Analiza i prikaz podataka o korištenju pametnog telefona

Hilc, Borna

Undergraduate thesis / Završni rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Transport and Traffic Sciences / Sveučilište u Zagrebu, Fakultet prometnih znanosti**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:119:973409

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Faculty of Transport and Traffic Sciences - Institutional Repository

SVEUČILIŠTE U ZAGREBU

FAKULTET PROMETNIH ZNANOSTI

Borna Hilc

Analiza i prikaz podataka o korištenju pametnog telefona

ZAVRŠNI RAD

Zagreb, 2019

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI
ODBOR ZA ZAVRŠNI RAD

Zagreb, 8. travnja 2019.

Zavod: **Zavod za inteligentne transportne sustave**
Predmet: **Baze podataka**

ZAVRŠNI ZADATAK br. 5164

Pristupnik: **Borna Hilc (0135236885)**
Studij: Promet
Smjer: Informacijsko-komunikacijski promet

Zadatak: **Analiza i prikaz podataka o korištenju pametnog telefona**

Opis zadatka:

Podaci s pametnih telefona mogu poslužiti kao izvor velike količine podataka o aktivnostima korisnika. Dostupne otvorene podatke prikupljene tijekom projekta UbiqLog potrebno je pripremiti za obradu te oblikovati i izraditi bazu podataka u koju će se spremiti podaci. Potrebno je izraditi grafičko sučelje koje će se povezati s bazom podataka s ciljem vizualizacije i jednostavnije analize podataka. Analizom podataka nastojati će se pronaći određeni obrasci ponašanja korisnika te načini na koje korisnici koriste pametne telefone.

Mentor:

prof. dr. sc. Hrvoje Gold

Predsjednik povjerenstva za završni ispit:

SVEUČILIŠTE U ZAGREBU

FAKULTET PROMETNIH ZNANOSTI

ZAVRŠNI RAD

ANALIZA I PRIKAZ PODATAKA O KORIŠTENJU PAMETNOG TELEFONA

ANALYSIS AND VISUALIZATION OF SMARTPHONE USAGE DATA

Mentor: prof. dr. sc. Hrvoje Gold

Student: Borna Hilc

Neposredni voditelj/komentor:.

JMBAG: 0135236885

Tomislav Erdelić, mag. ing. el. techn. inf

Zagreb, 2019

Sažetak

Zadatak ovog završnog rada je implementirati grafičko sučelje pomoću kojega će se odabirom parametara grafički prikazivati podatci. Prije izrade grafičkoga sučelja prikupljene podatke je bilo potrebno pretvoriti u oblik prihvatljiv za uvoz u bazu podataka. Unutar izrađenoga grafičkoga sučelja moguće je izabrati više različitih parametara koji opisuju prikupljene podatke te napraviti filtriranje po istima. Grafičkim prikazom podataka uvelike se olakšava očitavanje i analiza podataka. Dobivene informacije mogu biti korištene za dobivanje navika i ponašanja korisnika te u mnoge druge svrhe.

KLJUČNE RIJEČI: baza podataka, SQL, analiza, C#

Summary

The purpose of this final paper is to implement a graphical interface for graphical display of data by selected parameters. Before creating the GUI, the collected data had to be converted into a format acceptable for import into the database. Within the created graphical interface, it is possible to select a number of different parameters that describe the collected data and to filter them by selected parameters. Graphic representation of the data makes it much easier to read the data. The information obtained can be used to obtain user habits and behavior, and for many other purposes.

KEYWORDS: database, SQL, analysis, C#

Sadržaj

1. Uvod 1
2. Metodologija prikupljanja podataka 2
3. Pohrana podataka u bazu podataka 5
3.1 Analiza aplikacije 7
3.2 Rukovanje pogreškama 9
3.3 Uvoz podataka u bazu podataka 11
4. Izrada grafičkog sučelja za prikaz podataka 15
5. Analiza podataka 24
6. Zaključak 41
Popis literature 42
Popis kratica 43
Popis slika 43
Popis priloga 44
Popis tablica 44

1. Uvod

Razvojem tehnologije mobilni uređaji su se pretvorili u dnevnike naših aktivnosti, prikupljajući podatke o našem ponašanju, navikama, kretanju, aktivnostima, a u novije vrijeme čak i biometrijske podatke. Vrlo bitna stavka prilikom korištenja podataka s pametnih telefona je to što su podaci prikupljeni tim načinom praktički jednaki normalnom ponašanju korisnika. Odnosno, nema promjena ponašanja korisnika te sam korisnik ne mora koristiti vanjski uređaj za prikupljanje podataka, već svoj svakodnevni uređaj. Time se vjerodostojnost podataka znatno povećava. Cilj ovoga rada je vidjeti načine, navike te svrhu korištenja pametnih telefona. Rad je podijeljen u šest cjelina:

1. Uvod
2. Metodologija prikupljanja podataka
3. Pohrana podataka u bazu podataka
4. Izrada grafičkog sučelja za prikaz podataka
5. Analiza podataka
6. Zaključak

U drugom poglavlju opisan je način prikupljanja podataka te tehnički uvjeti u kojima su prikupljani podatci.

U trećem poglavlju opisano je kreiranje dijagrama entiteta kao i ER dijagrama, izvlačenje podataka iz JSON datoteka te prilagodba za njihov unos u bazu podataka.

U četvrtome poglavlju prezentiran je način izrade grafičkoga sučelja tako da se na određenim dijelovima programa liniju po liniju objašnjavaju funkcije određenoga dijela kôda.

Peto poglavlje sadrži grafički prikaz podataka koristeći grafikone te analizu podataka.

2. Metodologija prikupljanja podataka

U današnjem svijetu gdje smo okruženi uređajima koji posjeduju veliku količinu senzora u sebi, imamo mogućnost praćenja ljudskog ponašanja te stvaranja određenih obrazaca ponašanja. Najpovoljniji uređaj za takvu analizu je pametan telefon zato što posjeduje veliku količinu senzora te se nalazi uz nas većinu vremena. Ova studija dizajnirana je kako bi se prikupio značajan broj podataka o korisnicima što bliže stvarnim uvjetima ponašanja. Osobe koje su sudjelovale u studiji bile su dobrovoljci, te nije postojao nikakav oblik nagrađivanja sudionika [1]. Uz to važno je naglasiti da su se sudionici koristili vlastitim pametnim telefonima, te su na njima bilježeni podatci. Koristeći vlastite pametne telefone tijekom studije, korisnici su mogli utjecati na konfiguraciju prikupljanja podataka te svojevoljno paliti/gasiti neke od senzora, time simulirajući uvijete iz stvarnog okruženja [1]. Dok pametni telefoni imaju mogućnost prikupljanja podataka postoji i mogućnosti gubitka podataka te element nesigurnosti koji proizlazi iz mogućnosti ručne konfiguracije senzora (npr. isključivanje *WiFi*-a kako bi se produžilo trajanje baterije) ili pak kvalitete samoga senzora (npr. geografske koordinate iščitane iz ID-a poziva) [1]. Za prikupljanje podataka koristila se aplikacija otvorenoga kôda imena *UbiqLog* [2]. U sučelju aplikacije moguće je pristupiti listi dostupnih senzora te isključiti ili uključiti određeni senzor, te promijeniti zadane intervale uzorkovanja [1]. Lista senzora te njihove zadane vrijednosti nalaze se u tablici 1.

Senzor	**Vrijednosti konfiguracije**
Aplikacija	Omogući = da, Interval zapisa u ms = 10000
Telefon	Omogući = da, Snimi komunikaciju = ne
SMS	Omogući = da
Lokacija	Omogući = da, Interval skeniranja u ms = 10000
Žiroskop	Omogući = ne, Prag tolerancije sile = 900
Temperatura	Omogući = ne, Mjerna jedinica = Celzijus
Kompas	Omogući = ne
Bluetooth	Omogući = ne, Interval skeniranja u ms = 60000
Orijentacija	Omogući = ne

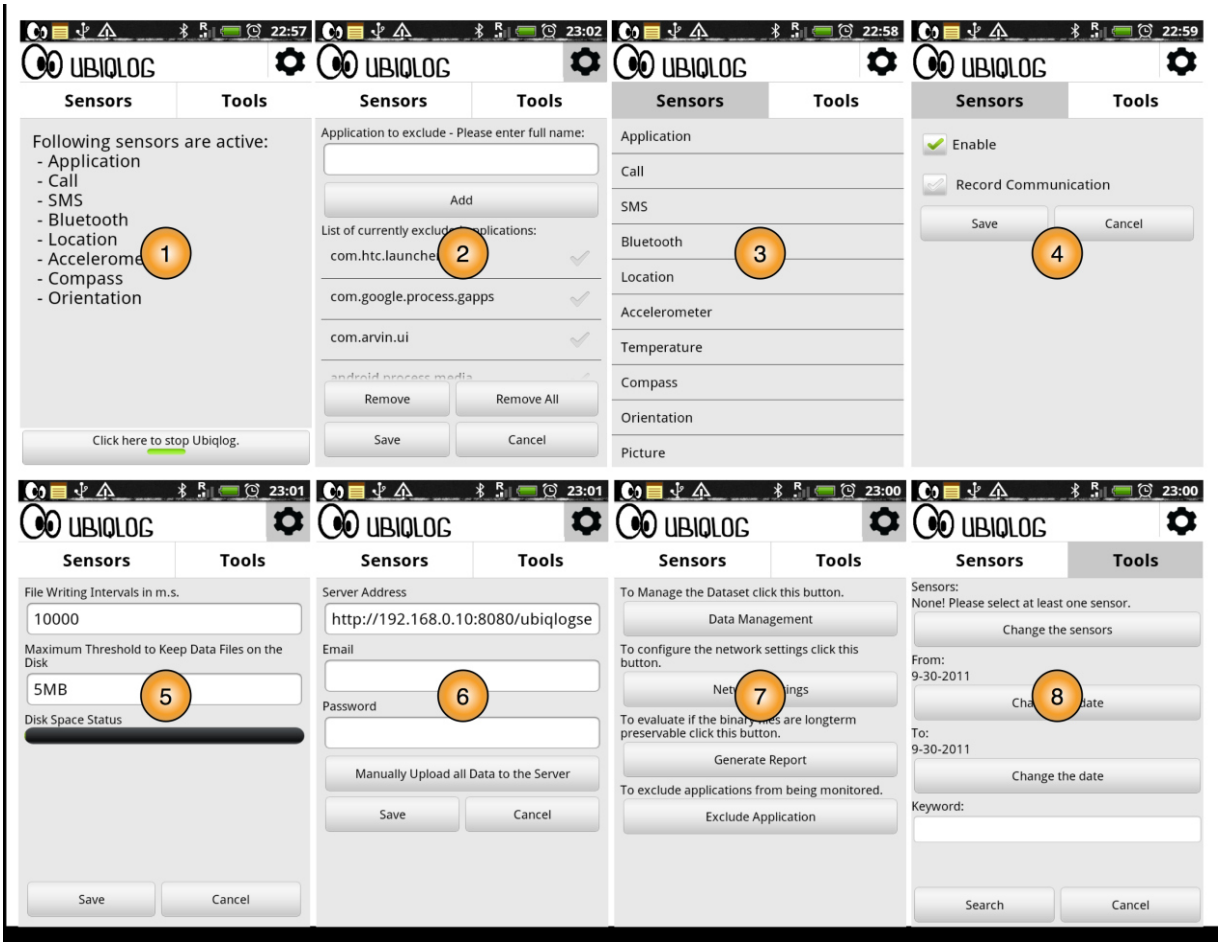
Tablica 1. Tablica zadanih vrijednosti senzora **[2]**

U studiji je prikupljeno 9.8 milijuna kontekstualnih informacija o korisnicima [1]. Neobrađeni podaci iz različitih senzora trebaju biti u skladu s općim formatom podataka, kako bi se olakšalo pretraživanje te daljnji dohvat podataka. Neobrađeni podaci se pretvaraju u konzistentnu strukturu čitljivu za *framework*. Za razliku od većine ostalih pristupa gdje se koristi XML format za pohranu podataka, kod implementacije ovog *frameworka* podaci se konvertiraju u JSON format, zbog toga što JSON dokumenti zauzimaju manje prostora na disku uz istu fleksibilnost kao XML dokumenti. Primjer obrađenih podataka SMS te aplikacijskog senzora moguće je vidjeti u nastavku [2].


```
{"Application":{"ProcessName":"com.facebook.orca","Start":"11-1-2014
14:40:56","End":"11-1-2014 14:40:56"}}

{"SMS":{"Address":"+98912600####","type":"2","date":"12-15-2013
10:41:33","body":"ANONYMIZED","Type":"2"}}
```


UbiqLog razvijen je za Android 2.0, 2.1 i 2.2 platforme. Aplikacija je testirana na *Motorola Milestone(Droid), HTC Legend, Samsung Galaxy S, HTC Desire i HTC Desire HD* mobilnim uređajima. Glavna misao vodilja tijekom izrade aplikacije bila je da grafičko sučelje bude jednostavno i lako dostupno korisnicima bez dubljeg znanja o funkcioniranju arhitekture uređaja [2]. Ulaskom u aplikaciju vidljiva je lista aktivnih senzora (Slika 1). U aplikaciji postoji opcija izuzimanja određene aplikacije iz *loggiranja*(2). Odabirom određenog senzora (3) moguć je ulazak u njegove postavke i konfiguracija (4). Moguće je još vidjeti postavke upravljanja podacima (5), konfiguriranje mrežnih postavki (6) te tražilice (7) [2].

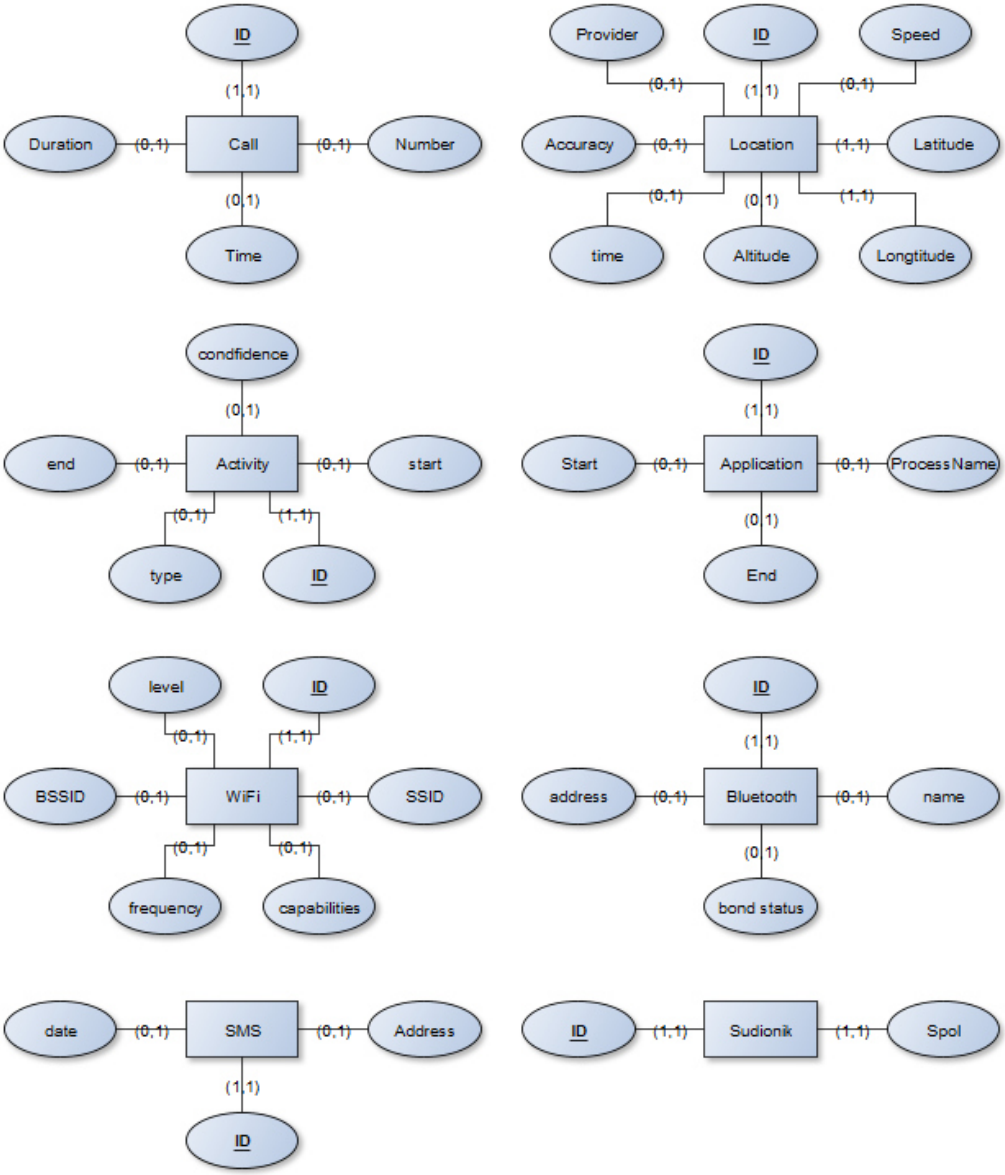


Slika 1. Prikaz grafičkog sučelja *UbiqLog* aplikacije [2]

Studija je provedena nad 35 sudionika u dobi između 19-32 godine, srednje dobi 22.5 godina uz standardno odstupanje od 5.6 godina. U studiji je sudjelovalo 25 žena i 10 muškaraca, te su svi sudionici bili studenti koji su dobrovoljno sudjelovali u studiji. Glavna mana ovakvoga načina provođenja studije bez nagrada bila je odustajanje velikoga broja sudionika. Studija je započeta s 54 sudionika, a završila je s 35 sudionika što znači da je 19 sudionika odustalo tijekom studije. Tijekom prikupljanja volontera za početak studije istima su objašnjene implikacije vezane uz privatnost, te su zamoljeni da daju pristanak za korištenje svojih osobnih podataka [1]. Valja naglasiti da su se uz već ranije navedene podatke prikupljali podaci o raspoloženju te spavanju sudionika. Ti podaci neće biti korišteni u ovome radu zato što su nepotpuni, te je samo dvoje sudionika pružilo potpune podatke, uz još dvoje koji su pružili podatke za manje od deset dana [1]. Sama studija je trajala 60 dana.

