces kontrole promena u infrastrukturi ili u bilo kom aspektu usluga, na kontrolisani način se omogućavaju promene sa minimalnom štetom.
Zahtev za promene (RFC)	Forma koja se koristi kako bi se sačuvali detalji zahteva za promene.

Tabela 7.1.1/1 - Rečnik pojmova uloga upravljanja promenama

Sada ćemo prikazati tabelu analize uticaja promene na određene bitne faktore(tabela 7.1/2).

Uticaj	Nema Rizika	Nizak	Umeren	Visok
Korisnik-Klijent	Nema uticaja na korisnike, kao i na kritične sisteme i servise.	Ima uticaj na minimalni broj korisnika, minimalni uticaj parcijalni deo sistema i ne kritičnog servisa.	Ima uticaja na nekoliko korisnika, značajan prekid kritičkih sistema ili kritičnih servisa.	Ima uticaja na veliki broj korisnika, kao i enorman prekid na kritičke delove sistema i na kritične delove servisa.
Uticaj IT resursa	Uključuje jedan IT resurs u okviru jedne radne grupe.	Uključuje IT resurse u okviru jedne radne grupe.	Uključuje IT resurse u više od dve radne grupe kao i stručnjaka.	Uključuje IT resurse u više radnih grupa kao i stručnjaka.
Kompleksnost implementacije	Tip promene - održavanje.	Niska kompleksnost, ne zahteva tehničku koordinaciju.	Značajna kompleksnost, zahteva samo tehničku koordinaciju.	Visoka kompleksnost, zahteva tehničku i biznis koordinaciju.
Trajanje promena	Ne očekuje se prekid.	Prekid se pomera na manje od jednog sata u vremenu koje nije kritično, utiče na klijente u vremenu koje nije kritično.	Prekid se pomera na manje od jednog sata u vremenu koje je kritično ili duže od 1 sata u vremenu koje nije kritično.	Prekid traje duže od jednog sata u vremenu koje je kritično, utiče na klijente u vremenu koje je kritično. Instalacija traje dugo. Backout neophodan.
Sigurnost	Backout plan nije potreban.	Nema sigurnosnih problema, kratak backout plan je	Utiče na nekritične podatke ili sigurnost servera, i sadrži	Utiče na kritične podatke ili sigurnost servera, i sadrži visok backout

		neophodan.	umeren backout plan koji ne prekoračava vremenski rok.	plan koji produžava vremenski rok.
Uticaj SLA(<i>Service Level Agreement</i>)	Nema uticaja na SLA vremena.	Mali uticaj.	Utiče na SLA u toku ne kritičnih vremena.	Ima uticaj na SLA u toku kritičnih vremena.

Tabela 7.1/2 - Tabela uticaja promene na bitne faktore sistema [8]

Tabela 7.1/2 predstavlja analizu koja prikazuje dodatne informacije koje su potrebne za planiranje promene kao i bitne informacije za direktora promene I CAB tim, kako da sprovedu način upravljanja promene.

Tabela koja je podeljena na nivoe uticaja, je podeljena u četiri nivoa ozbiljnosti uticaja, tako je i bojama obeleženo (Nema rizika, Nizak, Umeren i Visok). Za svaku situaciju, odnosno nivo ozbiljnosti za svaki uticaj promene je opisano koliko utiče na sistem i vreme prekida i slično.

U sledećoj tabeli 7.1/3 ćemo dati primer definisanja jedne promene za naš sistem.

Promena FC1		
Funkcionalnost	Prioritet	Opis
Mogućnost dodavanja nove kartice.	Umeren.	Potrebno je implementirati i dizajnirati GUI za profil korisnika, u okviru kojeg će moći dodavati više svojih kreditnih kartica, tako da može birati kada će koju, i zašta u budućnosti koristiti prilikom plaćanja.

Tabela 7.1/3 - Rečnik pojmova uloga upravljanja promenama

Tabela 7.1/3 predstavlja primer jednog zahteva za promenom. Svaki zahtev je obeležen svojom šifrom **FC**(Functional Change)**X** - gde je **X** redni broj promene. U ovom slučaju pravimo FC1 odnosno funkcionalnu promenu rednog broja 1.

Sada ćemo u tabeli 7.1/4 obeležiti uticaj FC1 promene na sve prethodno navedene faktore iz tabele 7.1/2.

