Analiza vjerojatnosti gubitaka i čekanja u ovisnosti o promjenjivom prometnom opterećenju

Goldner, Hrvoje

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI

Hrvoje Goldner

ANALIZA VJEROJATNOSTI GUBITAKA I ČEKANJA U OVISNOSTI O PROMJENJIVOM PROMETNOM OPTEREĆENJU

ZAVRŠNI RAD

Zagreb, 2020.

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI
ODBOR ZA ZAVRŠNI RAD

Zagreb, 17. ožujka 2020.

Zavod: **Zavod za informacijsko komunikacijski promet**
Predmet: **Tehnologija telekomunikacijskog prometa I**

ZAVRŠNI ZADATAK br. 5563

Pristupnik: **Hrvoje Goldner (0135242494)**
Studij: Promet
Smjer: Informacijsko-komunikacijski promet

Zadatak: **Analiza vjerojatnosti gubitaka i čekanja u ovisnosti o promjenjivom prometnom opterećenju**

Opis zadatka:

Opisati temeljne prometne veličine koje se koriste za određivanje mjerila izvedbe neke komunikacijske mreže i njihove međuodnose, kao i podvorbene modele pogodne za opisivanje i modeliranje različitih realnih sustava. Nakon opisa značajki Erlangove B i C formule (modela) prikazati slučajeve njihove uporabe. Razviti aplikaciju koja, nakon korisničkog unosa prometnih veličina, računa iznos vjerojatnosti blokiranja ili čekanja.

Mentor:

doc. dr. sc. Marko Matulin

Predsjednik povjerenstva za završni ispit:

Sveučilište u Zagrebu
Fakultet prometnih znanosti

ZAVRŠNI RAD

ANALIZA VJEROJATNOSTI GUBITAKA I ČEKANJA U OVISNOSTI O PROMJENJIVOM PROMETNOM OPTEREĆENJU

LOSS AND WAITING PROBABILITY ANALYSIS IN RELATION TO CHANGING TRAFFIC LOADS

Mentor: doc. dr. sc. Marko Matulin

Student: Hrvoje Goldner
JMBAG: 0135242494

Zagreb, rujan 2020.

SAŽETAK

Vjerojatnost blokiranja predstavlja vjerojatnost da su, za zadani intenzitet prometa i broj poslužitelja, u sustavu posluživanja s gubicima, svi poslužitelji zauzeti, te da se daljnji zahtjevi ne mogu obraditi, dok je vjerojatnost čekanja primjenjiva u sustavima posluživanja bez gubitaka, te predstavlja vjerojatnost da će zahtjev naići na zauzete poslužitelje te biti stavljen u red čekanja. Opisani su podvorbeni modeli, formule za računanje vjerojatnosti blokiranja i čekanja te izrada računalne aplikacije za izračun istih.

Ključne riječi: vjerojatnost blokiranja, vjerojatnost čekanja, Erlang B, Erlang C, podvorbeni sustavi, računalna aplikacija, kalkulator

SUMMARY

The blocking probability is the probability in loss systems that, for a given traffic intensity and number of servers, all servers are busy and the request is not serviced, while the waiting probability represents the probability in the waiting systems for all servers to be busy, and the request will be put in queue. Queueing systems and formulas for calculating blocking and waiting probability are described, as well as the development process of the computer application for calculating the aforementioned probabilities.

Keywords: blocking probability, waiting probability, Erlang B, Erlang C, queueing systems, computer application, calculator

SADRŽAJ

1. Uvod

Blokiranje i čekanje su fenomeni koji se pojavljuju u sustavima posluživanja s većim brojem korisnika od broja poslužitelja. Vjerojatnost blokiranja i vjerojatnost čekanja moguće je procijeniti za određeni sustav pomoću razvijenih modela i formula.

Cilj ovog rada je prikazati modele za opis stvarnih sustava, veličine koje utječu na kvalitetu pružanja usluge, te alate za dimenzioniranje sustava u cilju optimizacije i smanjenja negativnih utjecaja – vjerojatnosti blokiranja i čekanja.

Ovaj rad sastoji se od 7 poglavlja u kojima se prikazuju temeljne veličine, vrste, parametri koji utječu na rad sustava posluživanja te izrada kalkulatora za računanje parametara i dimenzioniranje sustava posluživanja:

1. Uvod,
2. Temeljne prometne veličine i međuodnosi,
3. Primjena različitih podvorbenih modela za opis realnih sustava,
4. Značajke i primjena Erlangove B formule gubitaka,
5. Sustavi posluživanja s čekanjem i Erlangova C formula,
6. Aplikacija za izračun vjerojatnosti gubitaka i čekanja,
7. Zaključak.

U drugom poglavlju definira se promet i njegove vrste, opisane su osnovne prometne veličine, njihove mjerne jedinice te formule i načini računanja, korištene u dimenzioniranju telekomunikacijskih sustava posluživanja.

U trećem poglavlju opisuju se podvorbeni sustavi te se navode modeli koji se koriste za opisivanje i modeliranje sustava posluživanja, te njihove karakteristike i način označavanja.

Četvrto i peto poglavlje bavi se pojedinostima formula za računanje vjerojatnosti blokiranja i čekanja, te načinima korištenja, uvjetima i potrebnim parametrima za njihovo računanje.

Šesto poglavlje sadrži detaljan opis grafičkog sučelja, načina korištenja te programskog koda aplikacije za računanje vjerojatnosti blokiranja i čekanja, te pojedinostima odabranog jezika i razvojnog okruženja.

2. Temeljne prometne veličine i međuodnosi

Telekomunikacija se može definirati kao prenošenje, odašiljanje ili prijem znakova, signala, zapisa, slika, zvuka ili bilo kojeg drugog tipa predstavljanja informacije pomoću akustičnih, električnih, elektroničnih ili elektromagnetnih sredstava, [1].

Primjena teorije vjerojatnosti u planiranju, održavanju, promatranju performansi te radu telekomunikacijskih sustava naziva se teorija teleprometa, čiji cilj je učiniti promet mjerljivim definiranim jedinicama kroz matematičke modele te iz istih izvesti vezu između razine usluge (*Grade of Service* – GoS) i kapaciteta sustava tako da teorija postane alat za planiranje investicija. Zadatak teleprometnog inženjerstva je, koristeći teoriju teleprometa, odrediti metode upravljanja koje će osigurati da stvarna razina usluge ispunjava zahtijevanu te također skup izvanrednih radnji u slučaju preopterećenja sustava ili tehničkih poteškoća na istom, [2].

2.1. Promet

Telekomunikacijski promet (telepromet) može se općenito definirati kao skup događaja vezanih za potraživanje korištenja resursa telekomunikacijske mreže, [3].

U teoriji teleprometa, pojam „promet“ koristi se kao sinonim za intenzitet prometa te se još naziva i prometnim opterećenjem, a predstavlja broj zauzetih resursa u danom trenutku, [4].

Iz izraza (1), promet je uvijek jednak umnošku broja poziva ili zahtjeva (λ) i prosječnog vremena zadržavanja, tj. vremena posluživanja (T_s). Mjerna jedinica za intenzitet prometa je erlang (simbol: Erl). Jedan erlang predstavlja intenzitet prometa dovoljan da u skupini dostupnih poslužitelja drži zauzetim jedan poslužitelj tijekom cijelog vremena promatranja, [4, 5].

$$A = \lambda \cdot T_s \tag{1}$$

Gdje je:

- A – promet (erlang),
- λ – intenzitet nailazaka zahtjeva – broj zahtjeva ili poziva u jedinici vremena,
- T_s – prosječno vrijeme posluživanja zahtjeva ili poziva (vremenska jedinica).

Nadalje, prema [6], intenzitet nailazaka zahtjeva se može raščlaniti na umnožak broja terminala, tj. korisnika i prosječnog broja zahtjeva ili poziva po terminalu ili korisniku kako slijedi:

$$\lambda = N_{ptp} \cdot n_c \quad (2)$$

gdje je:

- N_{ptp} – broj terminala, tj. korisnika,
- n_c – prosječan broj zahtjeva ili poziva po terminalu ili korisniku u vrijeme promatranja.

Prema [4], prosječni intenzitet prometa u ovisnosti je od trenutnog intenziteta prometa na način prikazan u izrazu (3):

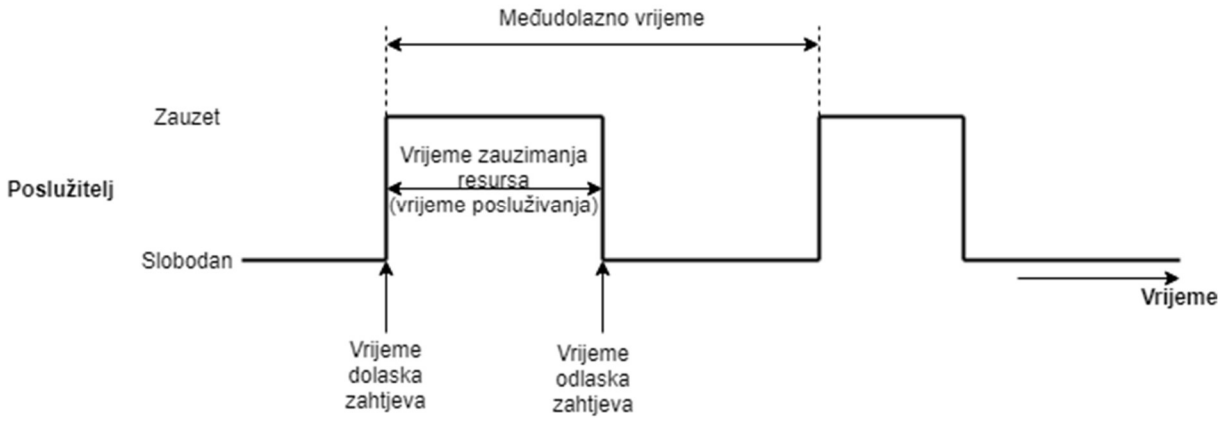
$$\overline{A}(t_1, t_2) = \frac{1}{t_2 - t_1} \cdot \int_{t_1}^{t_2} A(t)dt \quad (3)$$

gdje je:

- $\overline{A}(t_1, t_2)$ – prosječni intenzitet prometa u vremenskom razdoblju od t_1 do t_2,
- t_1 – početak promatranja,
- t_2 – kraj promatranja,
- $A(t)$ – intenzitet prometa u trenutku t.

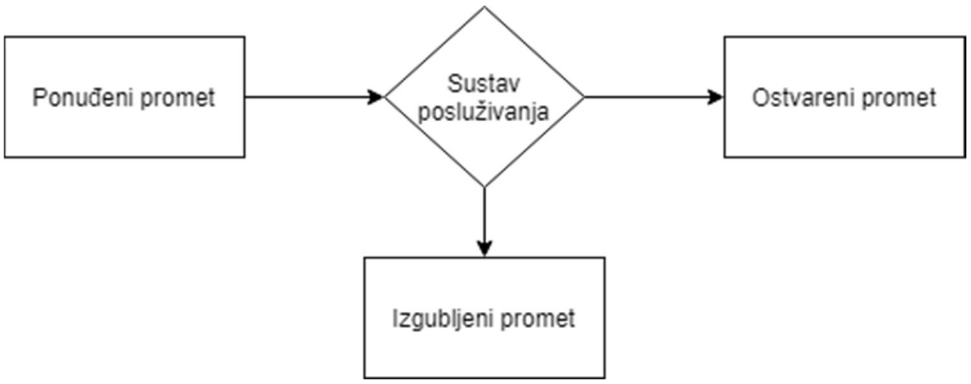
Prometni proces može se opisati pomoću karakterističnih veličina, prikazano na slici 1:

- uzorak dolazaka poziva,
- vrijeme zauzimanja resursa (vrijeme posluživanja),
- broj izvora,
- vrijeme između dolaska zahtjeva (među-dolazno vrijeme).