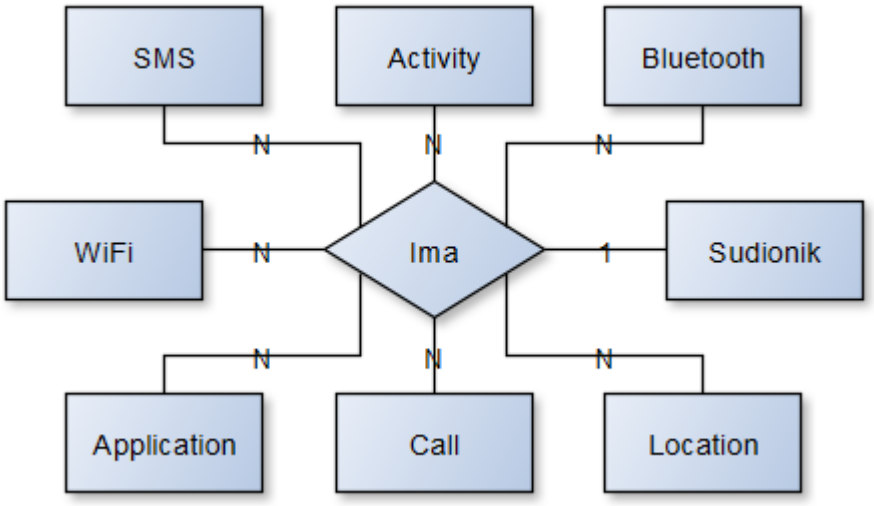
3. Pohrana podataka u bazu podataka

Kako bi se olakšao uvid u odnose podataka u aplikaciji i strukturirao način pohrane podataka u bazu podataka potrebno je odrediti entitete aplikacije i njihove atribute te odnose između entiteta, koji se prikazuju ER dijagramom. Dijagram entiteta (Slika 2) te ER dijagram (Slika 3) s pripadajućim kardinalitetima po kojima će biti kreirane tablice koje prikazuju Slika 2 i Slika 3.



Slika 2. Dijagram entiteta

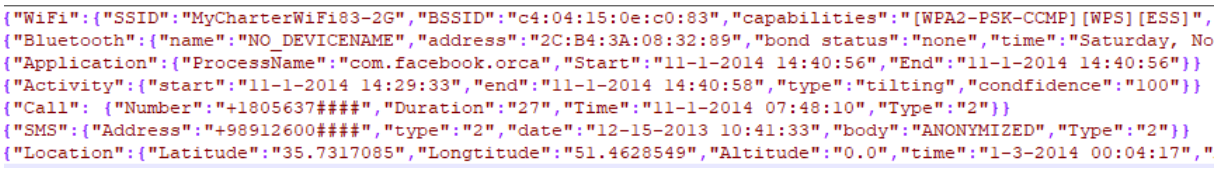
Entitet je skup objekata iz realnog svijeta koji imaju naglašena zajednička svojstva [3]. Ovdje je većina entiteta označava senzore koji su prikupljali podatke, izuzev entiteta „Sudionik“ koji obilježava osobe koje su sudjelovale u studiji. Može se pogledati entitet WiFi kao primjer, koji je opisan određenim atributima koji opisuju njegova svojstva odnosno obilježja [3]. Odnosno to su podatci koje je senzor prikupljao te identifikacijski atribut (ID), s jedinstvenom vrijednošću. U slučaju entiteta „Sudionik“ atributi su samo „Spol“ i „ID“. Pri utvrđivanju kardinaliteta određuju se donja i gornja granica [3]. Vrijednost nula (0) za donju granicu označava da se prilikom unosa ili ažuriranja nove instance entiteta vrijednost atributa može, ali i ne mora unijeti, dok vrijednost jedan (1) označava da se vrijednost atributa mora unijeti. Vrijednost jedan (1) za gornju granicu označava da se prilikom unosa ili ažuriranja nove instance entiteta, unijeti najviše jedna vrijednost za razmatrani atribut. Gledajući kardinalitete atributa entiteta WiFi, vidljivo je jedini obavezni atribut „ID“ (card(ID,WiFi)=(1,1)), dok su ostali opcionalni (primjerice card(SSID ,WiFi)=(0,1)).



Slika 3. ER dijagram

ER dijagramom se određuje u kakvoj su međusobnoj vezi entiteti. Budući da entitet “Sudionik“ može imati više svih ostalih entiteta on ja sa svima njima u vezi 1:N. Dok svaki od preostalih entiteta može pripadati samo jednome ispitaniku. Primjerice, jedan „Sudionik“ može imati više (N) pridruženih SMS-ova, dok jedan SMS pripada točno jednom ispitaniku.

Podaci prikupljeni tijekom izrade ovog rada nalaze su u JSON obliku (Slika 4), te ih je bilo potrebno pretvoriti u oblik pogodan za uvoz u bazu podataka. Za to je korištena aplikacija za deserijalizaciju JSON datoteka napisana u C# programskom jeziku.



Slika 4. Oblik izvornih datoteka

3.1 Analiza aplikacije

Za deserijalizaciju JSON datoteka korišten je Json.NET, framework visokih performansi za rad u .NET okruženju [4] . Za njegovo korištenje bilo je potrebno preuzeti „Newtonsoft.Json“ dodatak te dodati biblioteku koristeći naredbu „`using Newtonsoft.Json;`“

```
1. int id = 35;
2. int counter = 1;
3. string line;
4. System.IO.StreamReader file = new
   System.IO.StreamReader(@"path" + id + ".txt");
5. StreamWriter SWifi = new StreamWriter(@"path" + id +
   ".txt");
6. StreamWriter SBlue = new StreamWriter(@"path" + id +
   ".txt");
7. StreamWriter SApp = new StreamWriter(@"path" + id +
   ".txt");
8. StreamWriter SAct = new StreamWriter(@"path" + id +
   ".txt");
9. StreamWriter SCall = new StreamWriter(@"path" + id +
   ".txt");
10. StreamWriter SSMS = new StreamWriter(@"path" + id +
   ".txt");
11. StreamWriter SLoc = new StreamWriter(@"path" + id +
   ".txt");
12.
```

Prilog 1. Deklariranja varijabli za učitavanje datoteka

U Prilogu 1. vidljiv je dio kôda koji prikazuje deklariranje varijable „id“ koja označava jednoga od 35 sudionika studije čiji su pripadajući podaci prikupljeni u tekstualne datoteke oblika 1.txt, 2.txt, 3.txt, pri čemu naziv datoteke označava redni broj sudionika. „StreamReader“ instanca omogućava čitanje tekstualne datoteke, dok „id“ varijabla označava koja će se datoteka čitati. „StreamWriter“ klasa omogućava zapis podatka u datoteku koja se također mijenja ovisno o vrijednosti „id“ varijable. Postoji 7 instanci „StreamWriter“ klase zato što je tijekom studije

prikupljano 7 različitih vrsta podataka, te ih potrebno zapisati u 7 različitih datoteka kako bi se pojednostavio proces uvoza podataka u bazu podataka (jedna datoteka=jedna tablica). U Prilogu 2. prikazana je *while* petlja koja uzima liniju iz učitane datoteke te postavlja varijablu „line“ na njenu vrijednost, datoteka se čita sve dok postoji linija s tekstom odnosno dok vrijednosti linije nije „null“ tj. nema vrijednosti. Valja naglasiti da je potrebno čitati liniju po liniju datoteke zbog lošega formata izvorne JSON datoteke gdje nije vidljiv kraj svake linije ako se cijela datoteka pročita odjednom. Dalje slijedi „if“ naredba koja provjerava niz znakova kojima počinje učitana varijabla „line“, te ako je izjava istinita izvršava deserijalizaciju navedene linije te se ona sprema u svoju kreiranu klasu (Prilog 3). Svaka klasa sadrži atribute pripadajućeg entiteta. Nakon toga određeni „StreamWriter“ ispisuje željene objekte klase u izlaznu datoteku koristeći „|“ kao separator podataka uvećavajući brojač za 1. Naredba „`catch(Newtonsoft.Json.JsonReaderException)`“ hvata pogreške nastale zbog krivoga formatiranja određenih linija teksta u izvornoj datoteci te omogućava daljnje izvođenje aplikacije.

```
1. while ((line = file.ReadLine()) != null)
2.                 {
3.
4. if (line.StartsWith("{\"WiFi"))
5.                     {
6. try
7.                         {
8. Rootobject rootobject =
   JsonConvert.DeserializeObject<Rootobject>(line);
9. SWifi.WriteLine(id + " | " + rootobject.WiFi.SSID + "|" +
   rootobject.WiFi.BSSID + "|" +
   rootobject.WiFi.capabilities + "|" +
   rootobject.WiFi.level + "|" + rootobject.WiFi.frequency);
10.                             counter++;
11.                         }
12.
   catch(Newtonsoft.Json.JsonReaderException)
13.                         { }
```

Prilog 2. *While* petlja

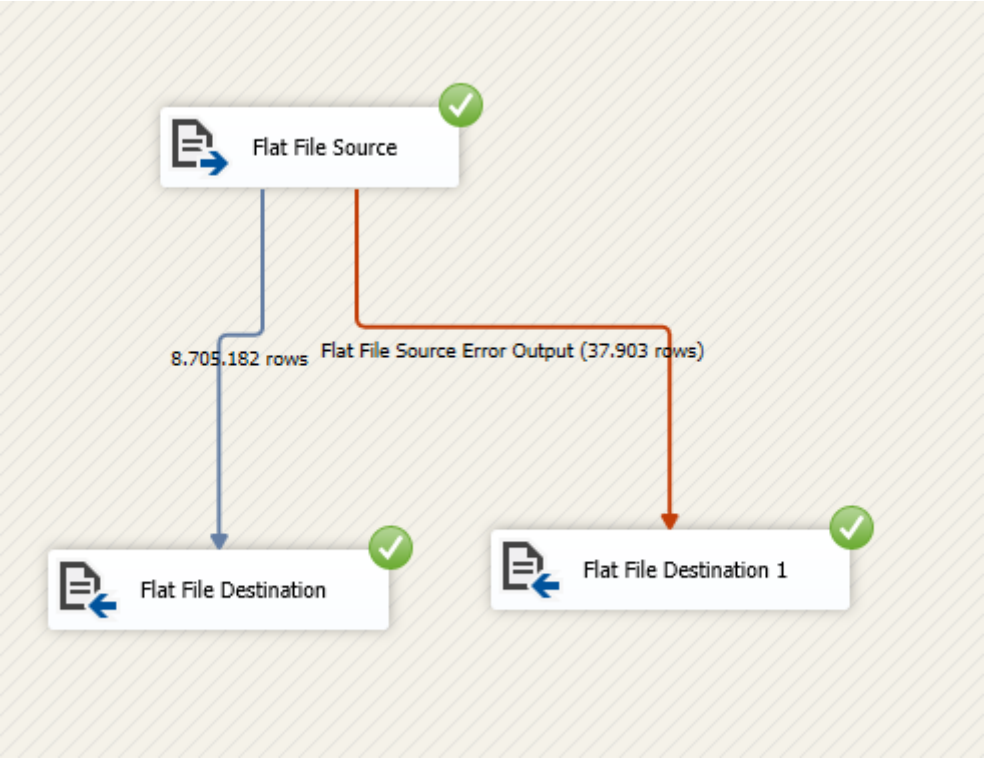
```
1.         public class Rootobject
2.         {
3.             public Wifi WiFi { get; set; }
4.             public Bluetooth Bluetooth { get; set; }
5.             public Application Application { get; set; }
```

```
6.              public Activity Activity { get; set; }
7.              public Call Call { get; set; }
8.              public SMS SMS { get; set; }
9.              public Location Location { get; set; }
10.
11.     }
12.
13.         public class Wifi
14.         {
15.             public string SSID { get; set; }
16.             public string BSSID { get; set; }
17.             public string capabilities { get; set; }
18.             public string level { get; set; }
19.             public string frequency { get; set; }
20.             public string time { get; set; }
```

Prilog 3. Klase za spremanje podataka

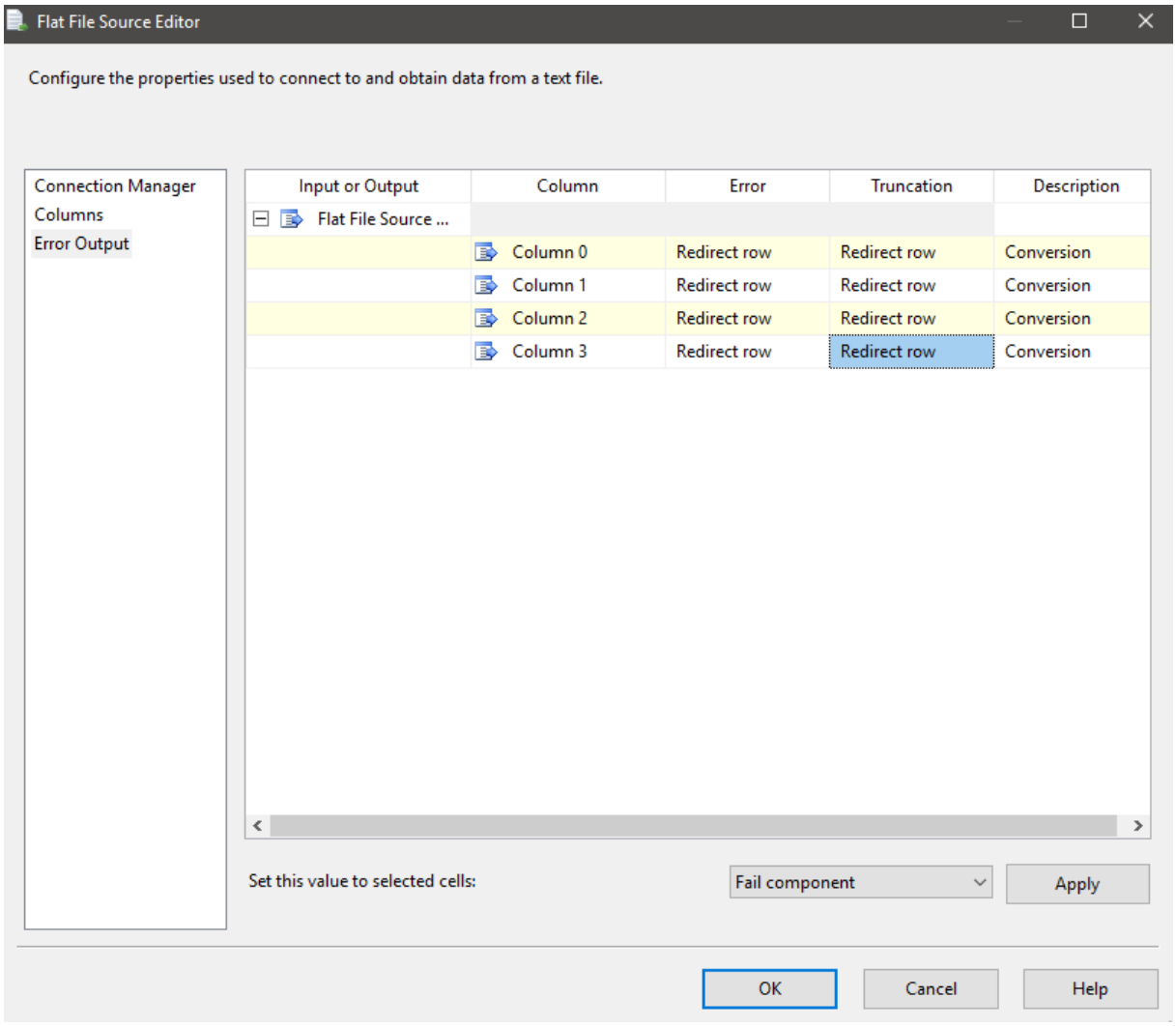
3.2 Rukovanje pogreškama

Zbog već ranije navedenoga lošeg formatiranja izvornih datoteka bilo je potrebno izvršiti filtriranje ispravnih linija od neispravnih koje onemogućavaju uvoz podataka u bazu podataka. Za tu svrhu je korišten SQL Server Integration Services (SSIS) te je *Data Flow task* postavljen kao što je prikazano na slici 5.