Uticaj promene FC1 na kritične faktore sistema						
Faktori	Korisnik-Klijent	Uticaj IT resursa	Kompleksnost implementacije	Trajanje promena	Sigurnost	Uticaj SLA
Nivo uticaja	Umeren	Umeren	Nizak	Nizak	Visok	Visok

Tabela 7.1/4 - Uticaj promene FC1 na sistem

Analiza uticaja promene FC1	
Uticaj	Opis
Korisnik-Klijent	Uticaj na korisnika je umeren zato što se uvođenjem nove kartice ne menjaju kritične funkcionalnosti sistema, već je opciono za korisnika da odabere drugu karticu, nije niskog uticaja zato što ipak dodaje određenim korisnicima veliki benefit.
Uticaj IT resursa	Uticaj na IT resurse je takođe umeren, zato što je potrebno da uključimo developere iz front-end tima, back-end, kao i dizajnera koji će implementirati novu promenu.
Kompleksnost implementacije	Uticaj na kompleksnost implementacije je nizak zato što već postoji struktura, kod i način za implementiranje kartice, na osnovu već implementirane primarne kartice, tako da već postoji rešenje za ovaj problem.
Trajanje promena	Trajanje promena je nisko, odnosno manje od 1 sata.
Sigurnost	Utiče na kritične podatke sistema, jer je u pitanju rad sa kreditnim karticama.
Uticaj SLA(<i>Service Level Agreement</i>)	Na SLA utiče u kritičnom vremenu.

Tabela 7.1/5 - Uticaj promene FC1 na sistem

Možemo zaključiti da je backout plan poprilično neophodan jer se radi o promenama koje utiču na sigurnost osetljivih informacija klijenata.

Izveštaj promene FC1	
Status zahteva za promenom(RFC)	Uspešno izvršen.
Datum komunikacije	25.11.2018.
Planirani datum promene	1.12.2018.
Datum promene je implementiran	1.12.2018.
Problemi	Nije bilo problema.
Backout plan	Backout plan: treba predvideti probleme koji bi nastali u toku implementacije, kao i pripremiti plan sprečavanja istih ukoliko dođe do problema, kao što je otkaz sistema, neautorizovan upad u sistem i slično. Korisnički podaci treba da budu zaštićeni.
Backout plat je ovlašćen od strane	Ljubomir Lazić šef tima CAB Tim
Uspešno izvršena promena	1.12.2018.

Tabela 7.1/6 - Izveštaj promene FC1

Revizija promene FC1	
Datum revizije	2.12.2018
Ishod promene	Promena je uspešno implementirana. Korisnici sistema su zadovoljni novom funkcionalnošću.
Promenu pregledao	Ivan Ilić
Backout plan	Nije korišćen
Promena - procedura	Ova promena se smatra procedurom iz razloga što je implementacija ovog tipa već postojala.
Zaključak	2.12.2018.

Tabela 7.1/7 - Revizija promene FC1

8. Upravljanje komunikacijom na projektu

Pod ovim poglavljem se identifikuju stejkholderi u projektu, njihovi zahtevi, kao i elementi koji se obrađuju i distribuiraju tokom procesa komunikacije. Pvi korak u ovom poglavlju koji je potreban da se preduzme jeste identifikovanje stejkholdera ovog Sistema. To su osobe koje koriste sistem (direktno ili indirektno).

Stejkholderi sistema
Razvojni tim
Korisnik
Menadžment tim
QA

Tabela 8/1 – Stejkholderi NSI sistema

Tabela iznad prikazuje stejkholdere sistema, i postoji mogućnost da se vremenom ova tabela menja i proširuje.

Opis uloga stejkholdera	
Razvojni tim	<ul style="list-style-type: none"> • Softver arhitekta • Softverski inženjeri • Frontend Developeri • Backend Developeri • Dizajneri
Menadžment tim	<ul style="list-style-type: none"> • CAM odbor • Projektni menadžeri • Menadžeri • Direktori
QA tim	<ul style="list-style-type: none"> • Test inženjeri • Analitičari • Tehničko osoblje • Tim za isporuku softvera • Tim za održavanje softvera

Tabela 8/2 – Uloge stejkholdera

U tabeli iznad su opisane uloge stejkholdera, tačnije, koje uloge tima spadaju u koju grupu stejkholdera, radi lakšeg razumevanja i objašnjenja samih uloga i samih stejkholdera.