Slika 1: Karakteristične veličine prometnog procesa, [6]

Prema slici 2, kod proračuna promet se općenito može podijeliti na ponuđeni, ostvareni i izgubljeni promet. Ostvareni promet je promet koji je poslužen na telekomunikacijskoj opremi. Ponuđeni promet je količina zahtjeva koji nastoje pristupiti sustavu te se ne može izmjeriti već procijeniti na temelju ostvarenog prometa koji je izmjerljiv. Izgubljeni promet je promet koji zbog specifikacije sustava posluživanja i količine ponuđenog prometa nije mogao biti poslužen u danom trenutku – razlika između ostvarenog i ponuđenog prometa, [6].



Slika 2: Vrste prometa

2.2.Faktor blokiranja

Faktor blokiranja označava se sa p ili p_b te se definira kao vjerojatnost da se poziv ili zahtjev neće poslužiti. Prema [5, 7], drugi naziv za faktor ili vjerojatnost blokiranja je razina usluge (GoS), te se preko veličina prometa sa slike 2 može definirati kao:

$$p_b = \frac{izgubljeni\ promet}{ponuđeni\ promet} \tag{4}$$

Iz toga proizlazi da su ponuđeni, ostvareni i izgubljeni promet u međusobnom odnosu, pod pretpostavkom da nema ponovnog pokušaja zahtjeva za uslugom:

$$A_p = A_{ost} + A_{izg} \tag{5}$$

$$A_p = \frac{A_{ost}}{(1-p_b)} \tag{6}$$

gdje je:

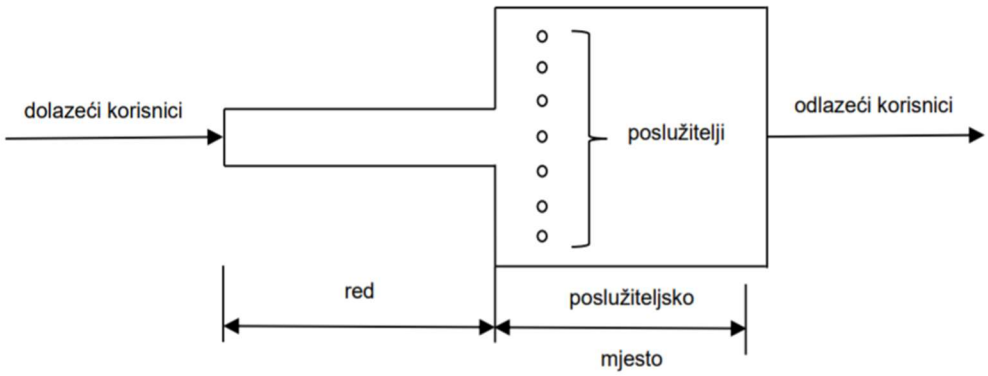
- A_p – ponuđeni promet,
- A_{ost} – ostvareni promet,
- A_{izg} – izgubljeni promet.

3. Primjena različitih podvorbenih modela za opis realnih sustava

Kako bi sustav posluživanja mogao u svakome trenutku poslužiti bilo koji broj korisnika, morao bi imati jednak ili veći broj poslužitelja od broja korisnika, što nije ekonomično, te u nekim slučajevima ni tehnički izvedivo. Pravilnim dimenzioniranjem sustava posluživanja pretpostavlja se i definira određena razina GoS, te se unaprijed zna koliko zahtjeva se može poslužiti, a koliko će ih se odbiti ili staviti u redove čekanja.

Zagušenje je stanje sustava u kojem nije moguće ostvariti vezu ulaza i izlaza zbog zauzetih poslužitelja ili komutacijskih čvorišta. S obzirom na način rukovanja zahtjevima ili pozivima u zagušenju, razlikuju se sustavi s gubicima i sustavi s čekanjem. Kod sustava s čekanjem, zahtjevi koji se ne mogu poslužiti se stavljaju u red na čekanje dok se ne oslobodi jedan od poslužitelja, dok sustavi s gubicima u slučaju zagušenja odbacuju zahtjeve koje zbog zauzetosti poslužitelja nije moguće poslužiti.

Model je teoretska razradba koja radi lakšeg razumijevanja omogućuje reprodukciju objekta koji se proučava – pojednostavljeni prikaz određenog stvarnog sustava, [8]. Podvorbeni modeli, prikazani na slici 3, opisuju sustave koji se sastoje od korisnika, reda čekanja, poslužitelja i pravila prema kojima se obavlja posluživanje korisnika, [9].

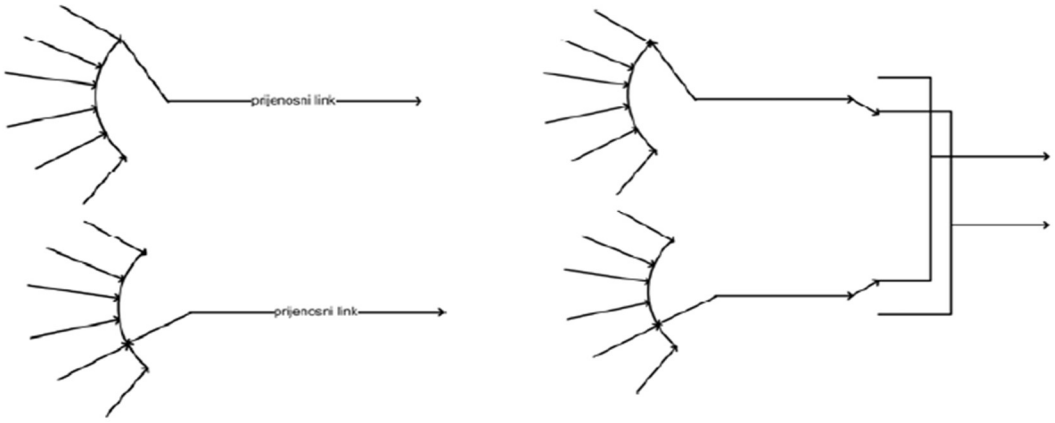


Slika 3: Shematski prikaz podvorbenog sustava, [9]

U podvorbenom sustavu, korisnici nailaze prema definiranoj distribuciji dolazaka u potrazi za uslugom koju pruža poslužiteljsko mjesto. Poslužiteljsko mjesto može imati jednog ili više poslužitelja te se pretpostavlja da jedan poslužitelj može posluživati jednog korisnika. Ako su svi poslužitelji zauzeti, korisnik se stavlja u red čekanja, gdje čeka na posluživanje ovisno o korištenoj disciplini posluživanja. Nakon posluživanja korisnik napušta sustav.

Teorija redova je disciplina koja se bavi proučavanjem i opisivanjem modela sustava s čekanjem na posluživanje, [10].

Sustav koji povezuje ulaz sa izlazom naziva se komutacijski sustav. Sustav u kojemu svaki ulaz može biti prospojen na svaki (slobodni) izlaz naziva se sustavom s potpunom dostupnošću, [5]. Potpuna dostupnost omogućava pristup svim korisnicima svim resursima, prikazano na slici 4.



Slika 4: Komutiranje s ograničenom dostupnošću (lijevo) i s potpunom dostupnošću (desno), [6]

Kako bi se mogli primijeniti prometni modeli na neki podvorbeni sustav, potrebno je odrediti elemente koji utječu na dolazak, prolazak i odlazak korisnika kroz podvorbeni sustav. U tu svrhu, podvorbeni sustav može se podijeliti na ulazni proces, strukturu sustava posluživanja i izlazni proces, [10].

3.1.Karakteristike ulaznog procesa podvorbenog sustava

Na ulazni proces podvorbenog sustava utječu sljedeće karakteristike:

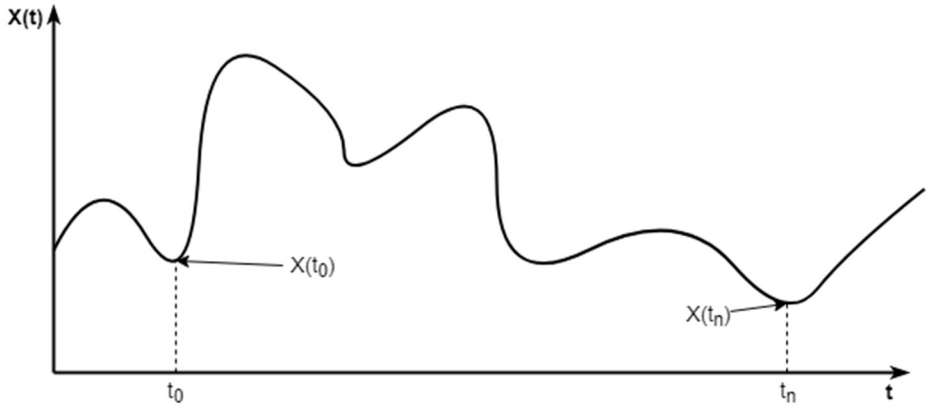
- veličina izvora – broj dolaznih korisnika,
- uzorak dolazaka zahtjeva,
- ponašanje dolaznih korisnika.

Veličina izvora može biti konačna i beskonačna, beskonačnim izvorom smatra se onaj izvor kod kojeg je broj dolaznih korisnika znatno veći od broja poslužitelja koji su u sustavu.

Uzorak dolazaka zahtjeva opisuje na koji način korisnici ulaze u podvorbeni sustav – nasumično ili se dolasci ravnaju po nekom pravilnom uzorku. Parametar koji se koristi za opisivanje uzoraka dolaska je međudolazno vrijeme između dva korisnika, prikazano na slici

1. Ukoliko korisnici ulaze na nasumičan način, potrebno je primijeniti određenu statističku distribuciju kako bi se mogao analizirati takav sustav.

Slučajni ili stohastički proces je funkcija koja dodjeljuje rezultat događaja vremenu u kojemu se događaj odvio: X(t), [11]. Slučajna varijabla je varijabla kojoj je vrijednost nepoznata već može biti neka vrijednost iz skupa očekivanih vrijednosti: $X(t_0 \ldots t_n)$, [12]. Na grafikonu 1 prikazan je stohastički proces kroz vrijeme, te slučajne varijable $X(t_0)$ i $X(t_n)$.



Grafikon 1: Stohastički proces X(t)

Najčešći slučaj, kod nasumičnog uzorka dolazaka, je da se za opisivanje uzoraka dolaska zahtjeva koristi Poissonov slučajni proces kod kojeg su međudolazna vremena eksponencijalno distribuirana, [10].

Poissonov proces je model koji opisuje niz diskretnih događaja, gdje je prosječno vrijeme između događaja poznato, ali nije poznat točan trenutak događaja. Nailazak novog događaja neovisan je o prijašnjim događajima, te nije moguć istovremeni dolazak više događaja, [13, 14].