Slika 5. Postavke SSIS-a

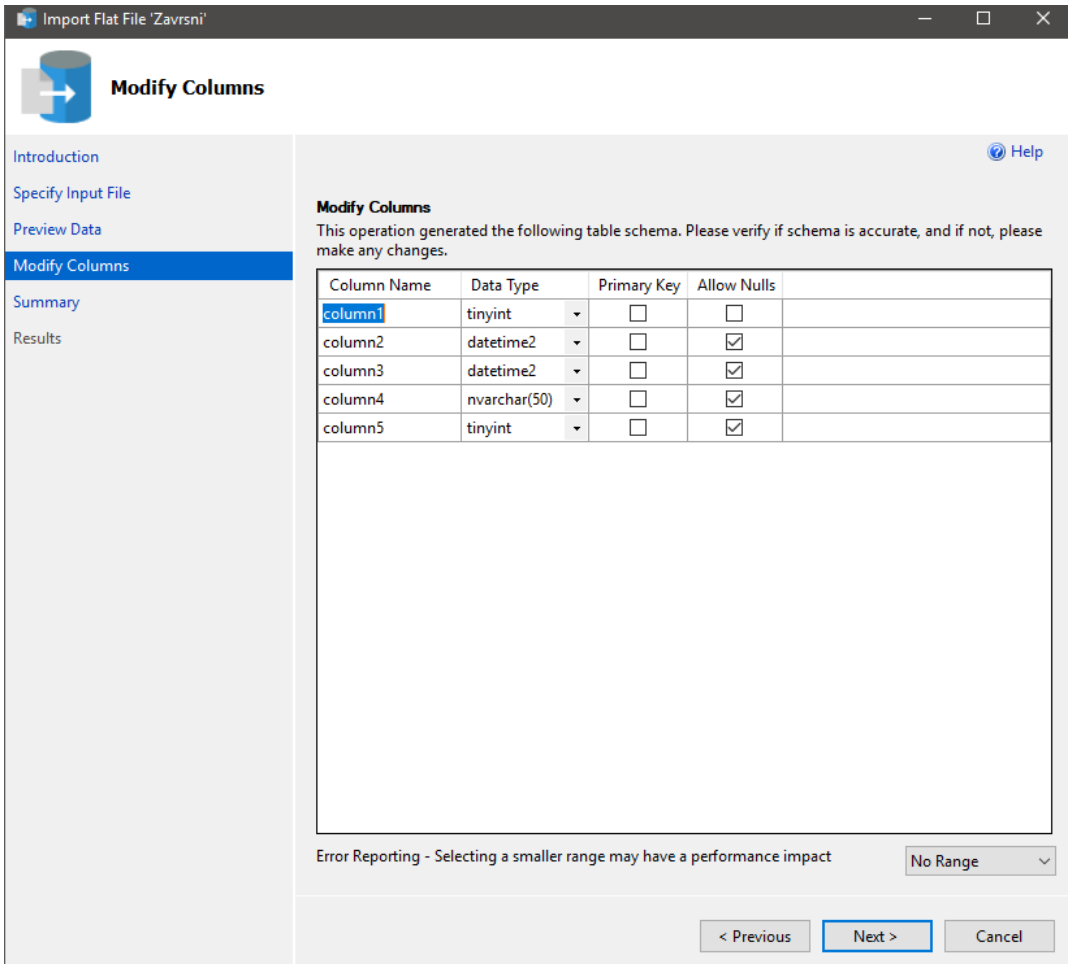
Nakon učitavanja izvorne datoteke pomoću *Data Flow task* bilo je potrebno odrediti postavke za filtriranje grešaka (Slika 6). Filtriranje je vršeno na način su linije preusmjerene u jednu datoteku, a greške u drugu gdje je uz liniju naveden razlog pogreške.



Slika 6. Postavke prilikom nailaska na pogrešku

3.3 Uvoz podataka u bazu podataka

Nakon što je odrađeno filtriranje pogrešaka podaci su spremni za uvoz u bazu podataka. Za navedeno je korištena funkcija „Import Flat File“ u SQL Server Management Studio-u. Također, potrebno je odrediti i karakteristike podataka kao što su ime stupca, tip podatka, omogućavanje „*null*“ vrijednosti, te oznaku primarnog ključa (Slika 7).



Slika 7. Postavke prilikom uvoza podataka

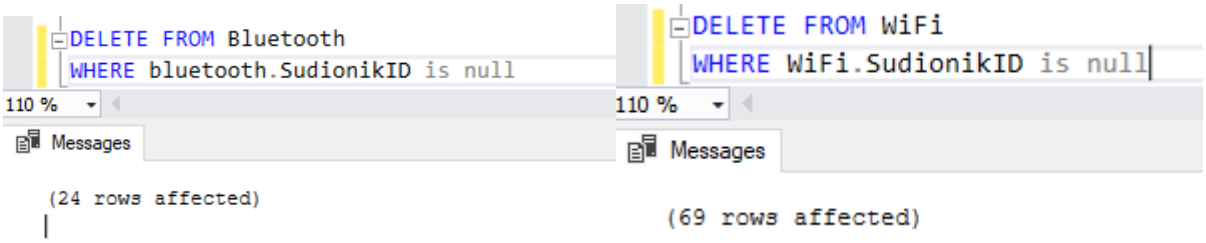
Po završetku uvoza podataka kreirana je još jedna tablica koja sadrži podatke o sudionicima studije s njihovim ID-em te spolom (Prilog 4). Tablica je popunjena koristeći *while* petlju s ID-em sudionika te su svi spolovi postavljeni na „Ženski“. Naredbom „UPDATE“ spol muških sudionika je promijenjen u „Muški“ za odgovarajuće ID-jeve.

```
1. CREATE TABLE Sudionik(
2. ID tinyint NOT NULL PRIMARY KEY,
3. Spol NVARCHAR(10) NOT NULL,
4. )
5.
6. declare @i int
7. set @i = 1
8.
9. while (@i <=35)
10. begin
11.       insert into Sudionik values (@i,'Ženski')
12.       set @i = @i + 1
13. end
```

```
14.
15. UPDATE Sudionik SET spol = 'Muški' WHERE id IN
   (1,3,6,8,9,10,12,20,22)
```

Prilog 4. Kreiranje i popunjavanje tablice Sudionik

Uvozom podataka došlo je do određenih grešaka koje je bilo potrebno ispraviti. Određeni redci u tablicama „*WiFi*“ i „Bluetooth“ sadržavali su vrijednosti „*null*“ te su uklonjeni koristeći funkciju „DELETE FROM“ (Slika 8).

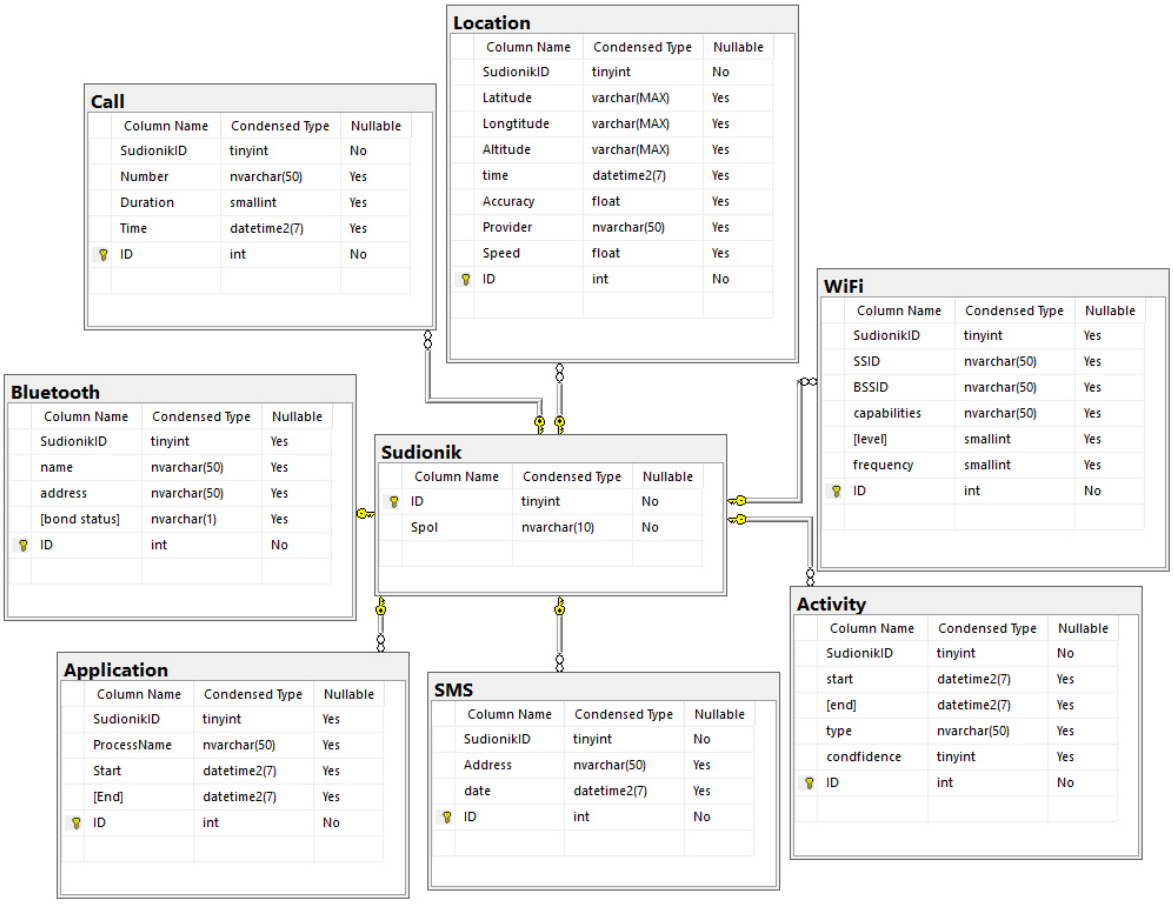


Slika 8. Uklanjanje redaka bez vrijednosti

Za normalan rad baze podataka potrebno je postaviti strani ključ u tablicama te generirati primarni ključ. Funkcija „ALTER TABLE“ omogućava izmjenu već postojeće tablice, a funkcijom „ADD FOREIGN KEY“ postavlja se strani ključ na određeni stupac. Dok funkcija „ADD PRIMARY KEY“ postavlja primarni ključ. Takav primjer nad tablicom „*activity*“ vidljiv je u prilogu 5.

```
1. ALTER TABLE activity
2. ADD FOREIGN KEY (SudionikID) REFERENCES Sudionik(ID);
3. ALTER TABLE activity
4. ADD ID int identity(1,1)not null;
5. ALTER TABLE activity
6. ADD PRIMARY KEY (ID);
```

Prilog 5. Postavljanje stranog i primarnog ključa

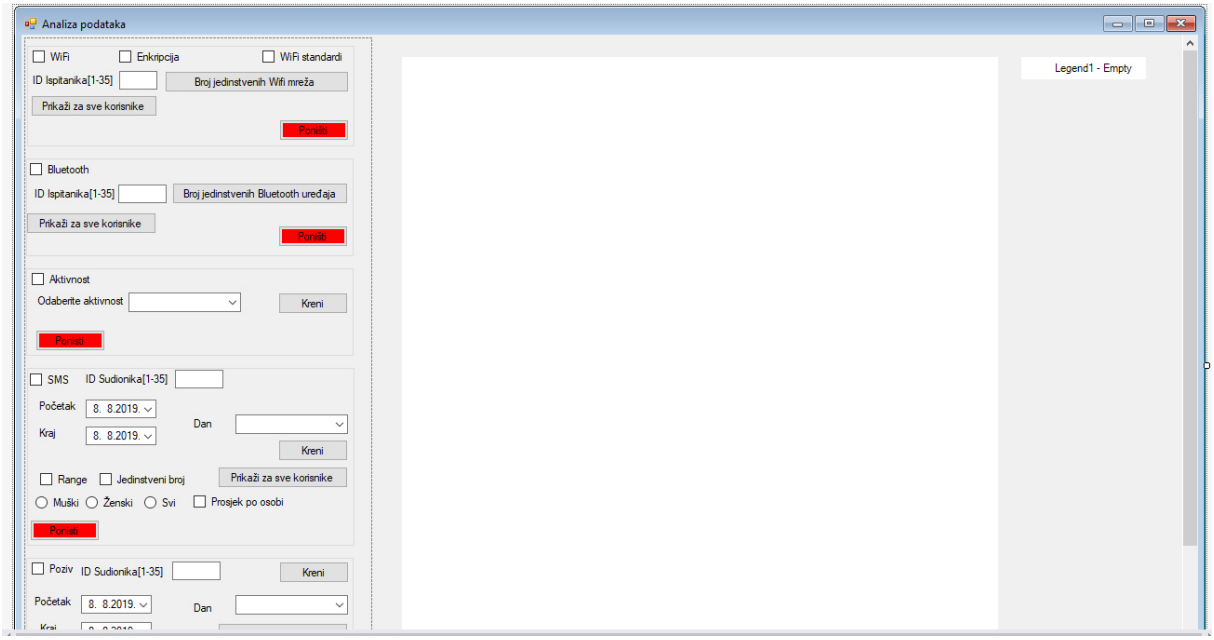


Slika 9. Dijagram baze podataka

Budući da svaki element tablice „Sudionik“ može biti povezan s više elemenata ostalih tablica (ali ne nužno), dok elementi ostalih tablica mogu pripadati isključivo jednome sudioniku [3]. Npr. sudionik je mogao ostvariti više poziva tijekom studije, ali ja svaki poziv mogao ostvariti samo jedan sudionik. Što znači da je veza među entitetima 1:N. Strani ključ je postavljen na „SudionikID“ zato što on predstavlja primarni ključ tablice „Sudionik“ odnosno referencira se na njega te se time ostvaruje veza između tablica (Slika 9) [3].

4. Izrada grafičkog sučelja za prikaz podataka

Kako bi se pojednostavio prikaz velikog broja podataka biti će korišteno grafičko sučelje, odnosno *Windows Forms Application* napisan u C# programskom jeziku te će grafovima unutar sučelja biti prikazani podaci. Pri izradi novoga projekta potrebno je odabrati *Windows Forms Application* te samu *formu* popuniti komponentama i programirati ih (Slika 10).



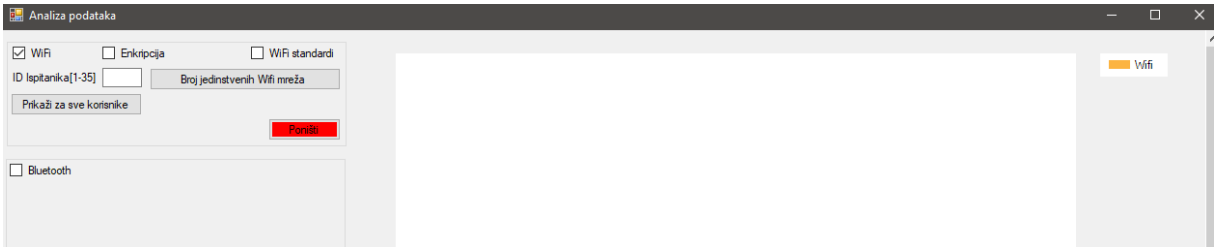
Slika 10. Izgled grafičkog sučelja sa svim kontrolama

Budući da program povlači podatke iz lokalne baze podataka potrebno je u Server Exploreru dodati bazu podataka te u programu dodati „`using System.Data.SqlClient;`“ kako bi se omogućilo korištenje SQL biblioteke. Pri pokretanju forme potrebno je definirati određene komponente koje će biti korištene (prikazane u prilogu 6).

```
1.              panel2.HorizontalScroll.Enabled = false;
2.              panel2.HorizontalScroll.Visible = false;
3.              panel2.HorizontalScroll.Maximum = 0;
4.              panel2.AutoScroll = true;
5.
   chart1.ChartAreas["ChartArea1"].AxisX.Interval = 1;
6.              dateTimePicker1.CustomFormat = "HH:mm tt";
7.              dateTimePicker1.Format =
   System.Windows.Forms.DateTimePickerFormat.Custom;
8.              dateTimePicker1.ShowUpDown = true;
9.              dateTimePicker2.CustomFormat = "HH:mm tt";
10.              dateTimePicker2.Format =
   System.Windows.Forms.DateTimePickerFormat.Custom;
11.              dateTimePicker2.ShowUpDown = true;
12.              dateTimePicker3.CustomFormat = "HH:mm tt";
13.              dateTimePicker3.Format =
   System.Windows.Forms.DateTimePickerFormat.Custom;
14.              dateTimePicker3.ShowUpDown = true;
15.              dateTimePicker4.CustomFormat = "HH:mm tt";
16.              dateTimePicker4.Format =
   System.Windows.Forms.DateTimePickerFormat.Custom;
17.              dateTimePicker4.ShowUpDown = true;
```

Prilog 6. Definiranje komponenti pri učitavanju forme

Aplikacija koristi *autoscroll* što znači da se dostupni dio za *scrollanje* sam postavlja do dijela gdje postoje komponente unutar same forme. `AxisX.Interval = 1;` postavlja interval osi-x tj. oznake na osi-x na zadanu vrijednost, u ovome slučaju 1. Komponenta za odabir vremenskog intervala je u prilagođenome (*custom*) formatu u obliku „`"HH:mm tt"`“ gdje „`"HH`“ označava sate u 24-satnom vremenskom intervalu „`"mm`“ minute u intervalu od 00-59, a „`tt`“ označava prilagođeni format [5]. Grafičko sučelje napravljeno je tako da se tek nakon označavanja *checkboxa* prikazuju komponente za odabir prikaza podataka, te dodaje seriju grafa za određenu vrstu podataka, isto tako nakon odznačavanja iste komponente se uklanjaju (Slika 11)(Prilog 7).