Identifikovani informacijski zahtevi:

Stejkholder	Opis	Način komunikacije	Kada (Koliko frekventno)
Razvojni tim	Raspored	Pismeno, elektronski	Nedeljno
	Planovi	Elektronski, usmeno	Mesečno, Nedeljno
	Ažuriranje statusa	Usmeno	Dnevno
	Izveštaj o napredku rada	Pismeno, elektronski	Dnevno
	Izveštaj sa Sastanka	Pismeno, elektronski	Nakon sastanka
Korisnik	Pretražuje igre na sistemu, kupuje igre, dodaje igre u korpu, izlistava sve igre...	Elektronski	Dnevno
Menadžment	Raspored	Pismeno, elektronski	Nedeljno
	Planovi	Elektronski, usmeno	Mesečno, Nedeljno
	Ažuriranje statusa	Usmeno	Dnevno
	Izveštaj o napredku rada	Pismeno, elektronski	Dnevno
	Izveštaj sa Sastanka	Pismeno, elektronski	Nakon sastanka
Projektni Menadžer	Dokument specifikacije	Pismeno, elektronski	Nakon kreiranja ili izmene
	Raspored	Pismeno, elektronski	Nakon sastanka
	Izveštaj sa Sastanka	Usmeno, elektronski	Nakon sastanka
	Statusni izveštaji	Usmeno	Mesečno
	Izveštaj o napredku rada	Elektronski	Dnevno
	Planovi	Elektronski, usmeno	Nakon kreiranja ili izmene

Tabela 8/3 – Informacioni zahtevi stejkholdera

U okviru tabele 8/3 uočavamo prethodno definisane stejkholdere, opisujemo zahteve, način komunikacije stejkholdera i frekvenciju komunikacije (koliko frekventno) sa sistemom.

Prikaz distribucije

Sledeća matrica distribucije ilustruje koji učesnici projekta de primati određene elemente komunikacije:

Dokument	Metod distribucije	Odobrenje menadžera	Menadžer projekta	Analitičar	Developer	Tester	Tehničko osoblje	Tim za isporuku i održavanje
Nedeljni status sastanka	EMAIL	X	X	X	X	X	X	X
SRS	EMAIL, MAIL	X	X	X		X	X	
SDS	EMAIL, MAIL	X	X		X	X		
SPMP	EMAIL, MAIL	X	X					
STP	EMAIL, MAIL	X	X	X	X	X		
SQAP	EMAIL, MAIL	X	X		X	X		
SDP	EMAIL, MAIL	X	X			X		
SPD	EMAIL, MAIL	X	X		X	X		
Izveštaji o performansama	EMAIL, MAIL	X	X	X	X	X	X	X
Ažuriranje statusa projekta	EMAIL, MAIL	X	X	X	X	X	X	X

Tabela 8/4 – Matrica učesnika projekta

U tabeli 8/4 prikazani su sledeći dokumenti:

- **SRS** (Software Requirements Specification) – Specifikacija softverskih zahteva
- **SDS** (Software Design Specification) – Specifikacija softverskog dizajna
- **SPMP** (Software Project Management Plan) – Plan upravljanja softverskim projektima
- **STP** (Software Test Plan) – Test plan softvera
- **SQAP** (Software Quality Assurance Plan) – Plan osiguranja kvaliteta softvera
- **SDP** (Software Development Plan) – Plan razvoja softvera

Takođe, u istoj tabeli, oznaka X predstavlja sva dokumenta i lica koja imaju direktan pristup. Kolona Metod distribucije opisuje način na koji se šalje, gde email predstavlja elektronski oblik komunikacije, a mail predstavlja štampani oblik.

Plan dokumentacije

Sledeća tabela prikazuje plan dokumentacije za dokumente koji se isporučuju kao rezultati projekta, a i oni koji nisu planirani za isporuku.

Dokument	Pripremio	Potvrdio	Datum pregleda	Verzija	Distribucija
Instalaciona dokumentacija	Tehničar dokumentacije	Developer	05.06.2018.	1.0	Repozitorijum za dokumenta, Developer, Tehničar dokumentacije, Projektni menadžer
Korisnička dokumentacija	Tehničar dokumentacije	Projektni menadžer	05.06.2018.	1.0	Repozitorijum za dokumenta, Projektni menadžer
Software	Zahtevi	Zahtevi	10.06.2018.	1.0	Dokument
Sastanci projektnog tima	Projektni menadžer	Učesnici sastanka	24h nakon sastanka	N/A	Repozitorijum za dokumenta, učesnici sastanka
Agenda sastanaka projektnog tima	Projektni menadžer	Učesnici sastanka	24h nakon sastanka	N/A	Repozitorijum za dokumenta, učesnici sastanka,

Tabela 8/5 – Plan dokumentacije

Plan rešavanja problema

Svi problemi moraju biti prosleđeni rukovodiocu projekta korišćenjem forme koja je dostupna u projektu. Kada je kompletirana, forma se se prosleđuje elektronski putem emaila.

