



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



ЗБОРНИК РАДОВА ФАКУЛТЕТА ТЕХНИЧКИХ НАУКА

Едиција: Техничке науке - зборници

Година: XXXIII

Број: 11/2018

Нови Сад

Едиција: „Техничке науке – Зборници“

Година: XXXIII

Свеска: 11

Издавач: Факултет техничких наука Нови Сад

Главни и одговорни уредник: проф. др Раде Дорословачки, декан Факултета техничких Наука у Новом Саду

Уредништво:

Проф. др Раде Дорословачки

Проф. др Драгиша Вилотић

Проф. др Срђан Колаковић

Проф. др Владимир Катић

Проф. др Драган Шешилија

Проф. др Миодраг Хаџистевић

Проф. др Растислав Шостаков

Доц. др Мирослав Кљајић

Доц. др Бојан Лалић

Доц. др Дејан Убавин

Проф. др Никола Јорговановић

Доц. др Борис Думнић

Проф. др Дарко Реба

Проф. др Ђорђе Лађиновић

Проф. др Драган Јовановић

Проф. др Мила Стојаковић

Проф. др Драган Спасић

Проф. др Драгољуб Новаковић

Редакција:

Проф. др Владимир Катић, главни
уредник

Проф. др Жељен Трповски, технички
уредник

Проф. др Драган Шешилија

Проф. др Драгољуб Новаковић

Др Иван Пинђер

Бисерка Милетић

Језичка редакција:

Бисерка Милетић, лектор

Софија Рацков, коректор

Марина Катић, преводилац

Издавачки савет:

Савет за библиотечку и издавачку делатност ФТН,
проф. др Радош Радивојевић, председник.

Штампа: ФТН – Графички центар ГРИД, Трг Доситеја Обрадовића 6, Нови Сад

CIP-Каталогизација у публикацији
Библиотека Матице српске, Нови Сад

378.9(497.113)(082)
62

ЗБОРНИК радова Факултета техничких наука / главни и одговорни уредник
Раде Дорословачки. – Год. 7, бр. 9 (1974)-1990/1991, бр.21/22 ; Год. 23, бр 1 (2008)-. – Нови Сад :
Факултет техничких наука, 1974-1991; 2008-. – илустр. ; 30 цм. –(Едиција: Техничке науке –
зборници)

Месечно

ISSN 0350-428X

COBISS.SR-ID 58627591

ПРЕДГОВОР

Поштовани читаоци,

Пред вама је једанаеста овогодишња свеска часописа „Зборник радова Факултета техничких наука“.

Часопис је покренут давне 1960. године, одмах по оснивању Машинског факултета у Новом Саду, као „Зборник радова Машинског факултета“, а први број је одштампан 1965. године. Након осам публикованих бројева у шест година, пратећи прерастање Машинског факултета у Факултет техничких наука, часопис мења назив у „Зборник радова Факултета техничких наука“ и 1974. године излази као број 9 (VII година). У том периоду у часопису се објављују научни и стручни радови, резултати истраживања професора, сарадника и студената ФТН-а, али и аутора ван ФТН-а, тако да часопис постаје значајно место презентације најновијих научних резултата и достигнућа. Од броја 17 (1986. год.), часопис почиње да излази искључиво на енглеском језику и добија поднаслов «Publications of the School of Engineering». Једна од последица нарастања материјалних проблема и несрећних догађаја на нашим просторима јесте и привремени прекид континуитета објављивања часописа двобројем/двогодишњаком 21/22, 1990/1991. год.

Друштво у коме живимо базирано је на знању. Оно претпоставља реорганизацију наставног процеса и увођење читавог низа нових струка, као и квалитетну организацију научног рада. Значајне промене у структури високог образовања, везане за имплементацију Болоњске декларације, усвајање нове и активне улоге студената у процесу образовања и њихово све шире укључивање у стручне и истраживачке пројекте, као и покретање нових мастер и докторских студија, доносе потребу да ови, веома значајни и вредни резултати, постану доступни академској и широј јавности. Оживљавање „Зборника радова Факултета техничких наука“, као јединственог форума за презентацију научних и стручних достигнућа, пре свега студената, обезбеђује услове за доступност ових резултата.

Због тога је Наставно-научно веће ФТН-а одлучило да, од новембра 2008. год. у облику пилот пројекта, а од фебруара 2009. год. као сталну активност, уведе презентацију најважнијих резултата свих мастер радова студената ФТН-а у облику кратког рада у „Зборнику радова Факултета техничких наука“.

Поред студената мастер студија, часопис је отворен и за студенте докторских студија, као и за прилоге аутора са ФТН или ван ФТН-а.

Зборник излази у два облика – електронском на веб сајту ФТН-а (www.ftn.uns.ac.rs) и штампаном, који је пред вама. Обе верзије публикују се сваки месец, у оквиру промоције дипломираних мастера.

У овом броју штампани су радови студената мастер студија, сада већ мастера, који су радове бранили у периоду од 05.06.2018. до 06.09.2018. год., а који се промовишу 19.12.2018. год. То су оригинални прилози студената са главним резултатима њихових мастер радова.