Slika 11. Prikaz komponenti nakon označavanja checkboxa

```
1.    private void checkBox1_CheckedChanged(object sender,
   EventArgs e)
2.          {
3.              if (WiFi_check.Checked == true)
4.              {
5.                  textBox1.Visible = true;
6.                  wifi_enkrip.Visible = true;
7.                  wifi_frek.Visible = true;
8.                  Ponisti.Visible = true;
9.                  label1.Visible = true;
10.                  button1.Visible = true;
11.                  svikorisnici.Visible = true;
12.                  chart1.Series["Wifi"].Enabled = true;
13.
14.              }
15.              else
16.              {
17.                  textBox1.Visible = false;
18.                  wifi_enkrip.Visible = false;
19.                  wifi_frek.Visible = false;
20.                  Ponisti.Visible = false;
21.                  label1.Visible = false;
22.                  button1.Visible = false;
23.                  svikorisnici.Visible = false;
24.                  chart1.Series["Wifi"].Enabled = false;
25.              }
```

Prilog 7. Prikaz/sakrivanje komponenti

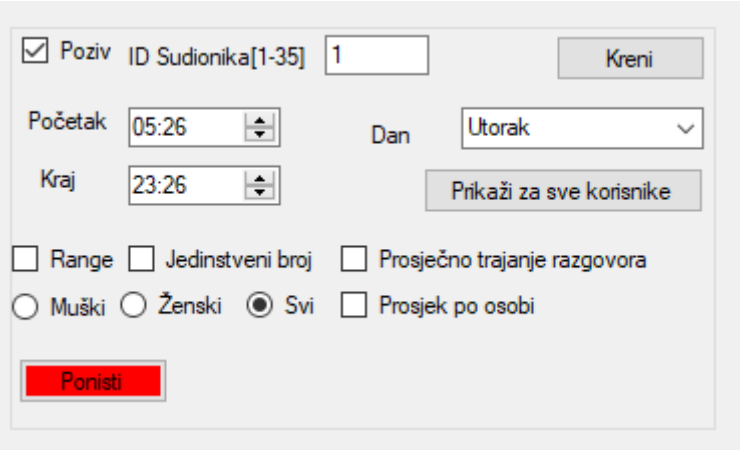
U ovome slučaju (Prilog 8) može se vidjeti da se označavanjem *checkboxa* „Enkripcija“ pokreću SQL komande koje računaju postotak korištenja određene vrste enkripcije. Iz dobivenog *queryja* se nakon toga uzima prvi red dok se ostali ignoriraju [6]. U ovome slučaju korištena metoda odgovara budući da *query* vraća samo jedan red odnosno postotak korištenja određene vrste enkripcije. Na primjer `chart1.Series ["Wifi"].Points.AddXY ("WPA2", wpa2count);` dodaje stupac WPA2 s vrijednosti wpa2count unutar serije *Wifi*. U 20 redu vidljivo je postavljanje formata brojeva na Y-osi na jednu decimalu te dodavanje postotnoga znaka.

```
1.  private void wifi_enkrip_CheckedChanged(object sender,
   EventArgs e)
2.          {
3.              if (wifi_enkrip.Checked == true)
4.              {
5.
```

```
6. SqlConnection conn = new SqlConnection("Data
   Source=DESKTOP-83NNLQJ\\SQLEXPRESS01;Initial
   Catalog=Zavrsni;Integrated Security=True");
7. conn.Open();
8. SqlCommand wpa2 = new SqlCommand("SELECT
   count(*)*100/(select COUNT(*) from WiFi) FROM WiFi WHERE
   capabilities like '%WPA2%'", conn);
9. Int32 wpa2count = (Int32)wpa2.ExecuteScalar();
10. chart1.Series["Wifi"].Points.AddXY("WPA2", wpa2count);
11. SqlCommand wpa = new SqlCommand("SELECT
   count(*)*100/(select COUNT(*) from WiFi) FROM WiFi WHERE
   capabilities like '%WPA%' and capabilities not like
   '%WPA2%'", conn);
12. Int32 wpacount = (Int32)wpa.ExecuteScalar();
13. chart1.Series["Wifi"].Points.AddXY("WPA", wpacount);
14. SqlCommand wep = new SqlCommand("SELECT
   count(*)*100/(select COUNT(*) from WiFi) FROM WiFi WHERE
   capabilities like '%WEP%'", conn);
15. Int32 wepcount = (Int32)wep.ExecuteScalar();
16. chart1.Series["Wifi"].Points.AddXY("WEP", wepcount);
17. SqlCommand bez = new SqlCommand("SELECT
   count(*)*100/(select COUNT(*) from WiFi) FROM WiFi WHERE
   capabilities is null", conn);
18. Int32 bezcount = (Int32)wep.ExecuteScalar();
19. chart1.Series["Wifi"].Points.AddXY("Bez zaštite",
   bezcount);
20. chart1.ChartAreas[0].AxisY.LabelStyle.Format = "{0.0}
   %";
21.
22.             }
23.         }
```

Prilog 8. Izvršavanje SQL naredbi i crtanje grafa unutar WiFi serije

Prikaz podataka za SMS, pozive i aplikacije je sličan te će biti objašnjen na primjeru poziva (Slika 12).



Slika 12. Prikaz postavki pri kreiranju grafa za pozive

Vidljiva su dva gumba za prikaz podataka, gumb „kreni“ te gumb „prikaži za sve korisnike“. Ako se koristi gumb „kreni“ potrebno je upisati ID korisnika čiji se podaci žele vidjeti. Pritiskom na sam gumb „kreni“ vrši se deklariranje varijabli te provjera stanja *checkboxa*. Postavljanje *string*-a „avgcount“ se mijenja ovisno o tome dali je *checkbox* „Prosječno trajanje razgovora“ označen ili nije (Prilog 9)(Red 5-6). Nakon toga se spaja s bazom podataka koristeći *connection string* u kojemu su navedeni naziv servera, naziv baze podataka na koju se spaja te da se koristi *windows* akreditacija za provjeru autentičnosti (Red 15). Ako je prosjek po osobi označen potrebna je provjera spola zbog neravnomjernoga udjela spolova u istraživanju, kako bi se dobio valjan broj za dijeljenje. Odabirom spola dodaje se određeni *string* koji će biti umetnut u SQL upit prilikom kreiranja upita, te se stvara serija za spol (Red 20-23). Ako je „Dan“ izabran vršit će se prikaz poziva samo na taj dan. Svaka stavka u padajućem izborniku ima svoj indeks , počevši od nule. SQL upit zahtjeva ime dana na engleskom jeziku, te se koristi DATEPART funkcija za odabir željenoga dijela datuma, odnosno WEEKDAY za dan u tjednu. Budući da su u SQL-u indeksu nešto drugačiji, točnije u rasponu od 1-7 u odnosu na indekse u padajućem izborniku 0-6, potrebno je uvećati vrijednost indeksa za 1 (Red 25-26). Označavanjem polja „Range“ prikazuju se pozivi u intervalu od jednoga sata u rasponu koji je odabran (Početak, Kraj). To se izvršava tako da se svakim prolaskom kroz petlju dodaje jedan sat odnosno 0.999 sati zbog načina na koji SQL *query* prima vremenske upite (Red 45-52). Funkcija gumba „Prikaži za sve korisnike“ radi na isti način samo se zanemaruje ID ispitanika, odnosno isti nije potreban.

```
1. private void call_kreni_Click(object sender, EventArgs e)
2. {
3. int pros = 1;
4. string spolserie = „Call“;
5. string avgcount = „count(Number)“;
6. if (call_pros_trajanje.Checked==true) { avgcount =
   „AVG(Duration)“; }
7. if (call_prosjek.Checked == true)
8. {
9.
10. if (radioButton1.Checked == true) { pros = 9; spolserie
   = „Call-Muški“; }
11. if (call_zenski.Checked == true) { pros = 26; spolserie
   = „Call-Ženski“; }
12. if (call_sve.Checked == true) { pros = 35; spolserie =
   „Call“; }
13.
14. }
15. SqlConnection conn = new SqlConnection(„Data
   Source=DESKTOP-83NNLQJ\\SQLEXPRESS01;Initial
   Catalog=Zavrsni;Integrated Security=True“);
16. conn.Open();
17. string dan = „“;
18. string dantext = „“;
19. string join = „FULL OUTER JOIN Sudionik ON SudionikID =
   Sudionik.ID“;
20. string spol = „“;
21. if (radioButton1.Checked == true) { spol = „and Spol
   like 'muški'“; spolserie = „Call-Muški“; }
22. if (call_zenski.Checked == true) { spol = „and Spol like
   'ženski'“; spolserie = „Call-Ženski“; }
23. if (call_sve.Checked == true) { spol = „“; spolserie =
   „Call“; }
24.
25. if (call_combo_dan.SelectedItem != null)
26. { int k = call_combo_dan.SelectedIndex; dan = „and
   DATEPART(WEEKDAY, [Time]) = „ + (k + 1) + „“; dantext =
   call_combo_dan.SelectedItem.ToString(); }
27.
28.
29. if (call_check_jedbroj.Checked == true)
30. {
31. avgcount = „count(distinct Number)“;
32. }
33. if (call_check_rang.Checked == false)
34. {
35.
36. SqlCommand broj = new SqlCommand(„select „+avgcount+“
   from call „ + join + „ where CAST([Time] AS time) >= '“ +
   dateTimePicker3.Value.ToString(„HH:mm“) + „' and
   CAST([Time] AS time)<= '“ +
```

```
   dateTimePicker4.Value.ToString(„HH:mm“) + „' and
   SudionikID=“ + call_txtbox_sud.Text + „ „ + dan + spol +
   „“, conn);
37. if (broj.ExecuteScalar() == null) {
   chart1.Series[spolserie].Points.AddXY(„Sudionik“ +
   call_txtbox_sud.Text + „ „ + dantext, 0); }
38. if (broj.ExecuteScalar() == DBNull.Value) {
   chart1.Series[spolserie].Points.AddXY(„Sudionik“ +
   call_txtbox_sud.Text + „ „ + dantext, 0); }
39. else
40. {
41. Int32 brojcount = (Int32)broj.ExecuteScalar();
42. chart1.Series[spolserie].Points.AddXY(„Sudionik“ +
   call_txtbox_sud.Text + „ „ + dantext, brojcount / pros);
43. }
44. }
45. if (call_check_rang.Checked == true)
46. {
47. var pocetak = dateTimePicker3.Value;
48. var kraj = dateTimePicker4.Value;
49. for (var i = pocetak; i < kraj.AddHours(-1); i =
   i.AddHours(0.999))
50. {
51. var z = i.AddHours(1);
52. SqlCommand broj2 = new SqlCommand(„select „+avgcount+“
   from call „ + join + „ where CAST([Time] AS time) >= '“ +
   i.ToString(„HH:mm“) + „' and CAST([Time] AS time) <= '“ +
   z.ToString(„HH:mm“) + „' and SudionikID=“ +
   call_txtbox_sud.Text.ToString() + „ „ + dan + spol + „“,
   conn);
53. Int32 brojcount2 = (Int32)broj2.ExecuteScalar();
54. chart1.Series[spolserie].Points.AddXY(„Sudionik“ +
   call_txtbox_sud.Text + „[„ + i.ToString(„HH: mm“) + „-„ +
   z.ToString(„HH:mm“) + „]h „ + dantext, brojcount2 /
   pros);
55. }
56.
57. }
58. }
59.
```

Prilog 9. Dio kôda za prikaz poziva

Za izradu sučelja za prikaz podataka lokacije korisnika korišten je GMap.NET, .NET *control* otvorenoga kôda. Pritiskom na gumb „Ruta“ vrši se spajanje sa SQL bazom podataka te se postavljaju osnovne funkcije GMap.NET-a; odabir *providera* čije će se mape koristiti. U ovome radu koristiti će se mapa od Google-a. Podešavanje moda pristupa odnosno otkuda se podatci povlače prikazan je u prilogu 10, redovi 3-6. U redovima 9 i 10 vidljivo je deklariranje

overlaya ruta i poligona, nakon čega se stvara nova lista objekata koristeći već predefiniranu klasu „PointLatLng“ koja prima geografsku širinu i geografsku dužinu tipa *double*. Za dohvat podataka je potrebno napisati SQL *query* (red 14), te deklarirati čitač koji će očitavati rezultate *queryja* (Red 16). Budući da se rute i poligoni crtaju za lokacije prikupljene tijekom cijele studije, za učitavanje svih tih lokacija potrebna je petlja. U ovome slučaju korištena je *while* petlja gdje u svakoj iteraciji učitavaju redovi koje SQL *query* vraća kao *string* te se konvertiraju u tip *double* kako bi se mogli dodati unutar klase „PointLatLng“ te u listu (red 19-26). Pri ispisu rute prvo se vadi prvi element u listi te se iz njega izvlače geografska širina i geografska dužina kako bi se početna pozicija mape postavila na te koordinate (red 34-37). Dok je za crtanje same rute dovoljno dodijeliti listu u kojoj se nalaze geografske točke (red 38).

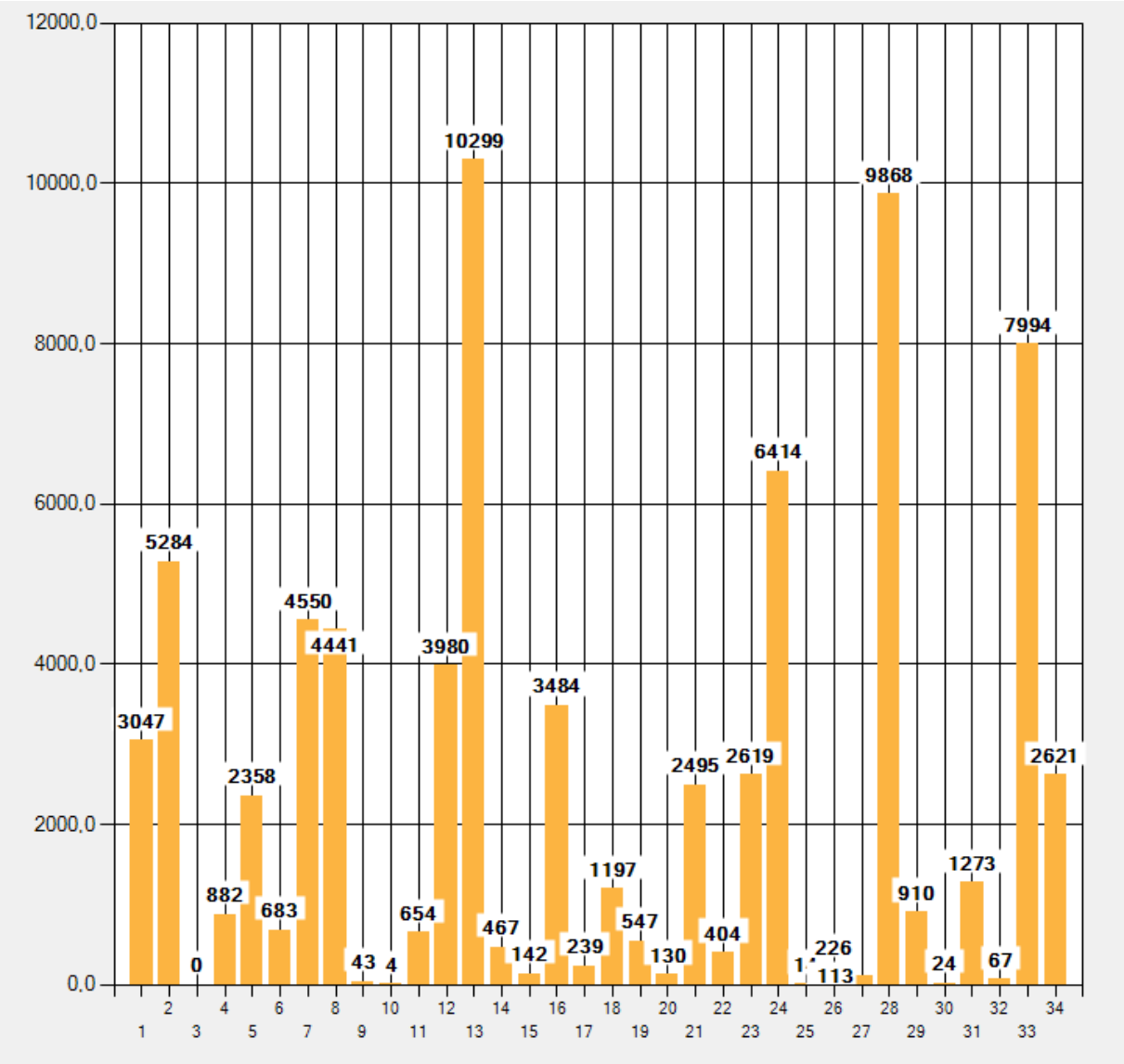
```
1. private void Ruta_Click(object sender, EventArgs e)
2.         {
3.             SqlConnection conn = new SqlConnection(„Dana
   Source=DESKTOP-83NNLQJ\\SQLEXPRESS01;Initial
   Catalog=Zavrsni;Integrated Security=True“);
4.             conn.Open();
5.             gMapControl1.MapProvider =
   Gmap.NET.MapProviders.GoogleMapProvider.Instance;
6.             Gmap.NET.Gmaps.Instance.Mode =
   Gmap.NET.AccessMode.ServerOnly;
7.             gMapControl1.ShowCenter = false;
8.
9.             GmapOverlay routes = new
   GmapOverlay(„routes“);
10.              GmapOverlay polygons = new
   GmapOverlay(„polygons“);
11.              List<PointLatLng> points = new
   List<PointLatLng>();
12.              string id = textBox1.Text.ToString();
13.
14.              using (SqlCommand sqlCommand = new
   SqlCommand(„select Latitude,Longtitude from Location
   where SudionikID = „ + id + „ order by time“, conn))
15.              {
16.                  using (var reader =
   sqlCommand.ExecuteReader())
17.
18.                  {
19.                      while (reader.Read())
20.                      {
21.
22.                          string value1 =
   reader.GetString(0);
```

```
23.                         string value2 =
   reader.GetString(1);
24.                         double val =
   Convert.ToDouble(value1,
   System.Globalization.CultureInfo.InvariantCulture);
25.                         double val1 =
   Convert.ToDouble(value2,
   System.Globalization.CultureInfo.InvariantCulture);
26.                         points.Add(new PointLatLng(val,
   val1));
27.
28.
29.
30.                     }
31.                 }
32.             }
33.
34.             PointLatLng elem = points[0];
35.             double lat = elem.Lat;
36.             double lng = elem.Lng;
37.             gMapControl1.Position = new
   Gmap.NET.PointLatLng(lat, lng);
38.             GmapRoute route = new GmapRoute(points,
   „ruta");
39.             route.Stroke = new Pen(Color.Red, 3);
40.             routes.Routes.Add(route);
41.             gMapControl1.Overlays.Add(routes);
42.         }
```