Uloge:

Sledeća tabela ilustruje uloge članova projektnog tima koje učestvuju u rešavanju problema

Funkcija tima	Uloga(e)
Menadžer projekta	<ul style="list-style-type: none">• Prima nove izveštaje problema• Organizuje sastanke• Sastavlja document o ukupnoj sumi svih problema
Menadžer konfiguracija	<ul style="list-style-type: none">• Mora da učestvuje u sastancima gde se rešavaju problem• Analizira uticaj rešenja problema na druge konfiguracione stavke
Analitičar kvaliteta (QA)	<ul style="list-style-type: none">• Mora ušestvovati na sastancima gde se rešavaju problem• Sakuplja informacije o deficitima procesa koji su doveli do nastanka problema

Tabela 8/6 – Prikaz plana

9. Zaključak

U okviru ove dokumentacije smo naveli sve potrebne aspekte sa strane upravljanja projektima softvera, odnosno skup aktivnosti i veština koje projektni menadžer koristi kako bi projekat učinio efektivnim, efikasnim, i koji predstavlja detaljan pregled i strukturu softverskog projekta. Taj pregled i ta struktura uključuje procenjivanje veličine softvera, procenjivanje troškova i resursa sistema koji su potrebni. Uz navedene troškove i resurse, tako upravljamo i sa rizicima i promenama koje mogu i koje će nastati u toku procesa razvijanja softvera, tako dok se sve ove aktivnosti obavljaju, se upravlja i komunikacijom između resursa u razvoju softvera.

Ovaj dokument predstavlja opštu ideju i početnički dokument kakav bi otprilike pravi projektni menadžer sastavljao, koji treba biti i te kako načitan i pun znanja i iskustva u upravljanju projektima. Na ovom projektu smo se naučili komunikaciji i raspodelu međusobnih resursa, odnosno vremena za izradu ove dokumentacije, i imali dobar timski rad gde su svi jednako radili, i svako postizao svoj zadati cilj koji je imao da ispuni. Rad u timu, kao i timski rad u timu, je takođe jedan od najbitnijih aspekata i osobina koje svaki projektni tim treba posedovati, jer bez timskog rada, se ne može postići cilj.

10. Literatura

[1] "The Economics of Software Quality" - Capers Jones, Olivier Bonsignour - Strana 8, poglavlje Defining Software Quality and Economic Value

Link: <http://ptgmedia.pearsoncmg.com/images/9780132582209/samplepages/0132582201.pdf>

[2] - QSM

Link: <http://www.qsm.com/resources/function-point-languages-table>

[3] - Nastavni materijali predmeta SE325 Upravljanje projektima razvoja softvera - L04 - TEHNIKE PROCENJIVANJA NA SOFTVERSKOM PROJEKTU - Poglavlje 1 - Procena napora razvoja softvera preko FP - METOD FUNKCIONALNIH TAČAKA (POENA) FP -

Prof. Dr Ljubomir Lazić, Boro Mijović, 2017/18 - <http://lams.metropolitan.ac.rs>

[4] - The Cost of Software Development in Europe IT Services Market Research

Link: <https://www.cleveroad.com/blog/the-cost-of-software-development-in-europe--it-services-market-research>

[5] - "SOFTWARE ESTIMATING RULES OF THUMB" - Capers Jones

Link: <http://studylib.net/doc/18248112/software-estimating-rules-of-thumb>

[6] - "Risk Management Guide for Information Technology Systems" - Gary Stoneburner, Alice Goguen, and Alexis Feringa - Strana 6, poglavlje - 2.3 KEY ROLES

Link: <https://www.archives.gov/files/era/recompete/sp800-30.pdf>

[7] - Chaos Report

<https://www.infoq.com/articles/standish-chaos-2015>

[8] - "Sample IT Change Management Policies and Procedures Guide" - Evergreen Systems, Inc. 2007. - Strana 14, poglavlje - 4.3.3 Business Risk and Impact Analysis

Link: <https://www.sans.org/summit-archives/file/summit-archive-1493830822.pdf>

11. Prilozi i alati

10.1 Prilozi

Prilog 1: SE325-PZ-NSI.docx

Prilog 2: SE325-PZ-Master-Test-Plan.docx

Prilog 3: SE325-PZ-NSI.pdf

Prilog 4: SE325-PZ-Master-Test-Plan.pdf

10.3 Korišćeni softverski alati

- Kalkulator funkcionalnih poena - http://groups.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/harvey/FP_Calc.html
- COCOMO II kalkulator - <http://csse.usc.edu/tools/COCOMOII.php>
- WBS Diagram - <https://planhammer.io/wbs-chart-software.html>
- MicrosoftProject - <https://products.office.com/en/project/project-and-portfolio-management-software>
- SAP PowerDesigner - <https://www.sap.com/products/powerdesigner-data-modeling-tools.html>