U slučaju sustava posluživanja, događaj predstavlja nailazak korisnika, tj. dolazak zahtjeva za uslugom.

Distribucije i njihove oznake koje se često koriste za opisivanje međudolaznih vremena:

- M – Poissonov slučajni proces – međudolazna vremena su eksponencijalno distribuirana,
- D – deterministička distribucija – međudolazna vremena su konstantna,
- E – Erlangova distribucija,
- G – opća distribucija.

Ponašanje dolaznih korisnika odnosi se na ponašanje korisnika kod zagušenja sustava, kada su svi poslužitelji zauzeti, odnosno kada im se usluga ne može isporučiti. Korisnik u tome slučaju ima izbor odustati od traženog zahtjeva ili pokušati ponovno. Osnovni tipovi neisporuke usluge:

- blokiranje poziva uz zadržavanje (*Lost Calls Held* – LCH) – blokirani pozivi se ne vraćaju u sustav, korisnik nakon odbijene usluge odmah uspostavlja novi zahtjev za uslugom – uz pretpostavku da će mu sustav biti odmah dostupan čim postane raspoloživ,
- blokiranje poziva s odbijanjem (*Lost Calls Cleared* – LCC) – blokirani pozivi su izgubljeni, svaki pokušaj ponovnog poziva tretira se kao novi poziv,
- blokiranje s čekanjem (*Lost Calls Delayed* – LCD) – model koji se koristi u pozivnim centrima – blokirani pozivi ostaju u sustavu dok resursi ne postanu raspoloživi za posluživanje poziva,
- blokiranje s ponovnim pokušajima (*Lost Calls Retried* – LCR) – pretpostavlja se da određeni broj odbijenih poziva ponovno pokušava ostvariti poziv, izveden iz LCC modela, [6].

3.2. Karakteristike strukture sustava posluživanja

Pod strukturom podvorbenog sustava smatra se broj i topologija poslužitelja, te kapacitet sustava. Kapacitet sustava definira najveći broj korisnika koje podvorbeni sustav može poslužiti te uključuje korisnike u redu čekanja i one koji se poslužuju.

Topologija poslužitelja može biti paralelna i serijska – kod paralelnih poslužitelja korisnik nakon reda čekanja dolazi na bilo koji slobodni poslužitelj, dok kod serijskih korisnika poslužuje neki ili svi poslužitelji prije nego što izađe iz sustava. U ovome radu bavit će se isključivo sustavima s paralelnim rasporedom poslužitelja.

3.3. Karakteristike izlaznog procesa podvorbenog sustava

Disciplina posluživanja je karakteristika izlaznog procesa koja podrazumijeva pravila prema kojem se poslužuju zahtjevi te određuje redoslijed kojim korisnici bivaju posluženi nakon čekanja. Discipline posluživanja mogu biti:

- slučajna (*Serve in Random Order* – SIRO),
- prvi došao – prvi poslužen (*First Come First Served* – FCFS),
- posljednji došao – prvi poslužen (*Last Come First Served* – LCFS),
- disciplina posluživanja s prioritetom (*Priority Queue* – PQ),

- *processor sharing* (PS) i druge

Nije dovoljno znati samo koji će korisnik kada doći na red posluživanja, već i koliko će trajati posluživanje po korisniku. Različiti korisnici imat će različito vrijeme posluživanja, pa je za potrebe korištenja matematičkih modela potrebno odrediti distribuciju po kojoj se ravna vrijeme posluživanja. Za razdiobu vremena posluživanja koriste se distribucije već navedene za opisivanje među-dolaznih vremena.

3.4.Kendallova oznaka

Kendallova oznaka sastoji se od šest simbola te služi kao jednoznačni opis ranije navedenih karakteristika podvorbenog sustava, te se piše na sljedeći način:

$$A / B / m / K / n / D$$

gdje je:

- A : distribucijska funkcija među-dolaznih vremena,
- B: distribucijska funkcija vremena posluživanja,
- m : broj poslužitelja,
- K : kapacitet sustava (broj poslužitelja + broj mjesta u redu),
- n : veličina izvora (broj dolaznih korisnika),
- D: disciplina posluživanja, [15].

Ukoliko je disciplina posluživanja prvi došao – prvi poslužen te kapacitet i veličina izvora beskonačni, te oznake se ne pišu.

3.5.M/M/1

Sustav M/M/1 odnosno M/M/1/∞/∞/FCFS je osnovni tip sustava koji se analizira u teoriji teleprometa. Po Kendallovim oznakama, karakteristike ovog sustava su:

- M – među-dolazna vremena ravnaju se po eksponencijalnoj distribuciji,
- M – trajanje posluživanja ravna se po eksponencijalnoj distribuciji,
- 1 – jedan poslužitelj.

Disciplina posluživanja je prvi došao-prvi poslužen, a kapacitet sustava i veličina izvora su beskonačni.

U ovom sustavu, s obzirom da postoji samo jedan poslužitelj, prometno opterećenje ρ jednog poslužitelja jednako je ostvarenom prometu, iz relacije: [6]

$$\rho = \frac{A_{ost}}{m} \quad (7)$$

jer je m = 1.

3.6.M/M/m

M/M/m/∞/∞/FCFS je osnovni sustav posluživanja s čekanjem . Pri ulasku korisnika u sustav posluživanja, korisnik se poslužuje na prvom dostupnom poslužitelju. Ukoliko nema dostupnih poslužitelja, korisnik se stavlja u red čekanja dok se ne oslobodi jedan od poslužitelja, [13].

Karakteristike ovog sustava su:

- M – među-dolazna vremena ravnaju se po eksponencijalnoj distribuciji,
- M – trajanje posluživanja ravna se po eksponencijalnoj distribuciji,
- m poslužitelja,
- izvor i kapacitet sustava neograničene veličine,
- disciplina posluživanja prvi došao-prvi poslužen.

U ovom sustavu nema izgubljenog prometa, ostvareni promet je jednak ponuđenom, jer se ni jedan zahtjev ne odbacuje. Sustav raspolaže s redom čekanja neograničene veličine, te svi zahtjevi koji ne mogu biti posluženi odmah, čekaju određeno vrijeme na posluživanje.

3.7.M/M/m/m

M/M/m/m/∞/FCFS sustavi poznati su i kao Erlangovi sustavi s gubitcima jer, za razliku od prethodno objašnjenog M/M/m sustava, nema reda čekanja, već se promet, koji zbog zagušenja nije moguće poslužiti, odbacuje, [10].

Karakteristike ovog sustava su:

- M – među-dolazna vremena ravnaju se po eksponencijalnoj distribuciji,
- M – trajanje posluživanja ravna se po eksponencijalnoj distribuciji,
- m poslužitelja,
- m – kapacitet sustava (broj poslužitelja + broj mjesta u redu čekanja),
- ∞ – neograničen broj izvora,
- disciplina posluživanja prvi došao-prvi poslužen.

Iz karakteristika je vidljivo da je kapacitet sustava, koji se sastoji od broja poslužitelja i broja mjesta u redu čekanja, jednak broju poslužitelja, što znači da nema reda čekanja, već se

zahtjevi u stanju zagušenja odbacuju i tretiraju kao da nikad nisu ni ušli u sustav, a svi eventualni ponovni pokušaji se promatraju kao novi zahtjevi za uslugom.

4. Značajke i primjena Erlangove B formule gubitaka

Erlangova B formula koristila se za dimenzioniranje gotovo svih komutacijskih čvorišta telefonskih mreža u 20. stoljeću. Naziva se još i Erlangovom formulom gubitaka, jer u ovoj formuli gubitak predstavlja vjerojatnost blokiranja komutacijskog čvorišta u stanju zagušenja, [7, 13].

Erlangova B formula primjenjiva je na M/M/m/m sustavima, što znači da sustav mora ispunjavati sljedeće zahtjeve:

- neograničen broj dolaznih korisnika – izvora,
- među-dolazna vremena i vrijeme posluživanja ravnaju se po eksponencijalnoj distribuciji,
- blokirani pozivi se odbijaju bez zadržavanja (LCC) i zauvijek su izgubljeni,
- ograničen broj poslužitelja,
- bez mjesta za čekanje,
- sustav s potpunom dostupnošću.

Erlangova formula gubitaka, za parametre prosječnog ponuđenog prometa i broja poslužitelja, daje vjerojatnost da će ti poslužitelji, uz dani ponuđeni promet, biti zauzeti – što se naziva i razinom usluge (GoS).

$$p_B(A_p, m) = \frac{\frac{A_p^m}{m!}}{\sum_{i=0}^{m} \frac{A_p^i}{i!}} \tag{8}$$

Gdje je:

- p_B – vjerojatnost da su svi poslužitelji zauzeti,
- A_p – ponuđeni promet u Erlanzima,
- m – broj poslužitelja.

Erlangova B formula korisna je za dimenzioniranja sustava, tj. određivanje broja poslužitelja s poznatim ponuđenim prometom i razinom usluge koja se očekuje od sustava.

Ako se za primjer uzme sustav s očekivanim prometom od 45 Erlanga te željenim brojem poslužitelja 50, preko Erlangove formule gubitaka dobije se rezultat, 0.054, odnosno vjerojatnost da će svi poslužitelji biti zauzeti je 5.4%.

Kako bi se došlo do broja poslužitelja potrebnih za posluživanje određene količine prometa uz određenu razinu GoS, potrebno je, preko Erlangove B formule, izračunati vjerojatnost gubitaka za određeni broj poslužitelja, te usporediti rezultat s željenom razinom GoS.

Ukoliko se za isti promet, od 45 Erlanga, želi dimenzionirati sustav s vjerojatnošću blokiranja 2%, potrebno je ispitati broj poslužitelja potreban za zadovoljavanje zadanog GoS. Iz prethodnog primjera, za 50 poslužitelja vjerojatnost blokiranja iznosila je 0.054, a sada je tražena vjerojatnost 0.02, pa je potrebno uvećati broj poslužitelja.

Primjena Erlangove formule gubitaka (8) za zadani primjer, uz proizvoljnu vrijednost od 53 poslužitelja, daje sljedeći rezultat:

$$p_b(Ap = 45, m = 53) = \frac{\frac{45^{53}}{53!}}{\sum_{i=1}^{i=53}\frac{45^i}{i!}} = \frac{\frac{45^{53}}{53!}}{\frac{45^1}{1!}+\frac{45^2}{2!}+\frac{45^3}{3!}+\ldots+\frac{45^{53}}{53!}} = 0.0312$$

što ne zadovoljava GoS, te je potrebno uvećati broj poslužitelja i ponovno ispitati vjerojatnost blokiranja. Za navedeni promet od 45 Erlanga, GoS zadovoljava najmanje 56 poslužitelja, jer je tada vjerojatnost blokiranja jednaka 0.016, dok je za 55 poslužitelja jednaka 0.0203 što je iznad navedene razine usluge.

Iz navedenog primjera vidljiva je problematika korištenja Erlangove B formule, s obzirom da je ista namijenjena za računanje vjerojatnosti blokiranja, teško je iz nje izraziti i izračunati veličinu prometa ili broj potrebnih poslužitelja.