Prilog 10. Kôd za iscrtavanje ruta i poligona

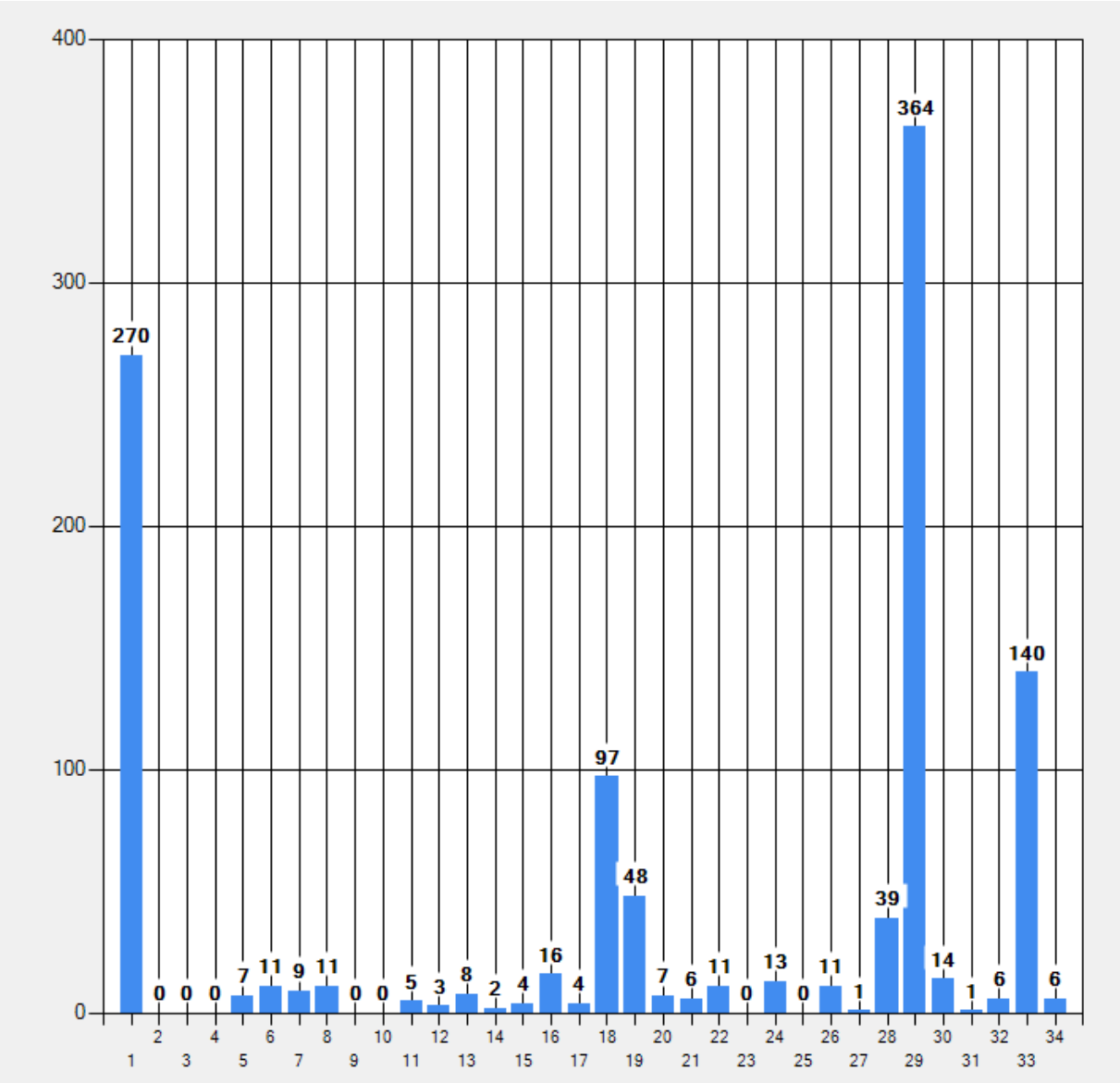
5. Analiza podataka

Prije samoga početka analize podataka treba naglasiti razlike u količini podataka prikupljenih od strane ispitanika. Određeni ispitanici pružili su puno veći broj podataka za razliku od drugih, dok su nekim ispitanicima na primjer nedostajali kompletni skupovi podataka određene vrste. Navedeno je potrebno uzeti u obzir pri analizi rezultata, kao i činjenicu da su ispitanici bili volonteri. Još jedan važan podatak koji se mora uzeti u obzir je vrijeme prikupljanja podataka. Ono se odnosi na period tijekom 2013 i 2014 godine uz par iznimaka. S obzirom na to da je najviše podataka prikupljeno o *WiFi* mrežama, analiza podataka istih trebala bi polučiti iznimno točne podatke. Podaci su prikupljani tako da je senzor skenirao sve mreže u dometu te zapisivao podatke o njima. Analizom je ustanovljeno da od svih mreža, njih 9% je koristilo usmjerivače koji su radili na 5GHz dok je 91% usmjerivača radilo u frekvencijskom pojasu od 2.4GHz. 5GHz usmjerivači pružaju veće brzine prijenosa te veći broj kanala (23) u odnosu na 3 kanala kod 2.4 GHz. U frekvencijskom pojasu od 2.4 GHz postoji određeno „zagušenje“ zbog velike količine uređaja koji rade u tom frekvencijskom pojasu. No ukoliko je potreban veći domet ili postoji puno zidova odnosno fizičkih barijera na putu signala, 2.4 GHz je bolji izbor [7]. Što se razine zaštite tiče odnosno korištene enkripcije rezultat je sljedeći: WPA2 61%, WPA 20%, WEP 9%, bez zaštite 9%. Uz opasku da su u kategoriju bez zaštite ušli svi podaci koji su imali vrijednost enkripcije *null*. Uzimajući u obzir koliko je lako probiti WEP zaštitu, skoro 18% uređaja praktički nema nikakvu zaštitu dok 20% koristi nešto bolju WPA enkripciju koja je i dalje puna mana. Zanimljiv podatak koji se još može izvući iz tih podataka je količina *WiFi* mreža u okruženju sudionika. Radi se o tisućama jedinstvenih mreža koje su sudionici registrirali tijekom studije (60 dana, Slika 13.).



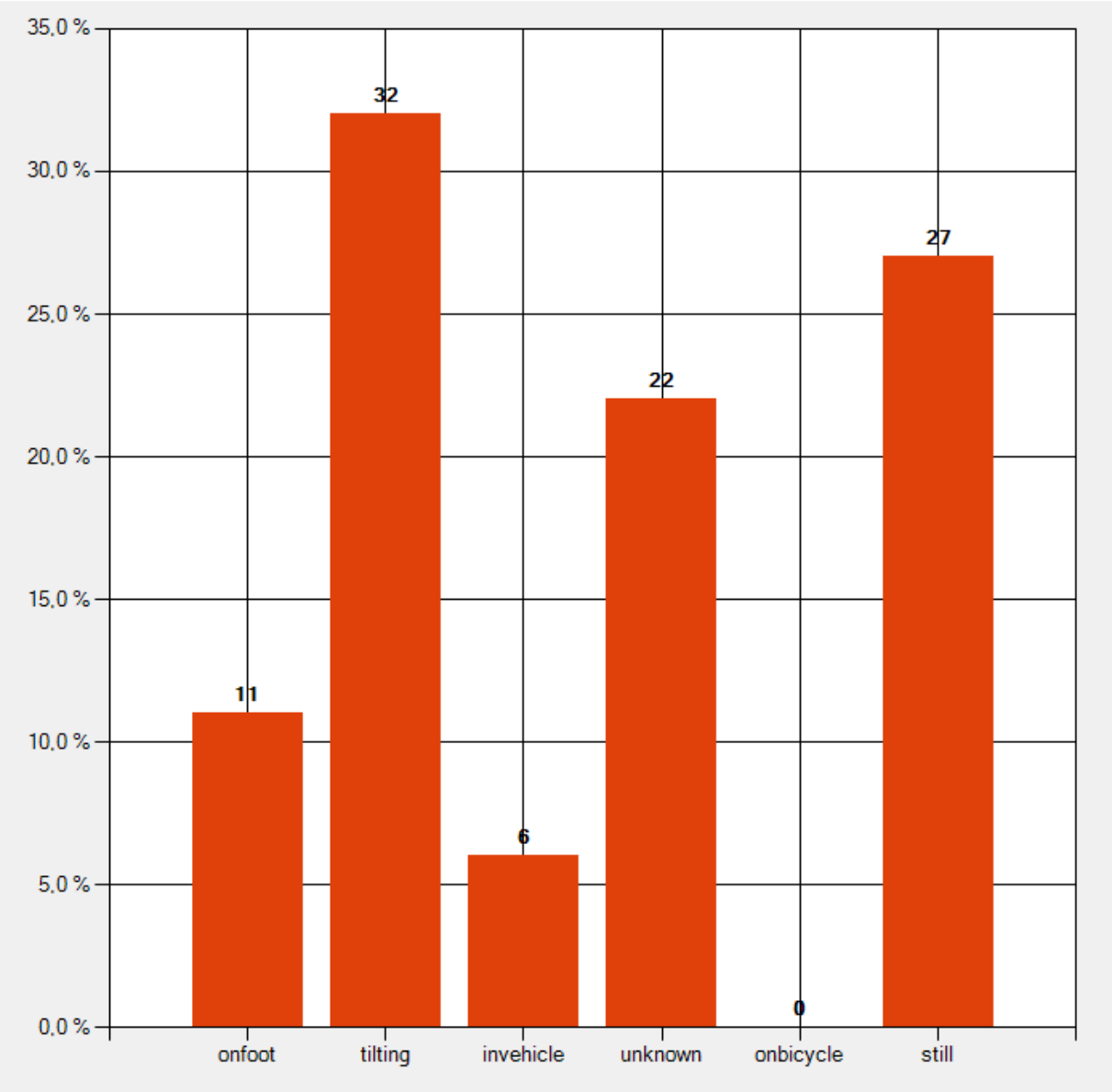
Slika 13. Broj jedinstvenih WiFi mreža po korisniku

Na Y-osi nalazi se ukupan broj skeniranih mreža, dok je na X-osi ID-korisnika (Slika 13). U usporedbi s time moguće je i povući paralelu s brojem ukupno skeniranih *bluetooth* uređaja (na Y-osi) koji je znatno manji (Slika 14). Iako *bluetooth* nije konstantno uključen na uređajima već većinom samo tijekom korištenja.



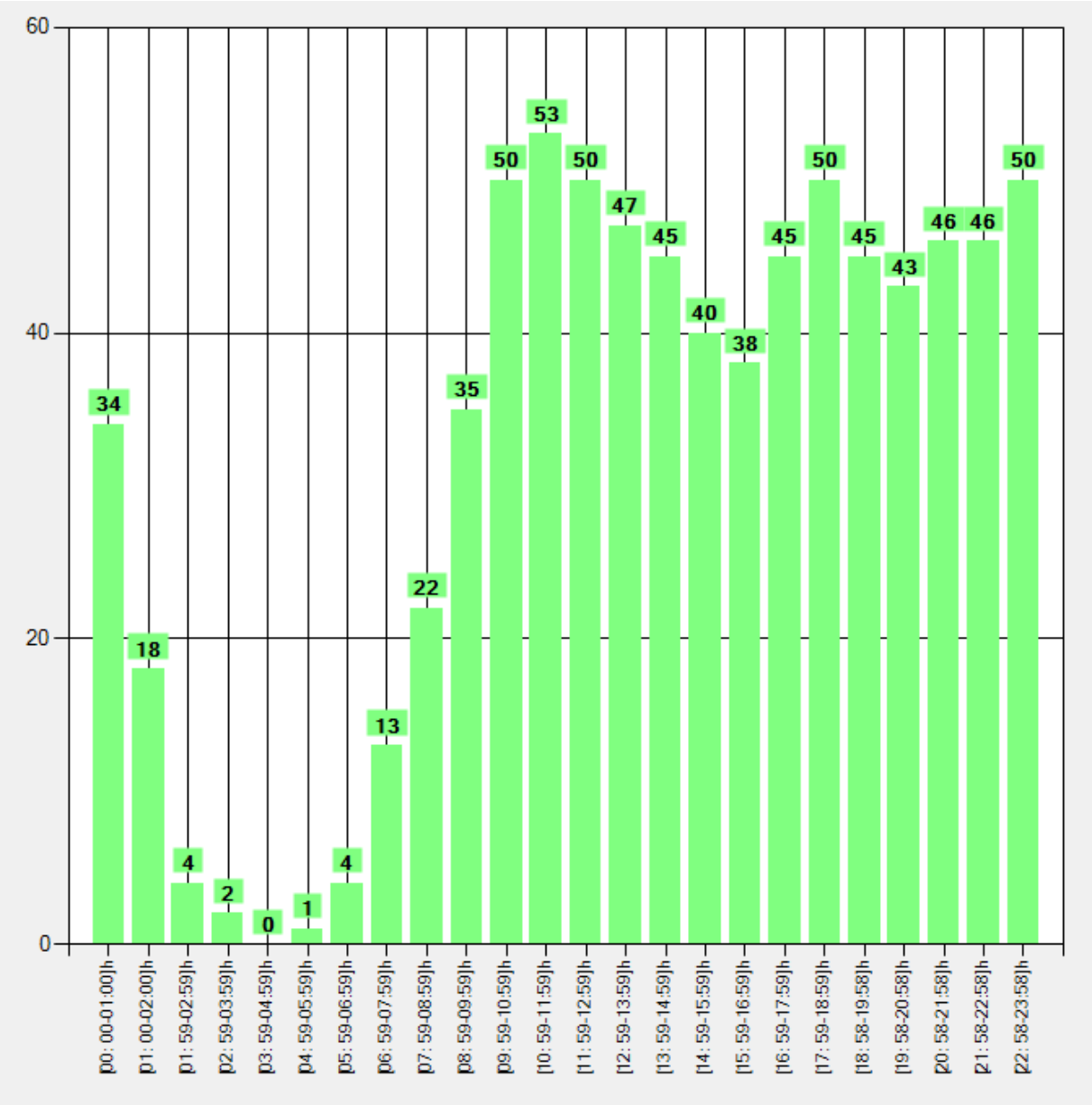
Slika 14. Broj jedinstvenih Bluetooth uređaja po korisniku

Dobiveni podaci o aktivnosti (Slika 15) prikupljeni su korištenjem žiroskopa, prikazuju postotak određenog uzorka kretanja. Podatke se može iskoristiti kako bi se odredio određeni mod kretanja korisnika. Iz grafa je vidljivo da se korisnik većinom kretao pješice (11%), dok je ukupno kretanje iznosilo 19%. Korisnik je većinu vremena proveo ne krećući se (59%) dok je za 22% podataka aktivnost nepoznata.



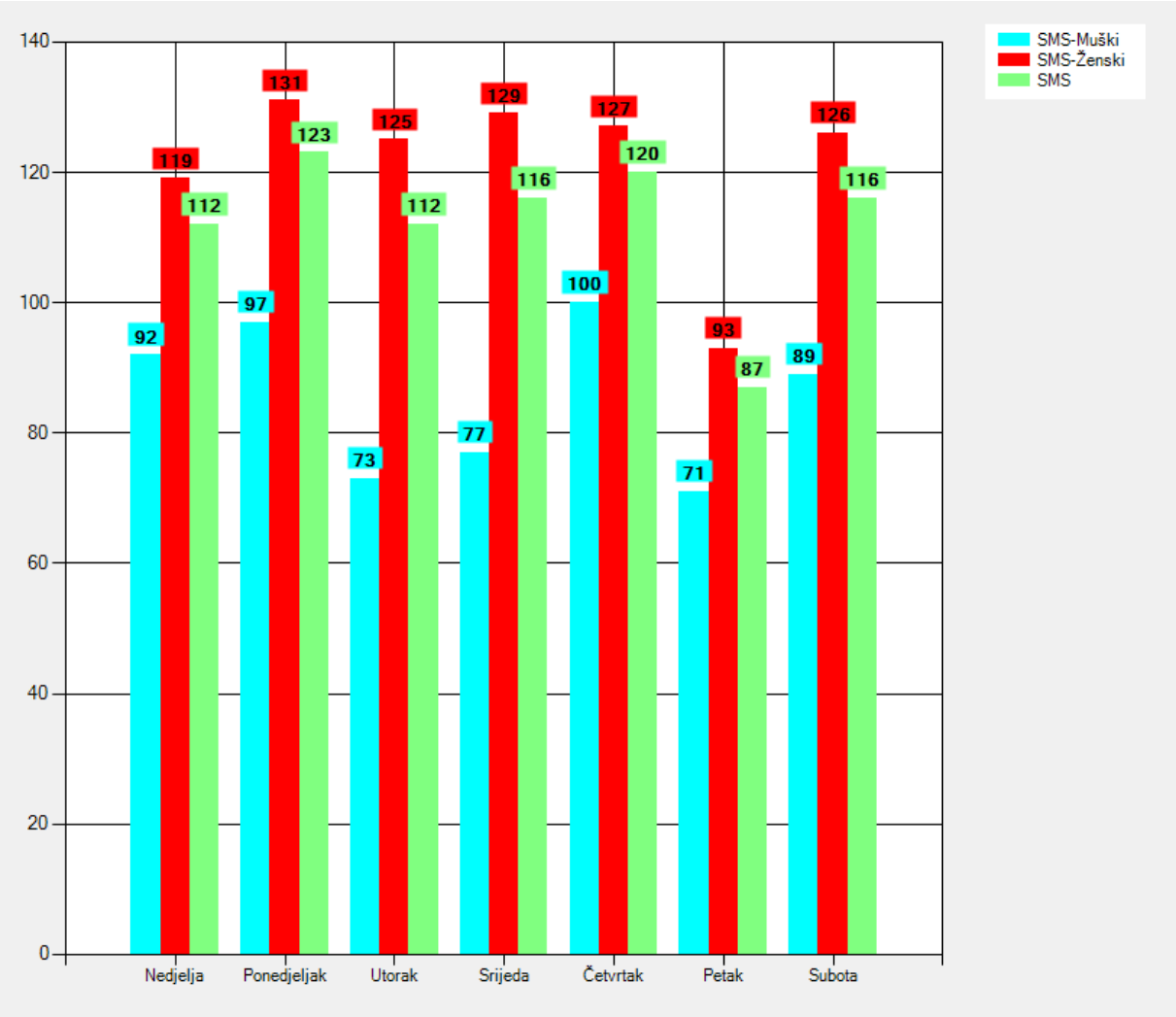
Slika 15. Aktivnost – žiroskop

Što se broja poruka tiče može se reći da je dosta ujednačen od 11 sati pa sve do ponoći, te da počinje padati nakon ponoći, iako je do 2 sata još uvijek dosta visok (Slika 16). Iz toga bi se moglo zaključiti da su sudionici spavali većinom između jedan sat i osam/devet ujutro.



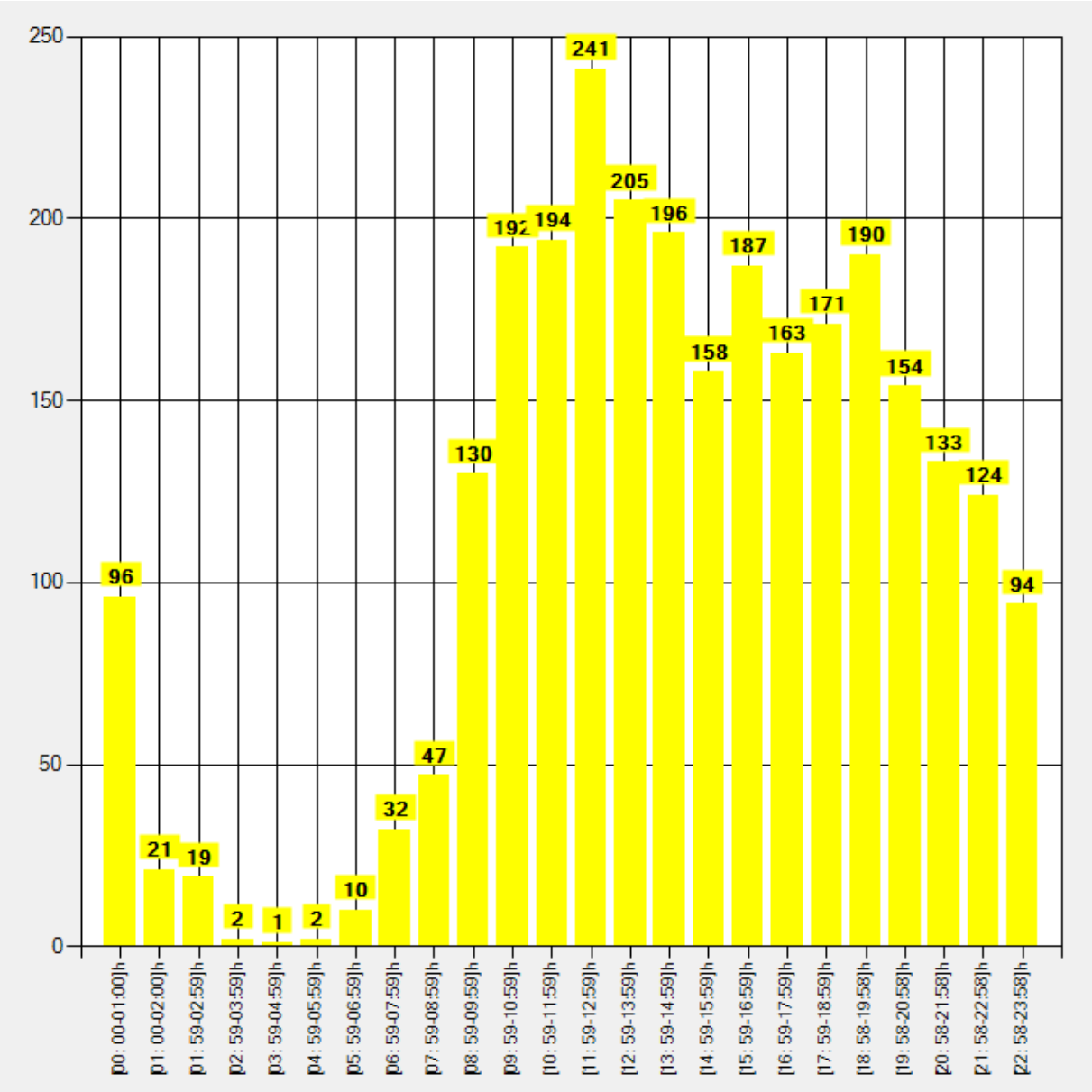
Slika 16. Vremenska raspodjela poruka

Prosječni broj poruka varira ovisno o spolu sudionika. Tijekom studije prosječan broj poruka poslanih od strane ispitanika ženskoga spola iznosio je 854 poruke dok je za muške ispitanike iznosio 603 poruke. Prosječan broj poruka svih ispitanika iznosio je 789. Gledajući poruke prema jedinstvenome broju tu su muški ispitanici bili u neznatnoj prednosti, 50 naprema 48 poruka. Gledajući raspodjelu poruka tijekom tjedna može se vidjeti da je najveći broj poruka (Y-os) poslan ponedjeljkom uz prilično jednaku raspodjelu ostalim danima izuzev petka tijekom kojega je bilo dosta manje poruka (Slika 17).



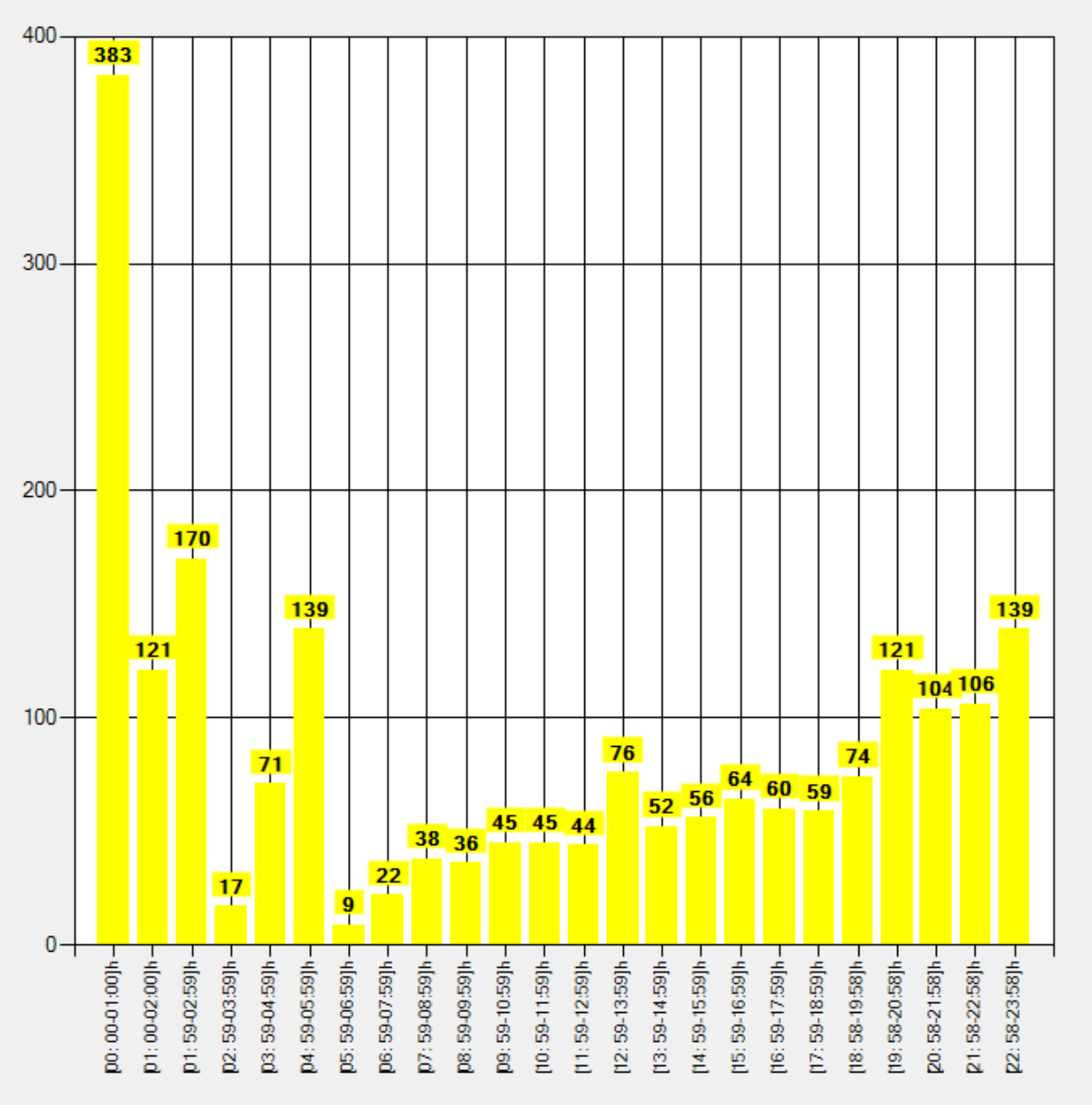
Slika 17. Raspodjela poruka tijekom tjedna

Gledajući vremensku raspodjelu poziva (Slika 18) može se vidjeti da je najveći broj poziva (Y-os) ostvaren između 12 i 13 sati, te da prosjek poziva iznosi 241 poziv. Što se uklapa u već navedeni period od jedan do devet ujutro u kojemu većina korisnika vjerojatno spava.



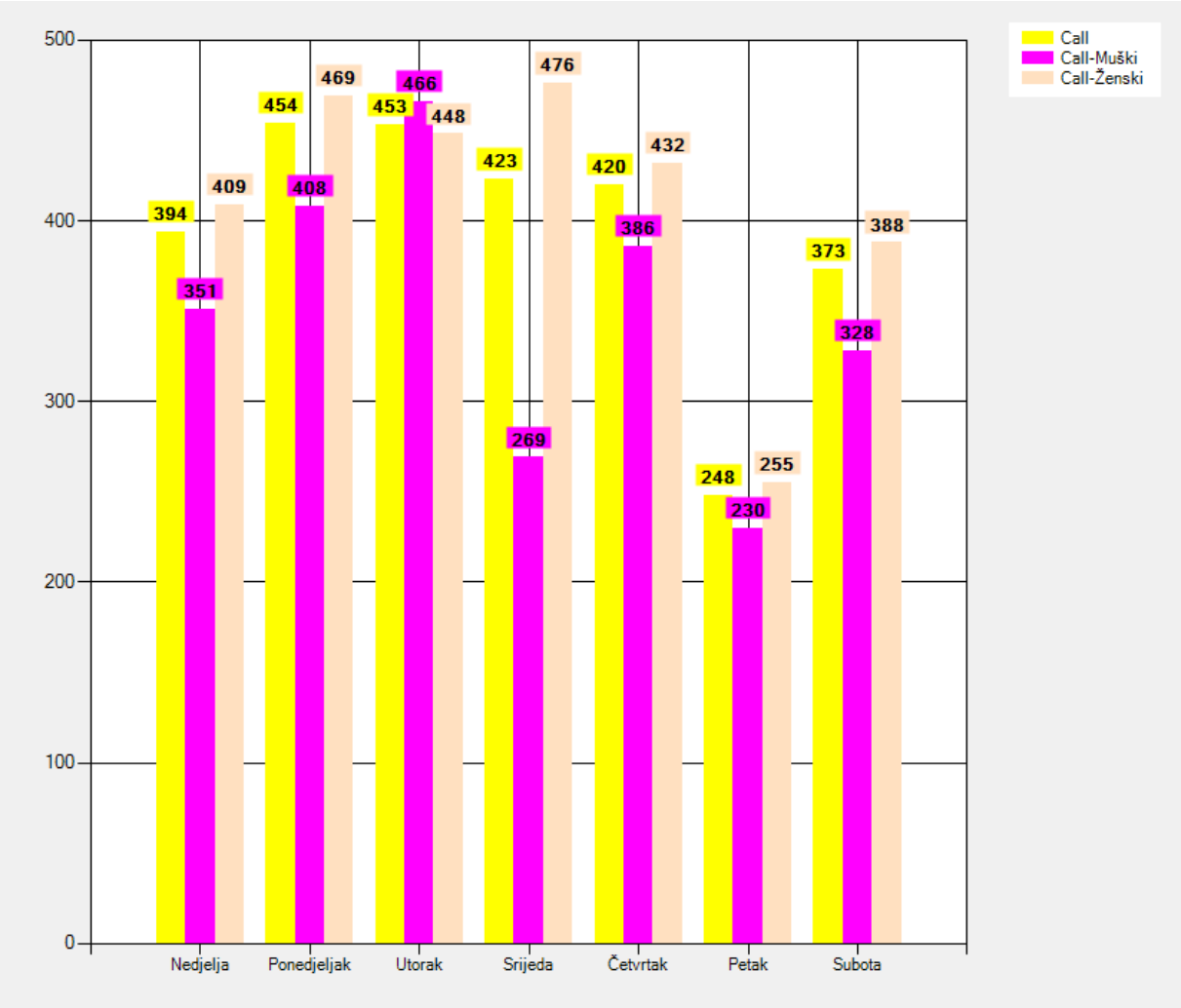
Slika 18. Vremenska raspodjela poziva

Ako se umjesto broja poziva promatra vrijeme razgovora u satu vidljivo je da su u noćnim satima razgovori znatno duži iako je u tome razdoblju manji broj samih razgovora (Slika 19).



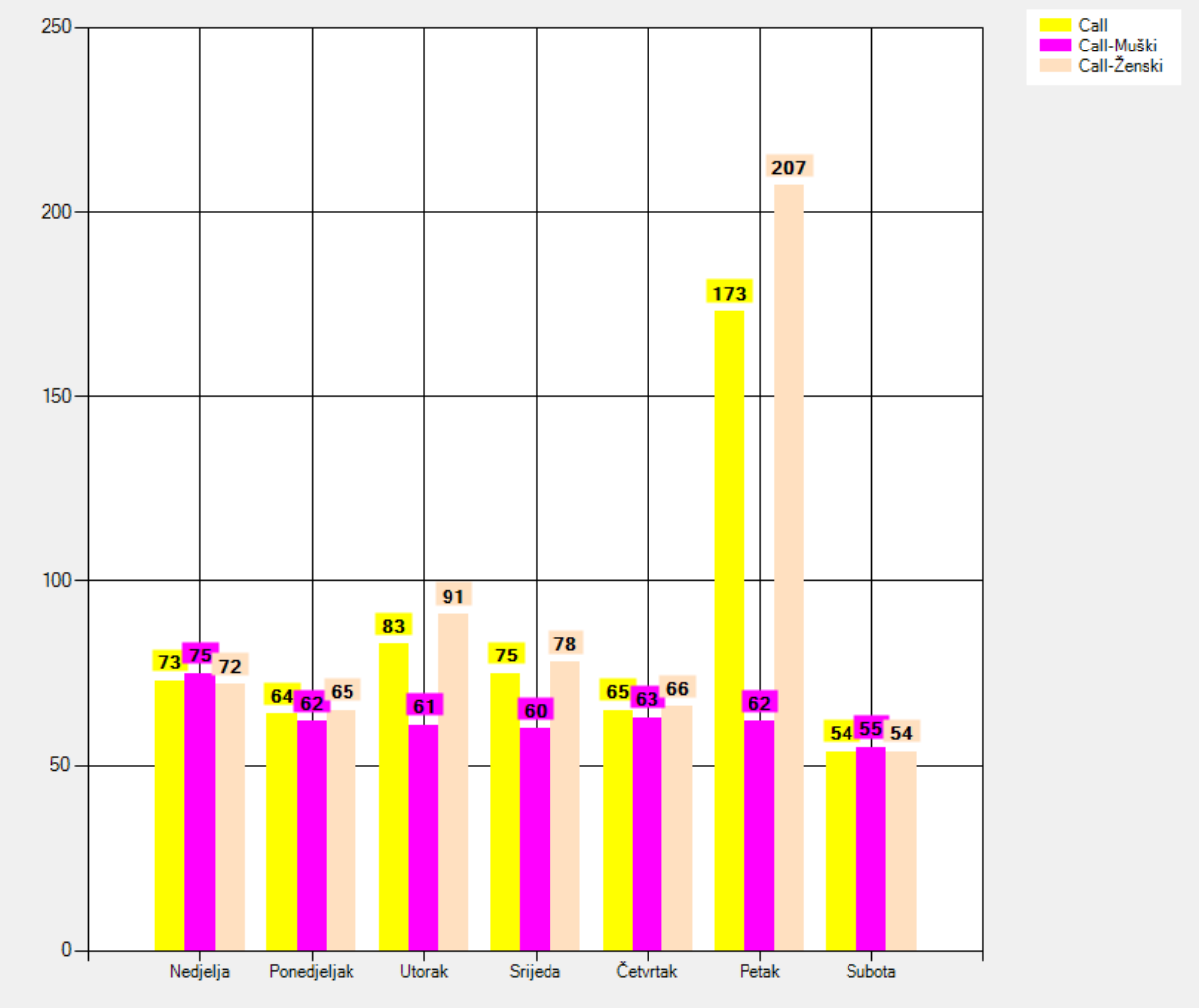
Slika 19. Vremenska raspodjela trajanja razgovora u sekundama

Gledajući raspodjelu trajanja razgovora između spolova, vidljivo je da je prosječno trajanje poziva muških ispitanika 62 sekunde, a ženskih 83 sekunde. Gledajuće samo jedinstvene brojeve nailazimo na istu stvar kao i kod SMS poruka. Muški ispitanici su imali neznatno više poziva, 59 naspram 57. Gledajući raspodjelu prema danima u tjednu može se vidjeti ponavljanje istoga uzorka kao i kod SMS-a, najmanji broj poziva petkom te donekle ujednačeni pozivi tijekom ostatka tjedna, iako su ovdje vidljive razlike između spolova (Slika 20).