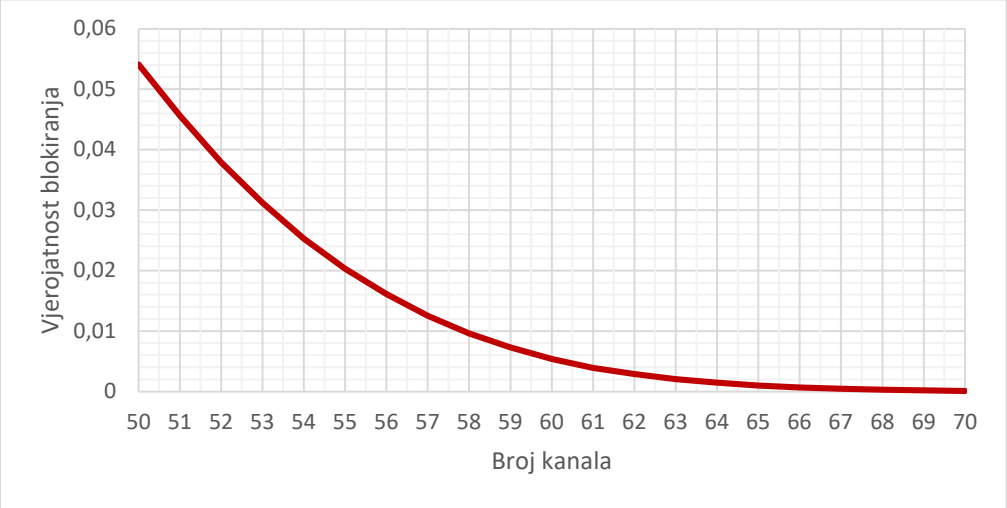
Rješenje ovog problema je korištenje tzv. Erlangovih tablica, iz kojih se lako iščitavaju tražene veličine prometa, broja poslužitelja ili vjerojatnosti blokiranja, prikazana u tablici 1.

Tablica 1: Erlang B tablica za promet od 45 – 50 Erlanga i broj poslužitelja od 50 - 60

m ↓	Ap → 45	46	47	48	49	50
50	0,0541	0,06334	0,07312	0,08334	0,09392	0,10479
51	0,04556	0,05404	0,06313	0,07273	0,08277	0,09316
52	0,03793	0,04563	0,05398	0,06291	0,07235	0,08221
53	0,0312	0,03809	0,04568	0,05391	0,0627	0,07198
54	0,02534	0,03143	0,03824	0,04573	0,05383	0,06248
55	0,02031	0,02561	0,03164	0,03837	0,04576	0,05375
56	0,01606	0,02061	0,02587	0,03184	0,0385	0,04579
57	0,01252	0,01636	0,02089	0,02612	0,03204	0,03862
58	0,00962	0,01281	0,01664	0,02116	0,02635	0,03222
59	0,00729	0,00989	0,01308	0,01692	0,02142	0,02658
60	0,00543	0,00752	0,01015	0,01336	0,01719	0,02167

Tablica se koristi na način da se za poznate dvije vrijednosti iščitava treća. Naprimjer, ukoliko je potrebno dimenzionirati sustav s ponuđenim prometom 50 Erlanga, i vjerojatnošću blokiranja manjom od 3%, iz tablice je vidljivo da je najmanji broj kanala koji zadovoljava navedene uvjete 59.

Za konstantnu veličinu prometa, s povećanjem broja poslužitelja vjerojatnost blokiranja težiti će prema nuli, odnosno za isti promet, dodavanjem poslužitelja povećava se vjerojatnost da će svi zahtjevi biti posluženi i da neće biti gubitaka, prikazano na grafikonu 2.



Grafikon 2: Ovisnost vjerojatnosti blokiranja o broju kanala za Ap = 45 Erl

5. Sustavi posluživanja s čekanjem i Erlangova C formula

Kod sustava posluživanja s čekanjem i neograničenom veličinom reda pretpostavlja se da nema izgubljenog prometa, svi zahtjevi će biti posluženi, [6].

$$A_p = A_{ost} = A \tag{9}$$

Erlangova C formula koristi iste parametre kao i Erlangova B formula gubitaka, ponuđeni promet i broj poslužitelja, uz dodatak rezultata Erlang B formule – vjerojatnost gubitaka. Njen rezultat je vjerojatnost čekanja π_m, koja se definira kao vjerojatnost da zahtjev neće biti poslužen odmah, nego da se stavlja na čekanje.

$$\pi_m(A,m) = \frac{p_B(A,m)}{1-\frac{A}{m}(1-p_B(A,m))} \tag{10}$$

Gdje je:

- π_m – vjerojatnost čekanja,
- A – ponuđeni promet,
- m – broj poslužitelja,
- p_b – vjerojatnost blokiranja,

te ako vrijedi da je ρ (A/m) u granicama [0,1>.

Erlangova C formula primjenjiva je na M/M/m sustavima, što znači da sustav mora imati sljedeće karakteristike:

- neograničen broj dolaznih korisnika – izvora,
- među-dolazna vremena i vrijeme posluživanja ravnaju se po eksponencijalnoj distribuciji,
- ograničen (konstantan) broj poslužitelja,
- neograničen broj mjesta za čekanje,
- sustav s potpunom dostupnošću.

Pretpostavka za korištenje Erlang C formule je da korisnik nikada ne odustaje od zahtjeva dok čeka u redu na posluživanje, [16].

Korisnost ove formule iskazana je u slučajevima gdje je razina usluge iskazana u vremenu čekanja na posluživanja, npr. čekanje u pozivnim centrima, posluživanje paketa na komutacijskim čvorištima paketnih mreža.

Ako se za primjer uzme sustav iz prethodnog poglavlja, s ponuđenim prometom 45 Erl, te 53 poslužitelja, za kojeg je vjerojatnost blokiranja iznosila 0.0312, vjerojatnost čekanja se može izraziti preko (10):

$$\pi_m(A = 45, m = 53) = \frac{0.0312}{1-\frac{45}{53}\cdot(1-0.0312)} = 0.17584,$$

iz čega proizlazi da je vjerojatnost da zahtjev bude stavljen na čekanje 17.58%.

Jedna od pretpostavki za korištenje Erlangove C formule je da je broj poslužitelja uvijek veći od ponuđenog prometa, jer u suprotnom formula daje nevaljani rezultat.

Neka je ponuđeni promet 50, te broj poslužitelja 50, u tom slučaju, prvo je potrebno izračunati p_b(Ap=50, m=50) po (8).

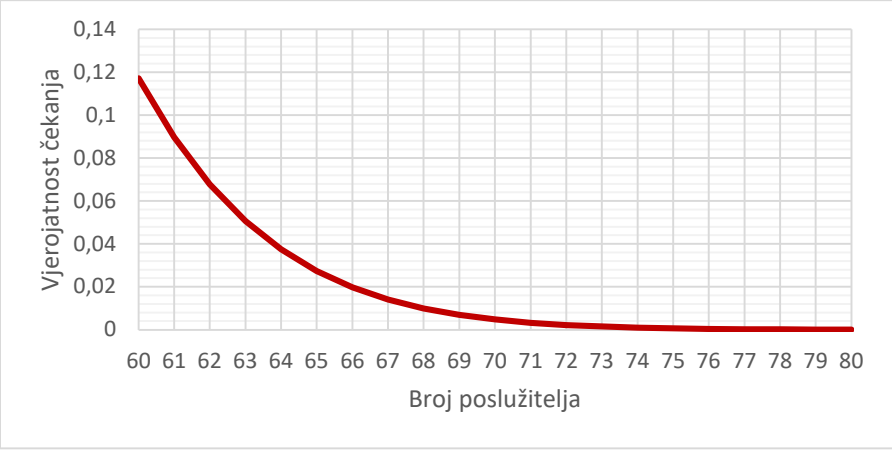
$$p_b(Ap = 50, m = 50) = \frac{\frac{50^{50}}{50!}}{\sum_{i=1}^{i=50}\frac{50^i}{i!}} = \frac{\frac{50^{50}}{50!}}{\frac{50^1}{1!}+\frac{50^2}{2!}+\frac{50^3}{3!}+\ldots+\frac{50^{50}}{50!}} = 0.10479,$$

te je onda vjerojatnost čekanja jednaka:

$$\pi_m(A = 50, m = 50) = \frac{0.10479}{1-\frac{50}{50}\cdot(1-0.10479)} = \frac{0.10479}{1-0.89521} = \frac{0.10479}{0.10479} = 1.$$

Za sve vrijednosti ponuđenog prometa veće od broja poslužitelja, vjerojatnost čekanja premašuje vrijednost 1, što nije moguć rezultat, što potvrđuje da ova formula nije pogodna za računanje kod takvih prometnih modela.

Vjerojatnost čekanja je u sličnoj ovisnosti o broju poslužitelja kao i vjerojatnost blokiranja, za istu veličinu prometa, povećanjem broja poslužitelja smanjuje se vjerojatnost da zahtjev bude stavljen na čekanje, prikazano na grafikonu 3.



Grafikon 3: Ovisnost vjerojatnosti čekanja o broju poslužitelja za Ap = 50 Erl

6. Aplikacija za izračun vjerojatnosti gubitaka i čekanja

Aplikacija za izračun Erlang B i Erlang C formule gubitaka i čekanja znatno olakšava izračun bilo kojeg od tri parametra formula, bez potrebe za korištenjem tablica.

Za izradu aplikacije odabran je jezik C# u *Windows Forms* (WinForms) grafičkom sučelju te *Visual Studio* razvojno okruženje, zbog dominantnosti *Windows* operativnog sustava u domeni osobnih računala te jednostavnosti izrade.

6.1. C# i WinForms

C# je moderni, objektno orijentirani programski jezik razvijen od *Microsoft*-a na .NET *frameworku*. Svoje korijene vuče iz C i C++ jezika, te ima sličnosti s Java i Javascript jezicima.

WinForms je komponenta .NET *framework*-a koja se bavi grafičkim sučeljem, odnosno kontrolama i događajima koji se pojavljuju kod interakcije s kontrolama.

U C#-u, program se sastoji od deklaracije imenskog prostora (*namespace)*, klase, atributa i metoda klase, glavne metode te skupa naredbi unutar metoda [17].

S obzirom na jednostavnost izrade aplikacije za izračun vjerojatnosti gubitaka i čekanja, uglavnom su se koristili osnovni programski elementi, petlje i grananja.