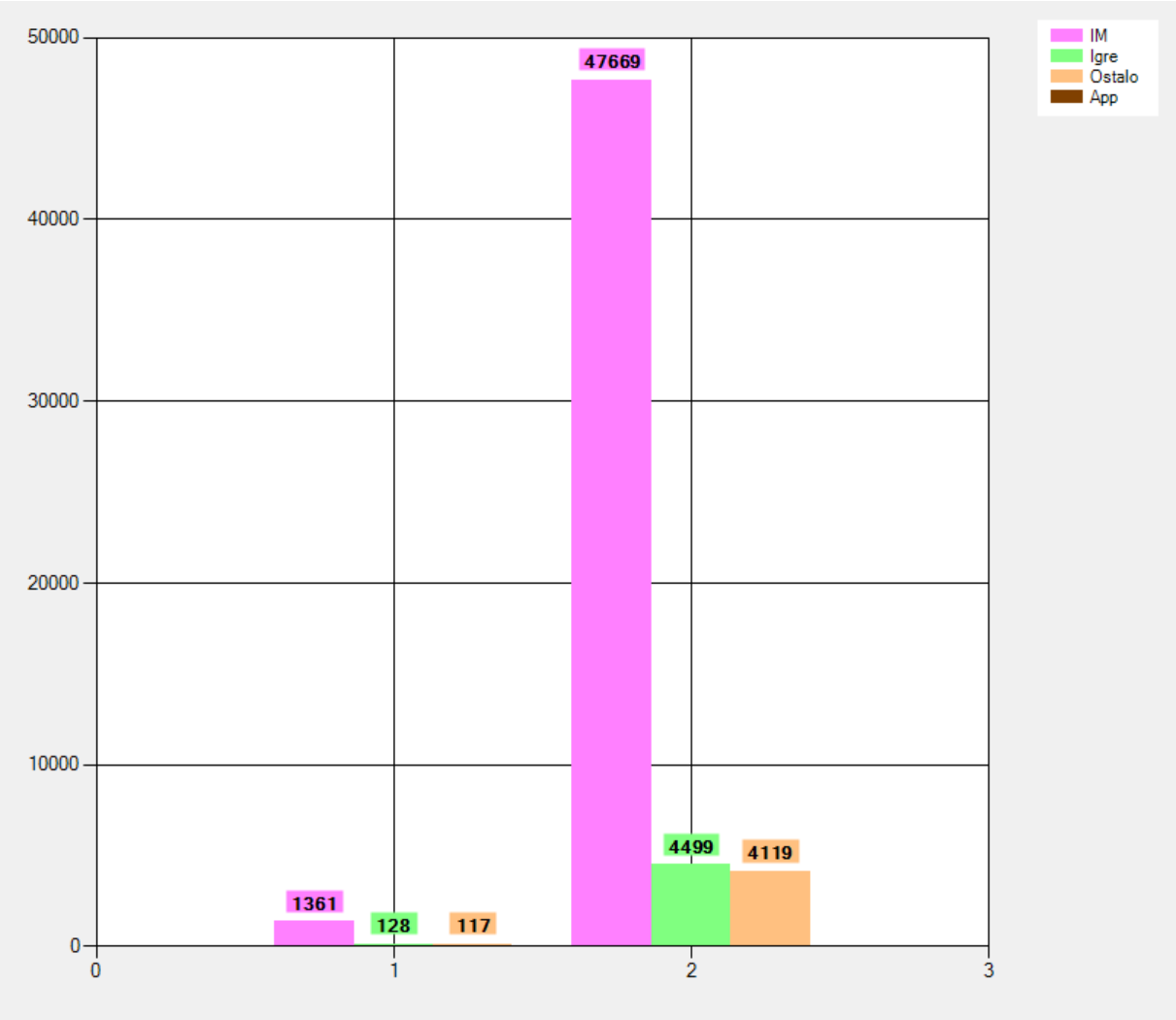
Slika 20. Raspodjela poziva tijekom tjedna

Gledajući prosječna vremena trajanja razgovora (Y-os) može se vidjeti da su sredinom tjedna nešto duži razgovori te ekstrem petkom tijekom kojega su razgovori duži nego u ostatku tjedna (Slika 21).



Slika 21. Raspodjela vremena trajanja razgovora tijekom tjedna

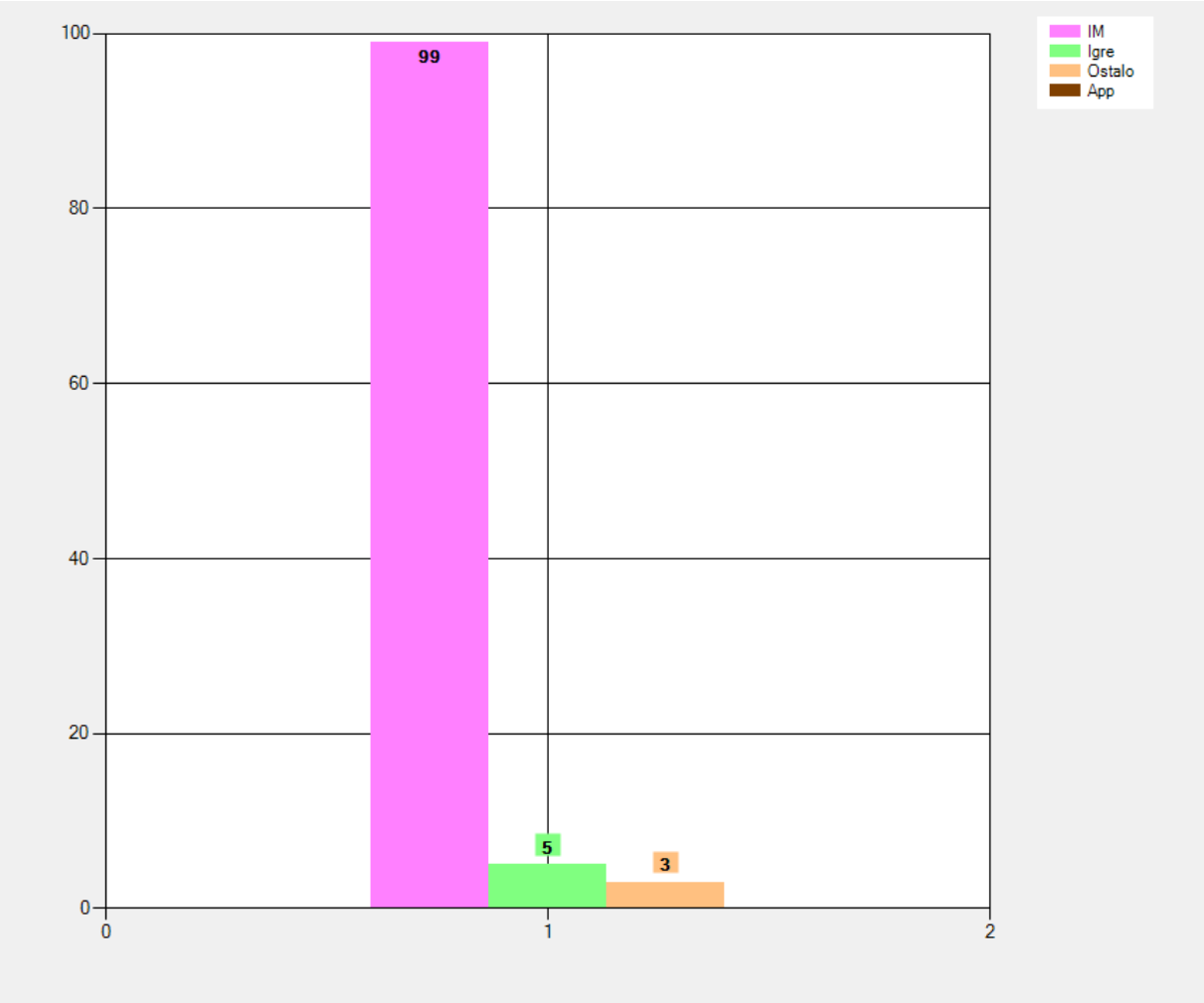
Budući da je aplikacije nemoguće automatski kategorizirati i iz samih naziva procesa iščitati o kojoj se aplikaciji radi, te je to učinjeno ručno. Iz toga razloga analiza je vršena za samo stotinjak najkorištenijih aplikacija. Ako se pogleda raspodjela prema tri kategorije u koje su aplikacije podijeljene vidljivo je da su Instant messaging (IM) aplikacije one s najvećim brojem otvaranje (prikazano na Y-osi, Slika 22 desno) što je i logično budući da se aplikacije pale svaki puta kada se šalje ili gleda primljena poruka. Prosjek po korisniku vidljiv je s lijeve strane. Gledajući raspodjelu prema spolovima može se vidjeti da su muški sudionici u IM aplikacije ušli prosječno 367 puta tijekom studije dok za ženske ta brojka iznosi 1706 što je značajna razlika. Što se igara tiče ta brojka je isto na strani ženskih ispitanika, prosječno 141 pokretanje naspram 91 muških sudionika studije. Ostale aplikacije su također više korištene od strane ženskih ispitanika 151 naprema 21.



Slika 22. Broj otvaranja aplikacija

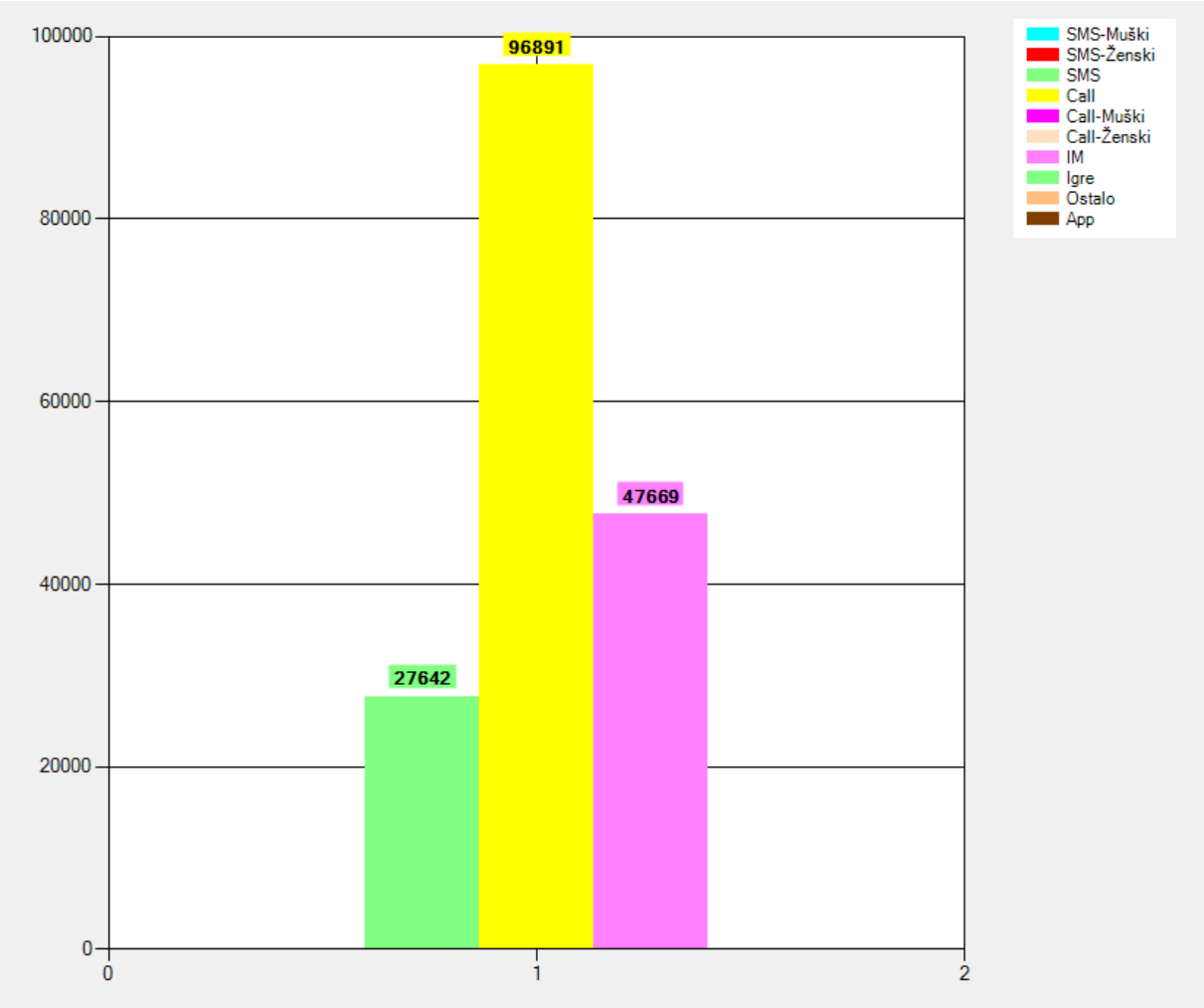
Ako se pogleda vremensko trajanje korištenja tih aplikacija (prikazano na Y-osi) vidljiva je podjednaka raspodjela kao i kod samoga pokretanja (Slika 23). Prosječno vrijeme korištenja IM aplikacija iznosi 99 sati, 5 sati za igre što je pomalo začuđujuće s obzirom na to da bi pretpostavka bila da će se u igrama provest duže vremena nego u IM aplikacijama. Za ostale aplikacije taj broj iznosi 3 sata. Kod ove analize treba uzeti u obzir način na koji android operacijski sustav upravlja aplikacijama, tj. postoji mogućnost da se na određene aplikacije gledalo kao aktivne dok nisu bile primarno korištene već samo u pozadini te je to moglo utjecati na rezultate u određenom postotku. Gledajući raspodjelu prema spolovima ženski sudionici koristili su IM aplikacije u prosjeku 132 sata tijekom studije dok su ih muški sudionici koristili samo 3h, kod igara taj broj je iznosio 5 sati za oba spola dok su ostale aplikacije muški sudionici koristili 41 minute, a ženski 280 minutu. Gledajući te podatke, te podatke o SMS

porukama i pozivima može se reći da su ženski ispitanici provodili veću količinu vremena na svojim pametnim telefonima tijekom studije.



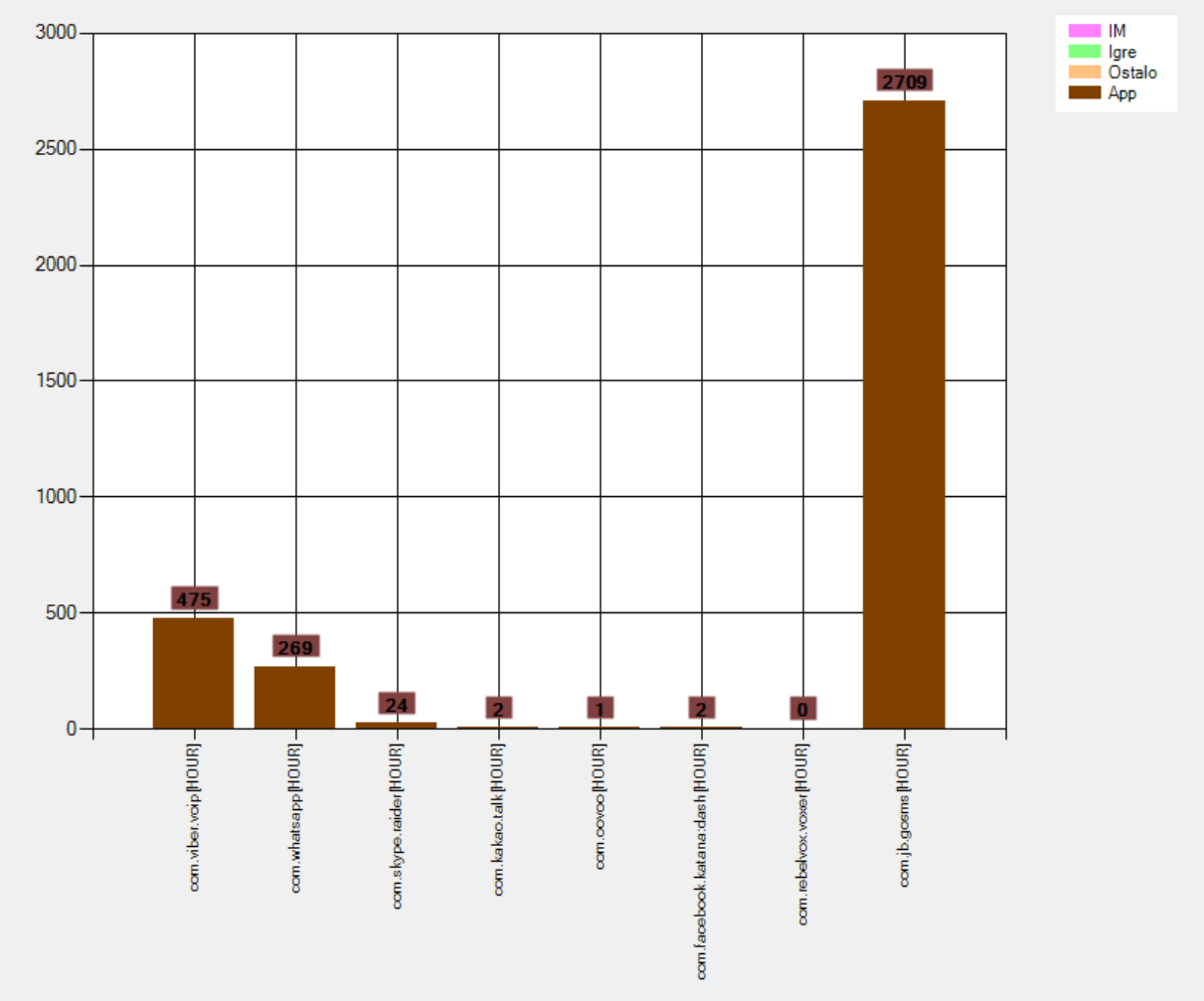
Slika 23. Prosječno vrijeme korištenja aplikacija tijekom studije u satima

Uspoređujući korištenje IM aplikacija s brojem SMS poruka i poziva (Slika 24) moguće je doći do zaključka kako su već u to vrijeme IM aplikacije počele preuzimati primat na tržištu poruka, dok su klasični telefonski pozivi i dalje primarna vrsta poziva. Na Y-osi grafa vidljivi su broj SMS poruka te broj poziva svih sudionika tijekom studije, dok je za IM aplikacije prikazan broj otvaranja aplikacije.