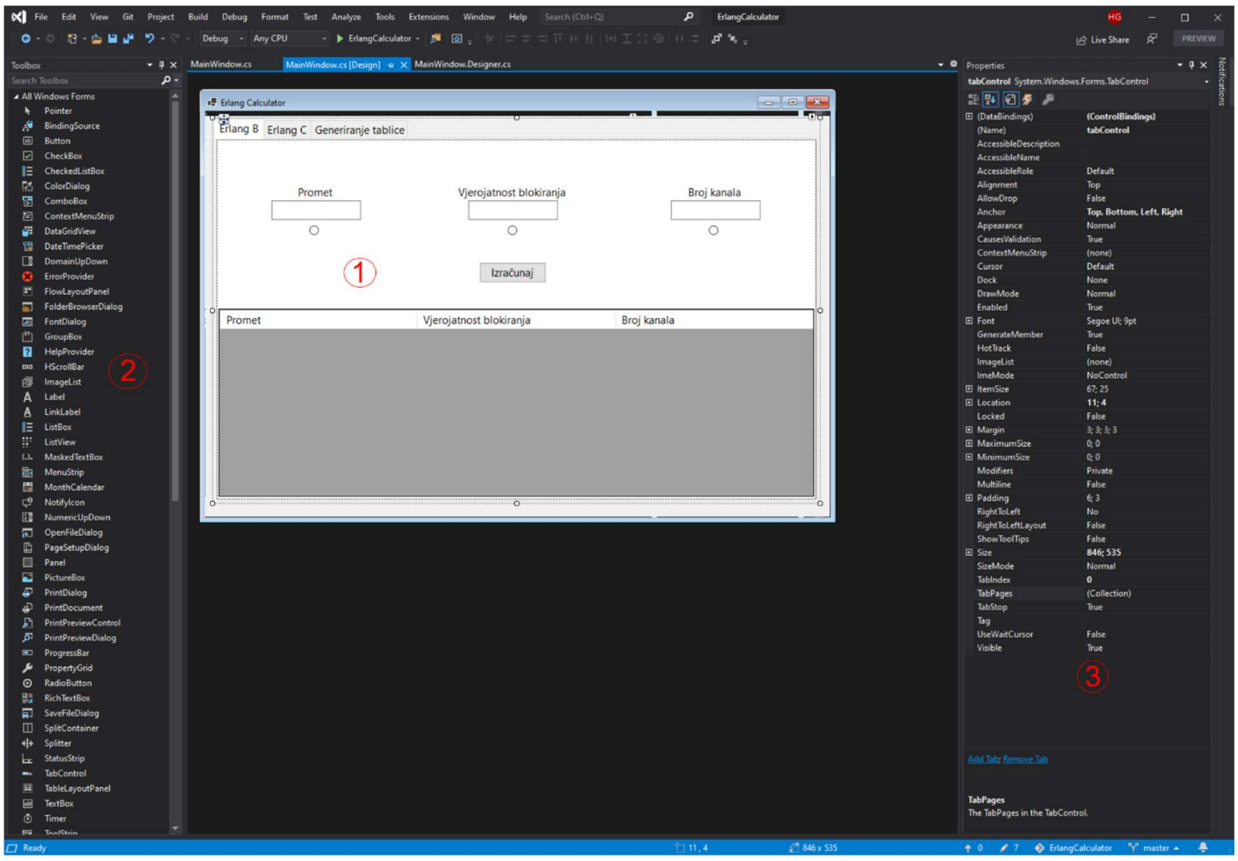
6.2. Visual Studio

Visual Studio je razvojno okruženje za izradu računalnih i mobilnih aplikacija, web stranica i servisa, razvijeno od strane *Microsoft*-a, s integriranom podrškom za C, C++, C#, *Visual Basic* .NET, F#, *JavaScript*, *TypeScript*, *Extensible Markup Language* (XML), *Extensible Stylesheet Language Transformations* (XSLT), *Hypertext Markup Language* (HTML), i *Cascading Style Sheets* (CSS). Implementirane su i razvojne platforme za *Microsoft*-ove proizvode, kao što su *Windows Application Programming Interface* (API), *Windows Forms*, *Windows Presentation Foundation*, *Windows Store* i *Microsoft Silverlight*, što znatno olakšava razvoj aplikacija za računala s *Windows* operativnim sustavom, [18].

Aplikacija, tj. projekt u ovom razvojnom okruženju sadržana je u tzv. *solution*-u, gdje se nalaze sve datoteke potrebne za funkcioniranje aplikacije. Izrađena aplikacija sastoji se od dvije datoteke, *MainWindow.cs* i *Program.cs* u kojima je sadržan programski kod potreban za funkcioniranje grafičkog sučelja, te napisani programski kod.

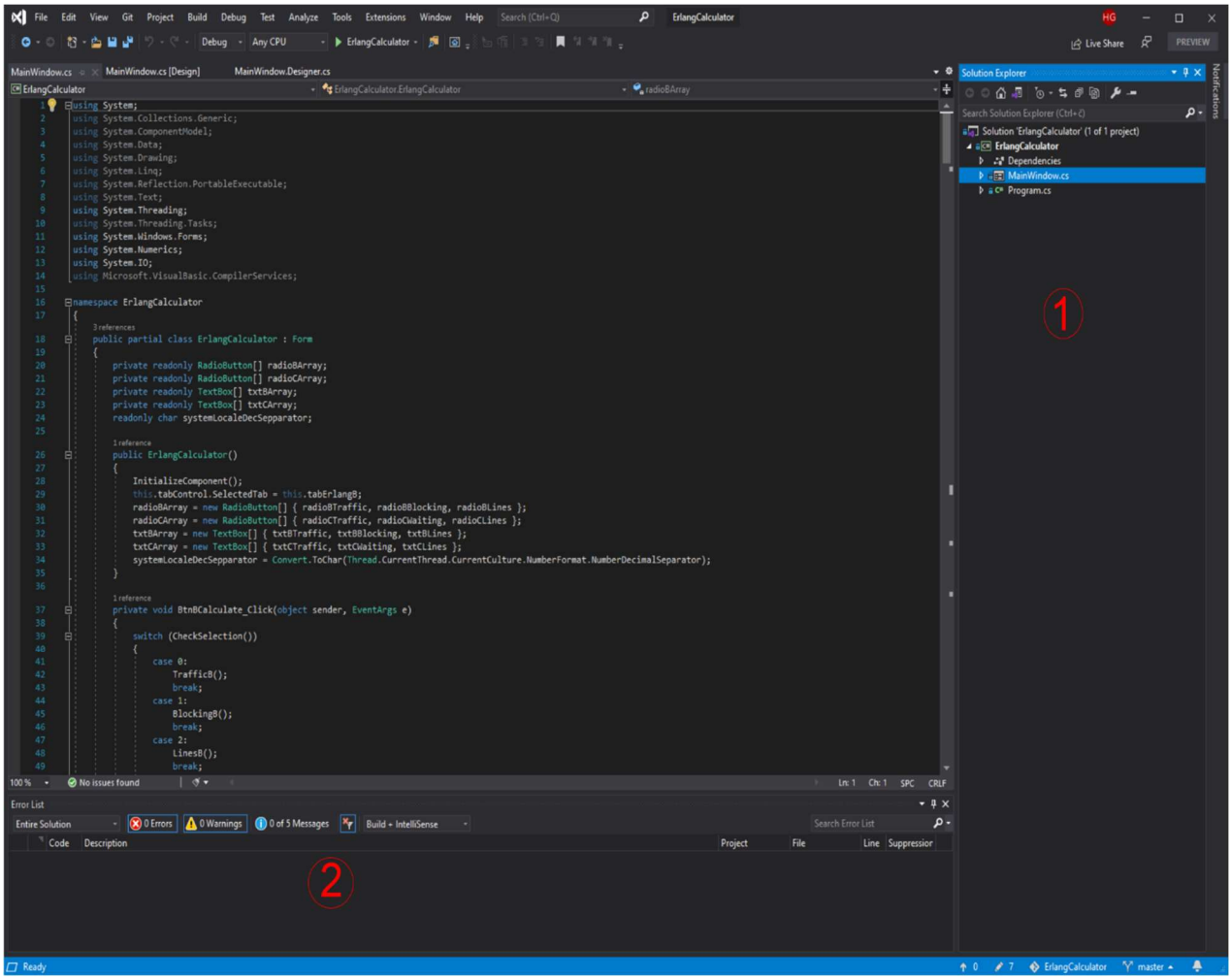
Na slici 5, prikazan je *Designer* dio razvojnog okruženja, koji se koristi za postavljanje kontrola grafičkog sučelja, tj. u kojem se postavljaju *WinForms* elementi vizualnog djela

aplikacije, koji se biraju iz *toolbox*-a, označenog brojem 2. Broj 1 prikazuje prototip grafičkog sučelja, te se na njega postavljaju kontrole iz *toolbox*-a. Svaka kontrola ima svoja svojstva, npr. ime, veličina, tekst, pozicija, itd., te tzv. događaje, kao što je naprimjer *Click* događaj koji se izvršava kad se na kontrolu pritisne lijevo dugme miša. Svojstva i događaji za označenu kontrolu postavljaju se u *Properties* dijelu, prikazanom brojem 3.



Slika 5: *Designer* dio razvojnog okruženja *Visual Studio*

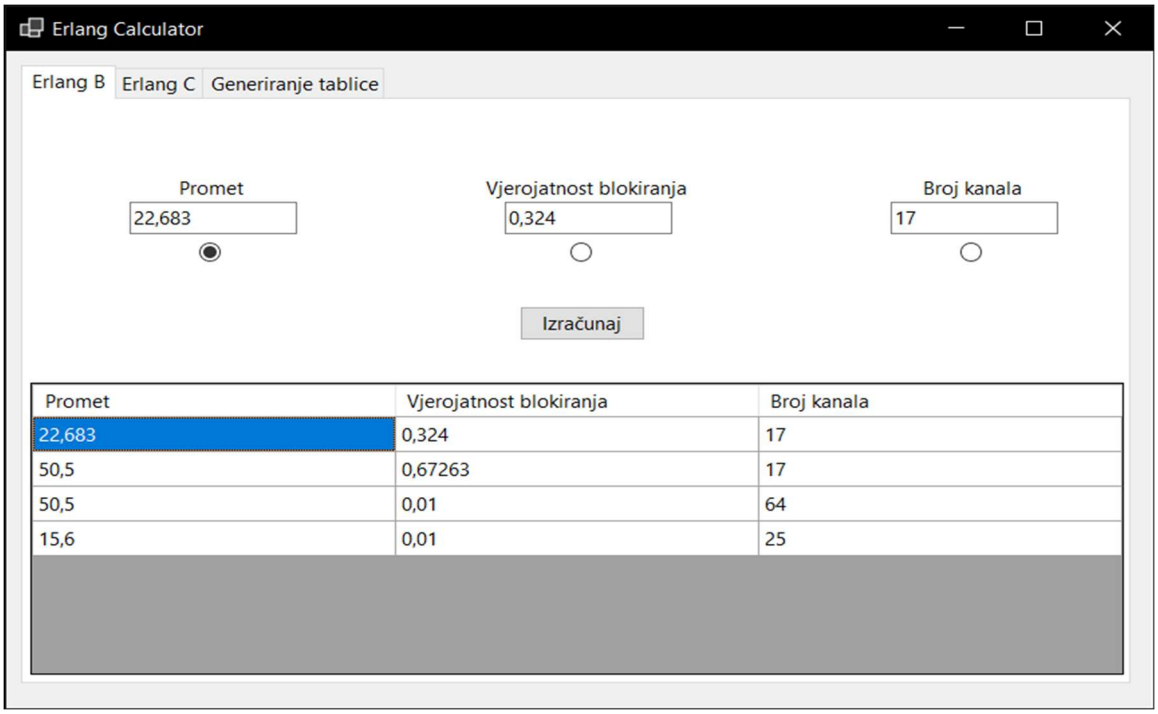
Izvan *Designer* dijela, Visual Studio bavi se programskim kodom sadržanim u datotekama. Na slici 6 prikazani su elementi tog pogleda. Sve datoteke vezane za projekt prikazane su u *Solution Explorer* dijelu, označen brojem 1. Bitan dio je i *Error list* prikazan brojem 2, u kojemu se prikazuje linija u kojoj je neispravno napisan kod, te objašnjenje greške i eventualni savjeti za rješavanje grešaka i upozorenja.



Slika 6: Dio razvojnog okruženja *Visual Studio* za pisanje i uređivanje programskog koda

6.3.Grafičko sučelje

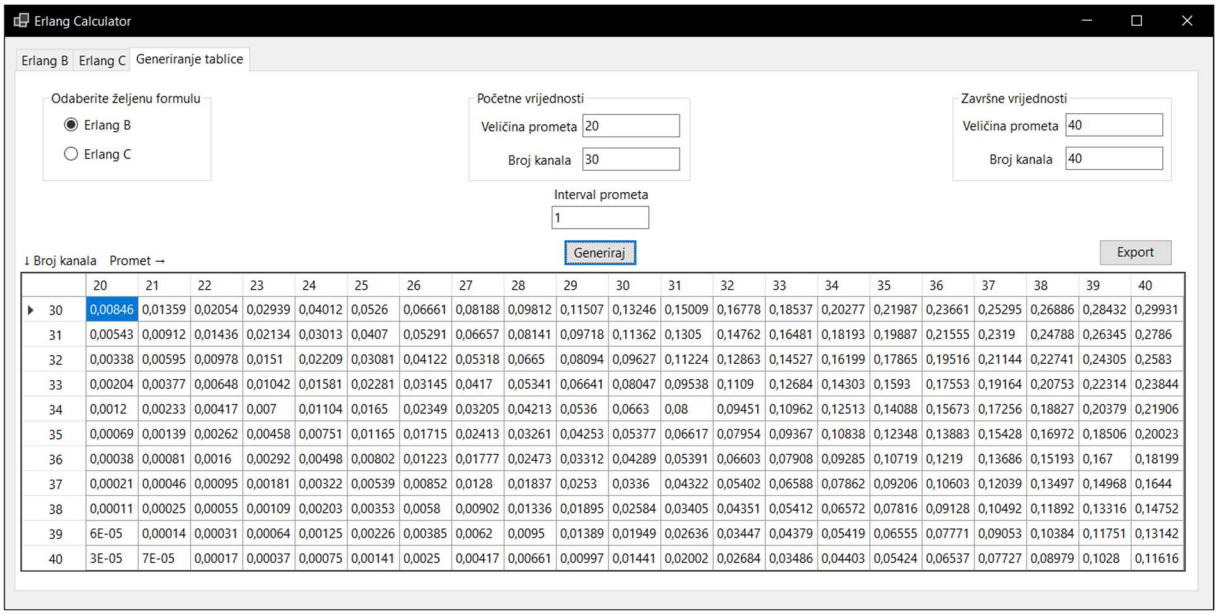
Grafičko sučelje aplikacije sastoji se od tri zasebna dijela, odvojena *tab control* elementom, Erlang B, Erlang C te dio namijenjen za generiranje tablica, prikazano na slici 7. Navigacija između dijelova izvedena je preko *tab*-ova, klikom na željenu sekciju prikazuje se samo taj dio aplikacije.



Slika 7: Erlang B sekcija aplikacije

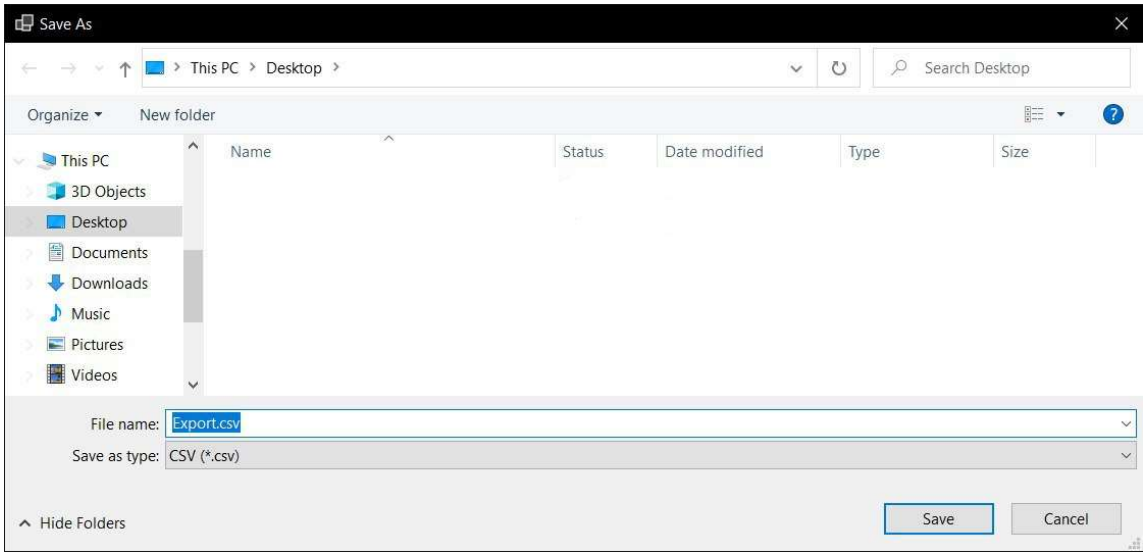
Erlang B i Erlang C sekcija sastoji se od istih elemenata, *text box* za upisivanje vrijednosti parametara, *radio button* za odabir željenog parametra za računanje, *button* za pokretanje postupka računanja te *data grid view* za prikaz prijašnjih izračunatih vrijednosti, vidljivo na slici 7.

Na slici 8 prikazana je sekcija za generiranje tablice, koja se sastoji od dva *radio button*-a za odabir željene formule za generiranje, te *textbox*-eva za unos početnih i završnih parametara veličine ponuđenog prometa i broja poslužitelja te *textbox*-a za unos intervala prometa, tj. za koliko će se početni promet uvećavati dok ne dođe do završnog prometa. *Button* „Generiraj“ računa i prikazuje tablicu za zadane parametre u *data grid view* na dnu. Nakon generiranja tablice, pojavljuje se i *button* „Export CSV“ pritiskom na koji se otvara dijalog za pohranu datoteke, prikazan na slici 9, gdje je moguće odabrati ime i mjesto na koje se pohranjuje generirana tablica u *Comma separated values* – CSV formatu, koja se može uvesti u *Excel* radi daljnje obrade ili ispisa.



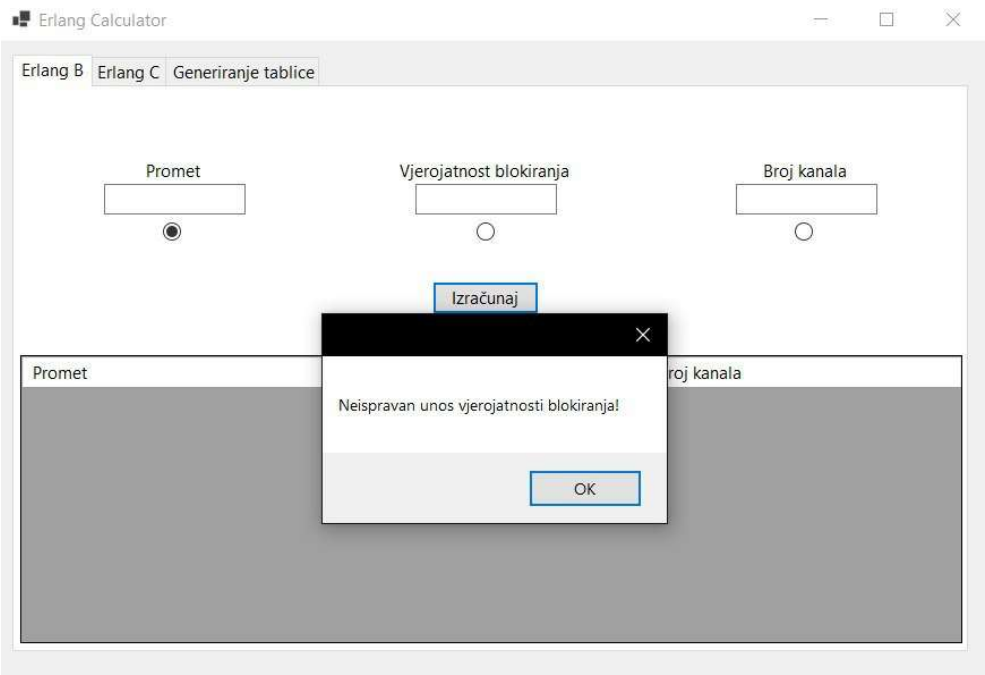
	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
30	0,00846	0,01359	0,02054	0,02939	0,04012	0,0526	0,06661	0,08188	0,09812	0,11507	0,13246	0,15009	0,16778	0,18537	0,20277	0,21987	0,23661	0,25295	0,26886	0,28432	0,29931
31	0,00543	0,00912	0,01436	0,02134	0,03013	0,0407	0,05291	0,06657	0,08141	0,09718	0,11362	0,1305	0,14762	0,16481	0,18193	0,19887	0,21555	0,2319	0,24788	0,26345	0,2786
32	0,00338	0,00595	0,00978	0,0151	0,02209	0,03081	0,04122	0,05318	0,0665	0,08094	0,09627	0,11224	0,12863	0,14527	0,16199	0,17865	0,19516	0,21144	0,22741	0,24305	0,2583
33	0,00204	0,00377	0,00648	0,01042	0,01581	0,02281	0,03145	0,0417	0,05341	0,06641	0,08047	0,09538	0,1109	0,12684	0,14303	0,1593	0,17553	0,19164	0,20753	0,22314	0,23844
34	0,0012	0,00233	0,00417	0,007	0,01104	0,0165	0,02349	0,03205	0,04213	0,0536	0,0663	0,08	0,09451	0,10962	0,12513	0,14088	0,15673	0,17256	0,18827	0,20379	0,21906
35	0,00069	0,00139	0,00262	0,00458	0,00751	0,01165	0,01715	0,02413	0,03261	0,04253	0,05377	0,06617	0,07954	0,09367	0,10838	0,12348	0,13883	0,15428	0,16972	0,18506	0,20023
36	0,00038	0,00081	0,0016	0,00292	0,00498	0,00802	0,01223	0,01777	0,02473	0,03312	0,04289	0,05391	0,06603	0,07908	0,09285	0,10719	0,1219	0,13686	0,15193	0,167	0,18199
37	0,00021	0,00046	0,00095	0,00181	0,00322	0,00539	0,00852	0,0128	0,01837	0,0253	0,0336	0,04322	0,05402	0,06588	0,07862	0,09206	0,10603	0,12039	0,13497	0,14968	0,1644
38	0,00011	0,00025	0,00055	0,00109	0,00203	0,00353	0,0058	0,00902	0,01336	0,01895	0,02584	0,03405	0,04351	0,05412	0,06572	0,07816	0,09128	0,10492	0,11892	0,13316	0,14752
39	6E-05	0,00014	0,00031	0,00064	0,00125	0,00226	0,00385	0,0062	0,0095	0,01389	0,01949	0,02636	0,03447	0,04379	0,05419	0,06555	0,07771	0,09053	0,10384	0,11751	0,13142
40	3E-05	7E-05	0,00017	0,00037	0,00075	0,00141	0,0025	0,00417	0,00661	0,00997	0,01441	0,02002	0,02684	0,03486	0,04403	0,05424	0,06537	0,07727	0,08979	0,1028	0,11616

Slika 8: Erlang B generirana tablica za veličinu prometa od 20 - 40 Erl i broj poslužitelja od 30 – 40 uz intenzitet prometa 1



Slika 9: Prozor za pohranjivanje datoteke

Kroz sve *textbox* elemente moguće je prolaziti preko *tab* tipke, te je moguće pokrenuti izračun pritiskom na tipku *Enter*, uz uvjet da su svi parametri popunjeni, u slučaju generiranja tablice, ili barem dva parametra popunjena u slučaju računanja vjerojatnosti gubitaka ili čekanja. Ukoliko neki parametar potreban za računanje nije popunjen, prikazuje se prikladna poruka korisniku preko kontrole *MessageBox*, prikazano na slici 10.