Slika 24. Usporedba korištenja IM aplikacija te klasičnih poziva i poruka

Usporedbom najkorištenijih IM aplikacija na slici 25 vidljivo je da većina korištenja otpada na GO SMS, viber i whatsapp, dok još mali dio udjela ima skype. Zanimljivo je da je facebook korišten samo 2 sata od strane svih sudionika ove studije, uz napomenu da je u to vrijeme postojala samo jedna aplikacija u sklopu koje je bio i IM, tako da ta 2 sata ne moraju nužno biti niti samo korištenje facebooka za IM.



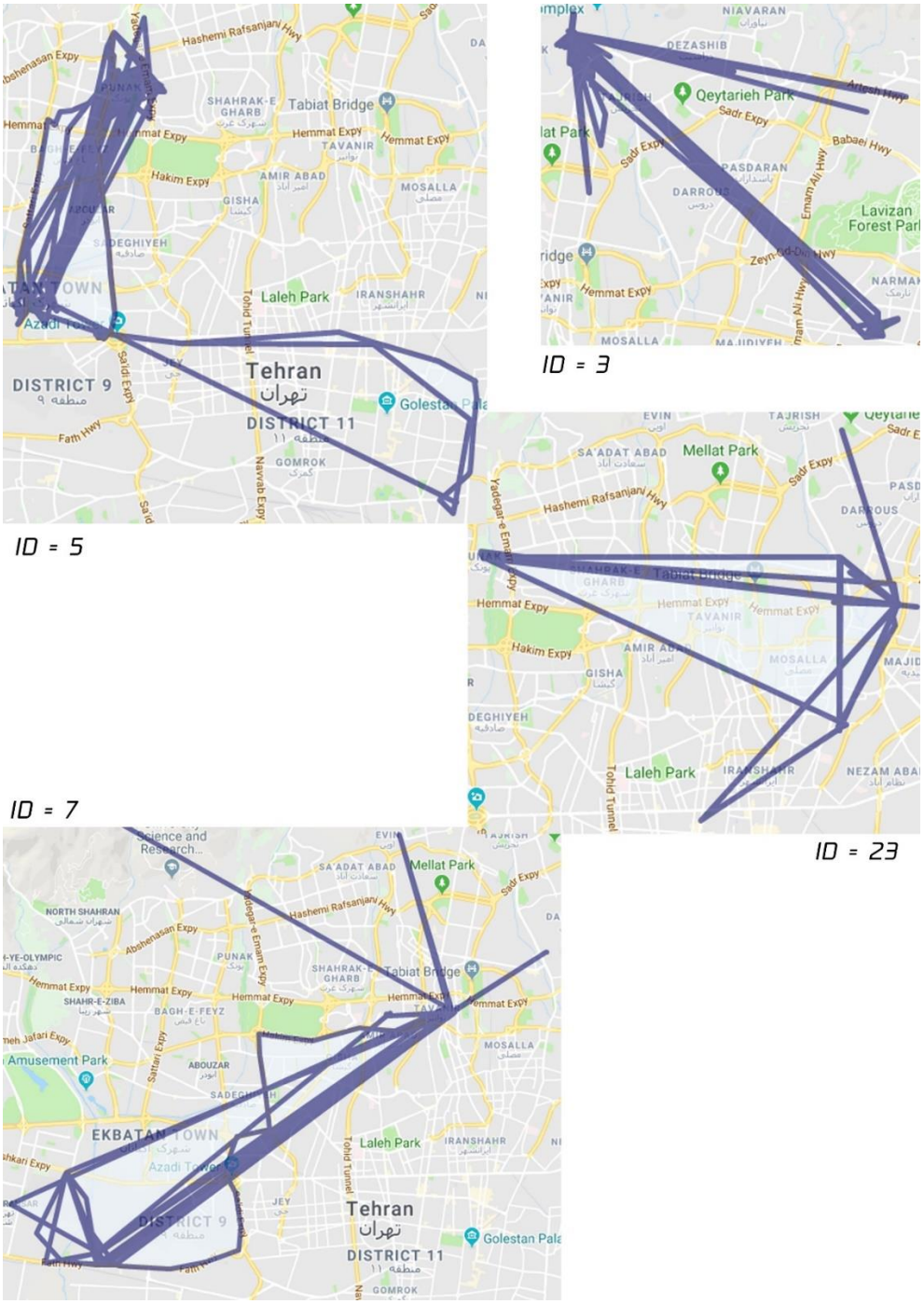
Slika 25. Raspodjela korištenja IM aplikacija

Od 35 sudionika studije iskoristive geografske podatke pružilo je 26 sudionika. Zanemarujući sudionika s ID-em 35, budući da njegovi podaci značajno odstupaju od prosjeka, uključujući prijeđen put (68.802,76 km) tijekom studije i broj dana u kojima postoje geografski podaci (108 dana) (Tablica 2)(Slika 26). Gledajući prosjeke sudionici su imali 40,5 dana geografskih podataka, 981,51 prijeđenih kilometara tijekom studije i 26,66 kilometara prijeđenih u danu. Korisnici su imali razne obrasce kretanja od onih koji su se kretali jednakim putanjama tijekom većine studije (Slika 27), do onih koji su imali jako veliki broj različitih putanja (Slika 28). Još treba napomenuti da je većina podataka prikupljena koristeći mobilnu mrežu a ne GNSS sustave, te su time i podaci lošije kvalitete odnosno manje preciznosti.

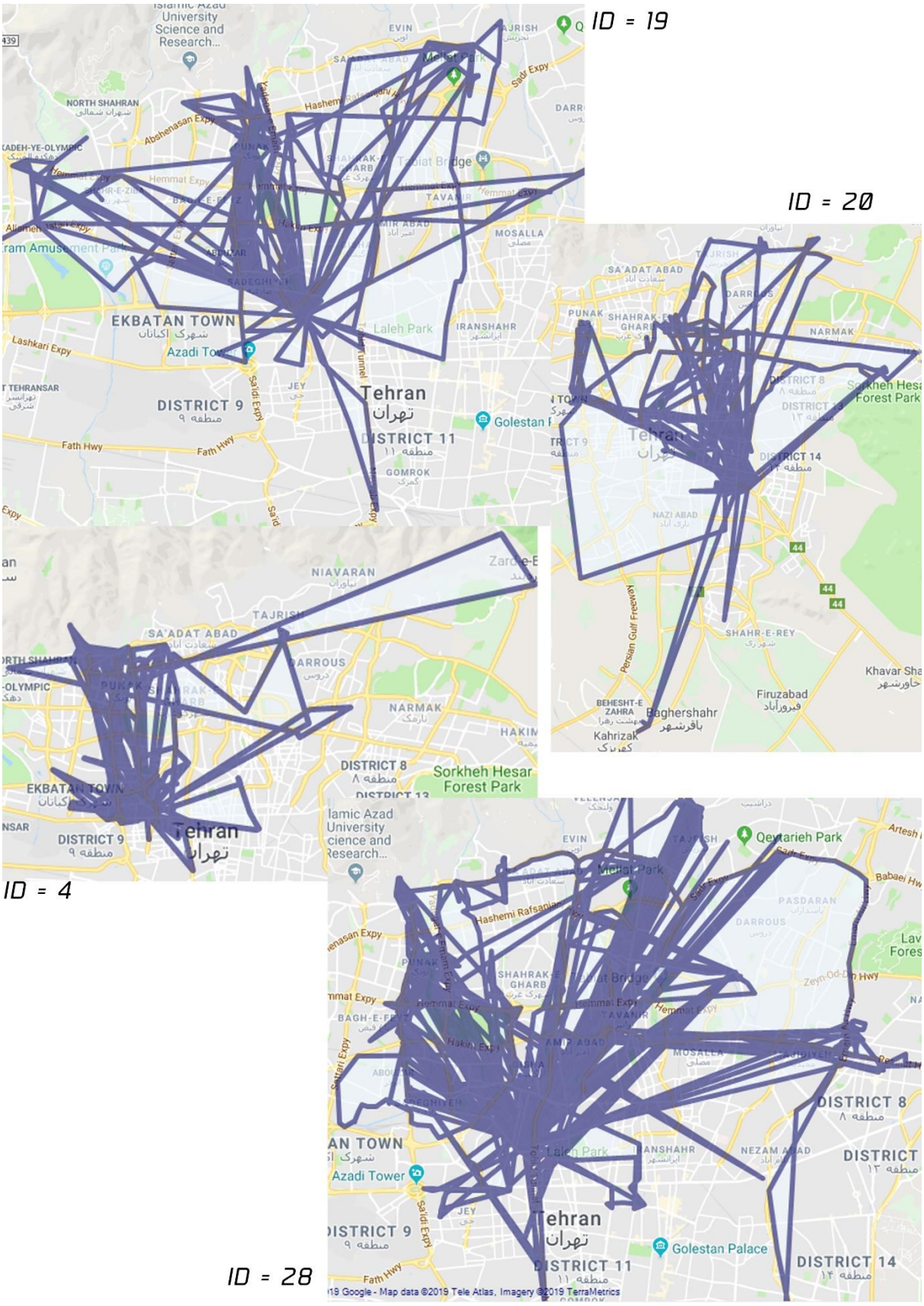
ID	Broj dana	Ukupno KM	Prosjek KM
1	11	321,32	29,21
3	40	837,15	20,93
4	30	866,67	28,89
5	56	601,2	10,74
7	51	2028,18	39,77
8	53	270,06	5,1
11	41	1706,47	41,62
12	49	427,75	8,73
13	3	367,06	122,35
14	15	278,53	18,57
16	51	90,46	1,77
18	64	1521,37	23,77
19	51	1056,17	20,71
20	46	1395,35	30,33
21	11	63,14	5,74
22	40	1069,92	26,75
23	30	865,06	28,84
24	43	1401,63	32,6
27	55	132,77	2,41
28	60	4502,14	75,04
29	56	2654,25	47,4
30	19	21,84	1,15
31	39	458,26	11,75
33	58	982,42	16,94
34	40	618,55	15,46
35	108	68802,76	637,06

Tablica 2. Prijeđen put korisnika tijekom studije

Slika 26. Podaci o kretanju sudionika s ID-em 35



Slika 27. Jednolične putanje kretanja



Slika 28. Raznovrsne putanje kretanja

6. Zaključak

Cilj ovoga rada bio je izrada grafičkog sučelja kojim će se olakšati prikaz i analiza velikoga broja podataka o korištenju pametnih telefona koji su u svojem izvornom obliku bili teško čitljivi.

Uočeno je nekoliko problema prilikom analize podataka. Uz jako mali uzorak sudionika studije zbog kojega pojedine devijacije određenih sudionika utječu na ukupne rezultate, veliki je problem stvarala nepotpunost prikupljenih podataka., npr. podaci za aktivnost, kod kojih je bilo kakva analiza teško izvediva zbog nedovoljnog broja podataka. Što se samih podataka tiče, odnosno izvlačenje istih iz dobivenih datoteka, pratili su ih određeni problemi zbog nekonzistentnosti podataka, te su neki podaci bili izgubljeni zbog toga.

Grafičkim prikazom podataka dobila se mogućnost predodžbe podataka koje bi gledajući same brojeve bilo nemoguće analizirati i predočiti. Samo dobiveni podaci bili su većinom su skladu s očekivanjima. Pojedine stavke su odstupale od početnih pretpostavki. Primjerice, broj *WiFi* mreža u okolini bio je izrazito velik te je odstupao od mojih očekivanja. Također sudionici su skoro pa ekskluzivno koristili IM aplikacije uz jako mali udio korištenja ostalih aplikacija, stvarajući predodžbu da pametne telefone koriste samo za komunikaciju. Iako treba imati na umu i vremenski kontekst u kojemu u podatci prikupljani, tako da tadašnja hardverska ograničenja uređaja (nemogućnost korištenja uređaja za nekakav kompleksniji rad i zabavu) mogu djelomično objasnite te podatke.

Uz veći broj sudionika te potpunije podatke rezultati analize mogli su biti znatno bolji, tj. bilo bi mogući izvući znatno više zaključaka o ponašanju i navikama korisnika.

Popis literature

[1] R. Rawassizadeh, E. Momeni i C. Dobbins, »Lesson Learned from Collecting Quantified Self Information via Mobile and Wearable Devices,« *Journal of Sensor and Actuator Networks,* 2015.

[2] R. Rawassizadeh, M. Tomitsch i K. a. T. Wac, »UbiqLog: a generic mobile phone based life-log,« *The Open University's repository of research publications,* 2013.

[3] T. Carić i T. Erdelić, »Baze podataka Predavanja«.

[4] »newtonsoft.com,« [Mrežno]. Available: https://www.newtonsoft.com/json/help/html/Introduction.htm. [Pokušaj pristupa 3.6.2019.].

[5] [Mrežno]. Available: https://docs.microsoft.com/en-us/dotnet/standard/base-types/custom-date-and-time-format-strings. [Pokušaj pristupa 8.8.2019.].

[6] [Mrežno]. Available: https://docs.microsoft.com/en-us/dotnet/api/system.data.sqlclient.sqlcommand.executescalar?view=netframework-4.8. [Pokušaj pristupa 9.8.2019.].

[7] »pcchip.hr,« [Mrežno]. Available: https://pcchip.hr/helpdesk/koja-je-razlika-izmedu-24-ghz-5-ghz-i-60-ghz-wi-fi-a/. [Pokušaj pristupa 15.08.2019.].

Popis kratica

ER - Entity Relationship (hrv. Odnos između entiteta)

JSON - JavaScript Object Notation

XML - EXtensible Markup Language

SMS - Short Message Service

HTC - High Tech Computer Corporation

HD - High definition

SSIS - SQL Server Integration Services

SSMS - SQL Server Management Studio

ID - Identity document / Identifier

SQL - Structured Query Language

WPA2 - Wireless Protected Access 2

WPA - Wireless Protected Access

WEP - Wired Equivalent Privacy

IM - Instant messaging

Popis slika

Slika 1. Prikaz grafičkog sučelja *UbiqLog* aplikacije 4
Slika 2. Dijagram entiteta 5
Slika 3. ER dijagram 6
Slika 4. Oblik izvornih datoteka 7
Slika 5. Postavke SSIS-a 10
Slika 6. Postavke prilikom nailaska na pogrešku 11
Slika 7. Postavke prilikom uvoza podataka 12
Slika 8. Uklanjanje redaka bez vrijednosti 13
Slika 9. Dijagram baze podataka 14
Slika 10. Izgled grafičkog sučelja sa svim kontrolama 15
Slika 11. Prikaz komponenti nakon označavanja checkboxa 16
Slika 12. Prikaz postavki pri kreiranju grafa za pozive 19
Slika 13. Broj jedinstvenih WiFi mreža po korisniku 25
Slika 14. Broj jedinstvenih Bluetooth uređaja po korisniku 26
Slika 15. Aktivnost – žiroskop 27
Slika 16. Vremenska raspodjela poruka 28
Slika 17. Raspodjela poruka tijekom tjedna 29

Slika 18. Vremenska raspodjela poziva 30
Slika 19. Vremenska raspodjela trajanja razgovora u sekundama 31
Slika 20. Raspodjela poziva tijekom tjedna 32
Slika 21. Raspodjela vremena trajanja razgovora tijekom tjedna 33
Slika 22. Broj otvaranja aplikacija 34
Slika 23. Prosječno vrijeme korištenja aplikacija tijekom studije u satima 35
Slika 24. Usporedba korištenja IM aplikacija te klasičnih poziva i poruka 36
Slika 25. Raspodjela korištenja IM aplikacija 37
Slika 26. Podaci o kretanju sudionika s ID-em 35 39
Slika 27. Jednolične putanje kretanja 39
Slika 28. Raznovrsne putanje kretanja 40

Popis priloga

Prilog 1. Deklariranja varijabli za učitavanje datoteka 7
Prilog 2. *While* petlja 8
Prilog 3. Klase za spremanje podataka 9
Prilog 4. Kreiranje i popunjavanje tablice Sudionik 13
Prilog 5. Postavljanje stranog i primarnog ključa 13
Prilog 6. Definiranje komponenti pri učitavanju forme 16
Prilog 7. Prikaz/sakrivanje komponenti 17
Prilog 8. Izvršavanje SQL naredbi i crtanje grafa unutar WiFi serije 18
Prilog 9. Dio kôda za prikaz poziva 21
Prilog 10. Kôd za iscrtavanje ruta i poligona 23

Popis tablica

Tablica 1. Tablica zadanih vrijednosti senzora [2] 2
Tablica 2. Prijeđen put korisnika tijekom studije 38

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.

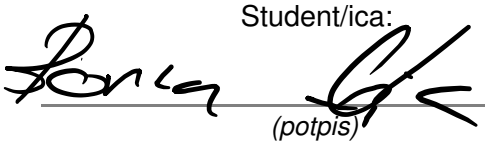
Izjavljujem također, kako nijedan dio rada nije iskorišten za bilo koji drugi rad u bilo kojoj drugoj visokoškolskoj, znanstvenoj ili obrazovnoj ustanovi.

Svojim potpisom potvrđujem i dajem suglasnost za javnu objavu završnog rada pod naslovom **Analiza i prikaz podataka o korištenju pametnog telefona**

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

U Zagrebu, 4.9.2019

Student/ica:



(potpis)