Slika 10: Skočni prozor za prikaz greške pri unosu

6.4.Programski kod

Pritiskom na tipku Izračunaj, u Erlang B sekciji prikazanoj na slici 7, poziva se metoda *BtnBCalculate_Click*, prikazana na slici 12, koja poziva metodu *CheckSelection*, prikazanu na slici 13, čija je uloga provjeriti koji parametar je potrebno izračunati, na način da se prvo provjerava koji *tab* je aktivan (Erlang B ili Erlang C), omogućuje se upis u *textbox* polja čije pripadajuće *radio button* kontrole nisu označene, te metoda vraća vrijednost 0 za izračun prometa, 1 za izračun vjerojatnosti, 2 za izračun broja poslužitelja tj. kanala, te -1 ukoliko niti jedan *radio button* nije označen. Za vrijednost -1, metoda *BtnBCalculate_Click* poziva metodu *AutoSelect*, prikazana na slici 11, koja će pokušati sama odabrati parametar za izračunavanje, na način da se bira parametar čiji *text box* je prazan, u slučaju da su dva *text box*-a popunjena.

```
private void AutoSelect()
{
    RadioButton[] radioBtnList;
    TextBox[] textBoxList;
    Button btn;
    if (this.tabControl.SelectedIndex == 0)
    {
        radioBtnList = radioBArray;
        textBoxList = txtBArray;
        btn = btnBCalculate;
    }
    else
    {
        radioBtnList = radioCArray;
        textBoxList = txtCArray;
        btn = btnCCalculate;
    }

    int emptyQty = 0;

    foreach (TextBox text in textBoxList)
    {
        if (string.IsNullOrWhiteSpace(text.Text)) { emptyQty++; }
    }

    if (emptyQty == 1)
    {
        for (int i = 0; i < textBoxList.Length; i++)
        {
            radioBtnList[i].Checked = string.IsNullOrWhiteSpace(textBoxList[i].Text);
        }
        CheckSelection();
        btn.PerformClick();
    }

}
```

Slika 11: Metoda *AutoSelect* za automatski odabir parametra za računanje

```
private void BtnBCalculate_Click(object sender, EventArgs e)
{
    switch (CheckSelection())
    {
        case 0:
            TrafficB();
            break;
        case 1:
            BlockingB();
            break;
        case 2:
            LinesB();
            break;
        default:
            AutoSelect();
            break;
    }
}
```

Slika 12: Metoda *BtnBCalculate_Click*

CheckSelection metoda se poziva i za svaki klik unutar *radio button* kontrole, kako bi se onemogućio upis vrijednosti za parametar koji se računa.

```
private int CheckSelection()
{
    RadioButton[] radioBtnList;
    TextBox[] textBoxList;
    if (this.tabControl.SelectedIndex == 0)
    {
        radioBtnList = radioBArray;
        textBoxList = txtBArray;
    }
    else
    {
        radioBtnList = radioCArray;
        textBoxList = txtCArray;
    }
    for (int i = 0; i < radioBtnList.Length; i++)
    {
        textBoxList[i].Enabled = !radioBtnList[i].Checked;
    }
    foreach (RadioButton radioBtn in radioBtnList)
    {
        if (radioBtn.Checked) { return Array.IndexOf(radioBtnList, radioBtn); }
    }
    return -1;
}
```

Slika 13: Metoda *CheckSelection*

Nakon odabira parametra kojeg je potrebno izračunati, pozivaju se odgovarajuće metode, *TrafficB*, *BlockingB* ili *LinesB*, koje za zadatak imaju pozvati odgovarajuće pomoćne metode za izračun parametra, te pohraniti rezultat u odgovarajući *text box* i u *data grid view* za pohranu prijašnji rezultata.

Za izračun prometa uz poznatu vjerojatnost blokiranja i broj poslužitelja, potrebno je uvećavati veličinu prometa i provjeravati je li vjerojatnost za tu veličinu prometa jednaka zadanoj vjerojatnosti, ukoliko je, ta veličina prometa je rezultat.

U aplikaciji je u metodi *FindTrafficB* prikazanoj na slici 14, isto implementirano na način da veličina prometa kreće od nule, te se uvećava za 1 sve dok iduće uvećanje za 1 ne prelazi zadanu vjerojatnost blokiranja, tada se uzima ta veličina prometa, i uvećava se za 0.0001 sve dok vjerojatnost ne dosegne zadanu vjerojatnost blokiranja. Metode *TestBlockingBLarge* i *TestBlockingB* pozvane iz metode *FindTrafficB*, uspoređuju vjerojatnost blokiranja izračunatu iz trenutne veličine prometa, sa zadanom vjerojatnošću, te vraćaju tip *bool*, istinu ili laž, ovisno o tome zadovoljava li trenutna veličina prometa zadanu vjerojatnost blokiranja, uz zadani broj poslužitelja. Promet se vraća u početnu metodu, gdje se zaokružuje na tri decimale i prikazuje u odgovarajućoj *textbox* kontroli.

```
private double FindTrafficB(double blocking, int lines)
{
    double traffic = 0;

    while (!TestBlockingBLarge(traffic, blocking, lines))
    {
        if (!TestBlockingBLarge(traffic, blocking, lines)) { traffic += 1; }
    }

    while (!TestBlockingB(traffic, blocking, lines))
    {
        if (!TestBlockingB(traffic, blocking, lines)) { traffic += 0.0001; }
    }

    return traffic;

}
```

Slika 14: Metoda *FindTrafficB* za izračun veličine prometa uz zadanu vjerojatnost blokiranja i broj poslužitelja

Izračun broja poslužitelja obavlja se na sličan način, provjerava se vjerojatnost blokiranja za jedan poslužitelj i zadanu veličinu prometa, te uspoređuje sa zadanom vjerojatnosti blokiranja, nakon čega se broj poslužitelja uvećava za jedan sve dok vjerojatnost blokiranja izračunata za taj broj poslužitelja ne bude manja ili jednaka zadanoj vjerojatnosti. Izračun je implementiran u metodi *CalculateLinesB* prikazanoj na slici 15.

```
private int CalculateLinesB()
{
    int lines = 0;

    if (!Double.TryParse(txtBTraffic.Text, out double traffic))
    {
        MessageBox.Show("Neispravan unos prometa!");
        return -1;
    }

    if (!Double.TryParse(txtBBlocking.Text, out double expectedBlocking))
    {
        MessageBox.Show("Neispravan unos vjerojatnosti blokiranja!");
        return -1;
    }

    expectedBlocking = Math.Round(expectedBlocking, 5);

    for (int i = 1; i < int.MaxValue; i++)
    {
        double calculatedBlocking = CalculateBlockingB(traffic, i, false);
        if (calculatedBlocking <= expectedBlocking)
        {
            lines = i;
            break;
        }
    }

    return lines;
}
```

Slika 15: Metoda *CalculateLinesB* za izračun broja poslužitelja uz zadanu vjerojatnost blokiranja i veličinu prometa

Kod izračuna vjerojatnosti blokiranja, u metodi *CalculateBlockingB,* prikazanoj na slici 16, nailazi se na problem kod pohrane međurezultata formule u varijable, zbog računanja

faktorijela broja poslužitelja. Tipovi podataka koji se koriste za pohranu u varijable, prikazani u tablici 2, nisu dovoljno veliki da pohrane faktorijele veće od 20, jer 20! iznosi 2,432,902,008,176,640,000, dok u najvećem tipu podatka u C# jeziku, *ulong,* može se najviše pohraniti 18,446,744,073,709,551,615, što bi značilo da Erlang B i C kalkulatori funkcioniraju samo do 20 kanala, jer je u formuli za izračun vjerojatnosti blokiranja, (8), u nazivniku faktorijela broja kanala.

Tablica 2: Tipovi podataka za pohranu brojeva u C#

Tip podatka u C#	Raspon	Vrsta vrijednosti
byte	0 - 255	cijeli broj
sbyte	-128 - 127	cijeli broj
short	-32,768 - 32,767	cijeli broj
ushort	0 - 65,535	cijeli broj
int	-2,147,483,648 - 2,147,483,648	cijeli broj
uint	0 - 4,294,967,295	cijeli broj
long	-9,223,372,036,854,775,808 - 9,223,372,036,854,775,807	cijeli broj
ulong	0 - 18,446,744,073,709,551,615	cijeli broj
float	-3.402823e38 - 3.402823e38	decimalni broj
double	-1.79769313486232e308 - 1.79769313486232e308	decimalni broj
decimal	(+/-)1.0 · 10e-28 - 7.9 · 10e28	decimalni broj
BigInteger	U teoriji beskonačan.	cijeli broj

Ovaj problem rješava se na način da se koristi *BigInteger* tip podatka iz *System.Numerics* imenskog prostora, koji, može pohraniti cijeli broj, u teoriji, bez ograničenja na veličinu. Problematika korištenja ovog tipa očitava se u matematičkim operacijama ostalih tipova podataka s ovim tipom, npr. kako bi se podijelio neki broj sa brojem pohranjenim u *BigInteger* tipu, i djeljenik i djelitelj moraju biti tipa *BigInteger*, gdje nastaje problem ukoliko

neki od njih nije cijeli broj, jer pretvorbom u *BigInteger* tip ostaje pohranjen samo cijeli dio broja.

```
9 references
private double CalculateBlockingB(double traffic = 0, int lines = 0, bool load = true)
{
    double blocking;

    if (load)
    {
        if (!Double.TryParse(txtBTraffic.Text, out traffic))
        {
            MessageBox.Show("Neispravan unos prometa!");
            return -1;
        }
        if (!Int32.TryParse(txtBLines.Text, out lines))
        {
            MessageBox.Show("Neispravan unos broja kanala!");
            return -1;
        }
    }


    double sum = ErlangSum(traffic, lines);
    BigInteger lines_Factorial = Factorial(lines);
    BigInteger trafflines = (BigInteger)(Math.Pow(traffic, lines) * 100000);

    blocking = Math.Round(((double)(BigInteger.Divide(trafflines, lines_Factorial)) / sum) / 100000, 5);
    return blocking;
}
```

Slika 16: Metoda *CalculateBlockingB* za izračun vjerojatnosti blokiranja

Za potrebe ovog rada, problem pretvorbe decimalnog broja u cijeli riješen je na način da se decimalni broj prije pretvorbe u cijeli množi sa 10^5 kako bi se sačuvalo pet decimalnih znamenki, te se rezultat dijeli s istim. Na taj način aplikacija je precizna u računanju vjerojatnosti do pet decimalnih mjesta. U radovima koje iziskuju veću preciznost, ovo nije adekvatno rješenje.

Za računanje Erlang C formule, logika za računanje svih parametara je ista, s razlikom formule korištene za računanje vjerojatnosti čekanja (10), u metodi *CalculateWaitingC*, prikazanoj na slici 17. Navedena metoda za dobivanje rezultata koristi vjerojatnost blokiranja, odnosno rezultat Erlang B formule, pa se iz iste poziva metoda *CalculateBlockingB* za zadane parametre veličine prometa i broja poslužitelja.

```
private double CalculateWaitingC(double traffic = 0, int lines = 0, bool load = true)
{
    double waiting;

    if (load)
    {
        if (!Double.TryParse(txtCTraffic.Text, out traffic))
        {
            MessageBox.Show("Neispravan unos prometa!");
            return -1;
        }
        if (!Int32.TryParse(txtCLines.Text, out lines))
        {
            MessageBox.Show("Neispravan unos broja kanala!");
            return -1;
        }
        if (traffic >= lines)
        {
            MessageBox.Show("Promet mora biti manji od broja kanala!");
            return -1;
        }
    }

    double pb = CalculateBlockingB(traffic, lines, false);

    waiting = Math.Round(pb / (1 - (traffic / lines) * (1 - pb)), 5);

    return waiting;
}
```

Slika 17: Metoda *CalculateWaitingC* za računanje vjerojatnosti čekanja (Erlang C formula)

Generiranje tablice implementirano je unutar metode *GenerateTable*, te funkcionira na način da se prvo prolazi kroz sve veličine prometa – početni promet se uvećava za zadani interval prometa sve dok se ne dosegne krajnja veličina prometa, kako bi se formirali stupci tablice. Nakon toga, za svaki broj poslužitelja, od početnog do krajnjeg, računa se vjerojatnost blokiranja ili čekanja, ovisno o odabranoj formuli, te se rezultati za taj broj poslužitelja pohranjuju u polje, iz kojeg se dalje pretvaraju u tekst i dodaju kao jedan red u tablici. Veličine prometa predstavljaju stupce, a brojevi poslužitelja predstavljaju redove. Dio koda metode za generiranje tablice koji se bavi formiranjem podataka i njihovim stavljanjem u tablicu, prikazan je na slici 18.

Kod generiranja ugrađen je zaštitni mehanizam da početne vrijednosti ne mogu biti veće od završnih, te da interval prometa mora striktno biti veći od 0.

Nakon što završi proces generiranja tablice, omogućuje se korištenje *button* kontrole „Export CSV“, pritiskom na koju se poziva metoda *SaveToCSV*, koja prikazuje dijalog za spremanje datoteke. Nakon odabira imena i mjesta spremanja datoteke, označuje se cijeli sadržaj tablice i pohranjuje u datoteku.

```
for (double i = startTraffic; i <= endTraffic; i += trafficInterval)
{
    i = Math.Round(i, decNum);
    columnCount++;
    dataGridGeneratedTable.Columns.Add(i.ToString(), i.ToString());
}

double[] result = new double[columnCount];
string[] resultString = new string[columnCount];
int column, row = 0;


for (int i = startLines; i <= endLines; i++)
{
    column = 0;
    for (double j = startTraffic; j <= endTraffic; j += trafficInterval)
    {
        if (radioGenErlangB.Checked)
        {
            result[column] = CalculateBlockingB(j, i, false);
        }
        else
        {
            result[column] = CalculateWaitingC(j, i, false);
        }
        column++;
    }

    for (int k = 0; k < result.Length; k++)
    {
        resultString[k] = result[k].ToString();
    }

    dataGridGeneratedTable.Rows.Add(resultString);
    dataGridGeneratedTable.Rows[row].HeaderCell.Value = i.ToString();
    row++;
}
btnExportCSV.Enabled = true;
```

Slika 18: Dio programskog koda metode *GenerateTable* za generiranje tablice

Sve metoda koje se bave računanjem imaju ugrađenu provjeru valjanosti unosa parametara, moguće je unositi samo brojke, te simbol za razdvajanje cijelog i decimalnog dijela broja. Ukoliko neki parametar potreban za računanje nije popunjen, prikazuje se prikladna poruka korisniku preko kontrole *MessageBox*, prikazano na slici 10.

7. Zaključak

Dimenzioniranje sustava posluživanja u cilju što bolje razine usluge nije lak posao, ali uz razumijevanje temeljnih prometnih veličina, kao što su intenzitet prometa i faktor blokiranja odnosno GoS, te statističkih pravila i modela, može se znatno razjasniti i optimizirati.

Sustavi posluživanja sastoje se od nadolazećih korisnika, reda čekanja, poslužiteljskog mjesta i izlaza za korisnike. Ovisno o karakteristikama svakog dijela, na sustave je moguće primijeniti poznate modele u svrhu predviđanja njihovih performansi pri promjenjivom prometnom opterećenju. Po načinu rada u zagušenju razlikuju se sustavi posluživanja s blokiranjem, tj. odbijanjem i sustave posluživanja s čekanjem. Koristeći Kendallove oznake moguće je jasno razlikovati sustave posluživanja ovisno o karakteristikama dijelova.

Erlangova B formula može se koristiti za računanje razine usluge, tj. vjerojatnosti blokiranja, za zadani intenzitet prometa i broj poslužitelja, što je korisno kod dimenzioniranja optimiziranog sustava, smanjenja troškova i povećanja učinkovitosti sustava pri velikom prometnom opterećenju. Primjenjiva je kod sustava za posluživanje s gubicima, M/M/m/m.

Erlangova C formula koristi se za računanje vjerojatnosti čekanja na posluživanje, te je primjenjiva u M/M/m sustavima, u kojima je broj poslužitelja veći od intenziteta ponuđenog prometa, jer u suprotnom ne daje valjane rezultate.

Problem s navedenim formulama je nemogućnost računanja ostalih parametara za zadanu vjerojatnost gubitaka ili čekanja.

Računanje ostalih parametara znatno olakšava izrađena aplikacija, koja za zadane vrijednosti bilo koja dva parametra, računa nepoznatu vrijednost. Aplikacija za računanje vjerojatnosti gubitaka i čekanja rješava taj problem, uz jasno postavljeno grafičko sučelje, te se s mogućnošću generiranja Erlang B i C tablica na jasan način prikazuju vrijednosti parametara, čime se ubrzava i pojašnjava sam proces dimenzioniranja sustava posluživanja.

U današnje vrijeme koje zahtjeva veliku razinu usluge i optimizaciju sustava, potrebno je koristiti nove tehnološke mogućnosti, uz teorijska znanja, kako bi se pojednostavili, ubrzali i poboljšali procesi koji dovode do ispravnog dimenzioniranja takvog sustava posluživanja, koji će zadovoljavati zahtijevani GoS bez obzira na predvidive promjene prometnog opterećenja.

LITERATURA

[1] Weik, M. H. : Communications Standard Dictionary, 1996.

[2] ITU-D : Teletraffic Engineering Handbook, 2005.

[3] ITU-T : E.600

[4] ITU-T : B.18

[5] Akimaru, H., Kawashima, K. : Telecommunication Networks and Computer Systems – Teletraffic Theory and Applications, 1999.

[6] Mrvelj, Š. : Predavanja iz kolegija Tehnologija telekomunikacijskog prometa 1, Fakultet prometnih znanosti, Zagreb, 2016.

[7] Freeman, R. L. : Telecommunication System Engineering, 2004.

[8] URL: https://www.rječnik.com/Model [pristupljeno: rujan 2020.]

[9] Herout, T. : Određivanje mjerila izvedbe poissonovskih i nepoissonovskih podvorbenih sustava, 2019.

[10] Chee-Hock Ng, Soong Boon-Hee : Queueing Modelling Fundamentals With Applications in Communication Networks, 2008.

[11] URL: https://web.sonoma.edu/users/f/farahman/sonoma/courses/ces540/lectures/Chapter6_Dig_Random_Proc.pdf, [pristupljeno: rujan 2020.]

[12] URL: https://www.investopedia.com/terms/r/random-variable.asp, [pristupljeno: rujan 2020.]

[13] Lakatos L., Szeidl, L., Telek, M. : Introduction to Queueing Systems with Telecommunication Applications, 2013.

[14] URL: https://towardsdatascience.com/the-poisson-distribution-and-poisson-process-explained-4e2cb17d459, [pristupljeno: rujan 2020.]

[15] Sztrik J. : Basic Queueing Theory, 2011.

[16] URL: https://en.wikipedia.org/wiki/Erlang_(unit)#Erlang_C_formula, [pristupljeno: rujan 2020.]

[17] URL: https://www.tutorialspoint.com/csharp/csharp_program_structure.htm, [pristupljeno: rujan 2020.]

[18] URL: https://visualstudio.microsoft.com/, [pristupljeno: rujan 2020.]

POPIS KRATICA

GoS – Grade of Service

Lost Calls Cleared – LCC

Lost Calls Delayed – LCD

Lost Calls Retried – LCR

Serve in Random Order – SIRO

First Come First Served – FCFS

Last Come First Served – LCFS

Priority Queue – PQ

Processor Sharing – PS

Windows Forms – WinForms

Extensible Markup Language – XML

Extensible Stylesheet Language Transformations – XSLT

Hypertext Markup Language – HTML

Cascading Style Sheets – CSS

Application Programming Interface – API

POPIS GRAFIKONA

POPIS SLIKA

POPIS TABLICA

Sveučilište u Zagrebu
Fakultet prometnih znanosti
10000 Zagreb
Vukelićeva 4

IZJAVA O AKADEMSKOJ ČESTITOSTI I SUGLASNOST

Izjavljujem i svojim potpisom potvrđujem kako je ovaj završni rad isključivo rezultat mog vlastitog rada koji se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu što pokazuju korištene bilješke i bibliografija.

Izjavljujem kako nijedan dio rada nije napisan na nedozvoljen način, niti je prepisan iz necitiranog rada, te nijedan dio rada ne krši bilo čija autorska prava.

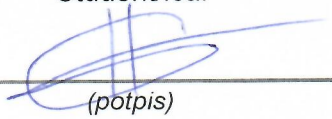
Izjavljujem također, kako nijedan dio rada nije iskorišten za bilo koji drugi rad u bilo kojoj drugoj visokoškolskoj, znanstvenoj ili obrazovnoj ustanovi.

Svojim potpisom potvrđujem i dajem suglasnost za javnu objavu završnog rada pod naslovom **ANALIZA VJEROJATNOSTI GUBITAKA I ČEKANJA U OVISNOSTI O PROMJENJIVOM PROMETNOM OPTEREĆENJU**

na internetskim stranicama i repozitoriju Fakulteta prometnih znanosti, Digitalnom akademskom repozitoriju (DAR) pri Nacionalnoj i sveučilišnoj knjižnici u Zagrebu.

U Zagrebu, 4.9.2020

Student/ica:



(potpis)