Известан број кандидата објавили су радове на некој од домаћих научних конференција или у неком од часописа. Њихови радови нису штампани у Зборнику радова.

Велик број дипломираних инжењера–мастера у овом периоду био је разлог што су радови поводом ове промоције подељени у две свеске.

У овој свесци, са редним бројем 11., објављени су радови из области:

- машинства,
- електротехнике и рачунарства,
- грађевинарства и
- саобраћаја.
-

У свесци са редним бројем 12. објављени су радови из области:

- графичког инжењерства и дизајна,
- архитектуре,
- инжењерског менаџмента,
- инжењерства заштите на раду и заштите животне средине,
- мехатронике,
- математике у техници,
- геодезије и геоматике,
- управљања ризиком од катастрофалних појава и пожара,
- инжењерства информационих система и
- биомедицинског инжењерства.

Уредништво се нада да ће и професори и сарадници ФТН-а и других институција наћи интерес да публикују своје резултате истраживања у облику регуларних радова у овом часопису. Ти радови ће бити објављивани на енглеском језику због пуне међународне видљивости и проходности презентованих резултата.

У плану је да часопис, својим редовним изласком и високим квалитетом, привуче пажњу и постане довољно препознатљив и цитиран да може да стане раме-уз-раме са водећим часописима и заслужи своје место на СЦИ листи, чиме ће значајно допринети да се оствари мото Факултета техничких наука:

„Високо место у друштву најбољих“

Уредништво

SADRŽAJ

STRANA

Radovi iz oblasti: Mašinstvo

1. Горан Јовичић, Јован Дорић,
АНАЛИЗА КАРАКТЕРИСТИКА ДИЗЕЛ МОТОРА КОРИШЋЕЊЕМ БИОДИЗЕЛ ГОРИВА 1881 – 1884
2. Miloš Stepanović, Mladomir Milutinović,
SIMULACIJA PROCESA INJEKCIONOG BRIZGANJA VIŠENAMENSKE POSUDE
ZAPREMINE 5 LITARA 1885 – 1888
3. Ana Šavija,
ZNAČAJ PRIMENE DIREKTIVA I STANDARDA EU ZA IZVOZNO ORIJENTISANA
PROIZVODNA PREDUZEĆA 1889 – 1892
4. Милош Пешић,
ГЛАВНА МЕРНО-РЕГУЛАЦИОНА СТАНИЦА СА ДВОСТЕПЕНОМ РЕДУКЦИЈОМ
ПРИТИСКА ГАСА 1893 – 1896
5. Никола Иланковић, Драган Живанић,
АНАЛИЗА УТИЦАЈНИХ ПАРАМЕТАРА НА ТАЧНОСТ МЕРЕЊА ТРАКАСТИМ
ДОЗАТОРОМ 1897 – 1900
6. Милош Голић,
ВИШЕНАМЕНСКИ ПРЕНОСНИК СНАГЕ ЕКСПЕРИМЕНТАЛНОГ ХИБРИДНО-
ЕЛЕКТРИЧНОГ ВОЗИЛА 1901 – 1903

Radovi iz oblasti: Elektrotehnika i računarstvo

1. Stanko Polić,
SOFTVER ZA POKRETANJE, ZAUSTAVLJANJE I OPORAVAK DISTRIBUIRANIH SERVISA 1904 – 1907
2. Nemanja Starčev,
PODRŠKA VIZUALIZACIJI STRUKTURE I BOJENJU KODA U VISUAL STUDIO CODE
EDITORU ZA JEZIKE BAZIRANE NA TEXTX ALATU 1908 – 1911
3. Dejan Krstić,
ELEKTRONSKI SISTEM MERENJA RAZDALJINE I GENERISANJA UPOZORENJA NA
PREPREKE KAO POMOĆ SLEPIM I SLABOVIDIM OSOBAMA 1912 – 1915
4. Dragan Žilić,
PROJEKTOVANJE INTEGRISANOG IZVORA REFERENTNOG NAPONA
U 0,35 μm AMS CMOS TEHNOLOGIJI 1916 – 1919
5. Robert Fruža, Vlado Porobić,
AKTIVNO OPTEREĆENJE ZA ISPITIVANJE DC NAPAJANJA..... 1920 – 1923
6. Miloš Stevanović, Dragan Milićević,
MOGUĆNOSTI SAVREMENIH MOTORNIH ZAŠTITNIH PREKIDAČA 1924 – 1927

7. Борко Цвејић, АПЛИКАТИВНО РЕШЕЊЕ КАО ИНТЕЛИГЕНТНИ САВЕТНИК ЗА ТРГОВИНУ АКЦИЈАМА НА БЕРЗИ	1928 – 1931
8. Uglješa Zelić, UTICAJ DISTRIBUTIVNIH GENERATORA NA REGULACIJU NAPONA U DISTRIBUTIVNOJ MREŽI	1932 – 1935
9. David Milivojević, DSL I GENERATOR KODA ZA AUTOMATIZACIJU RAZVOJA PHP APLIKACIJA	1936 – 1939
10. Milica Jovičić, Vladan Krsman, UTICAJ KONFIGURACIJE MERENJA NA TAČNOST ANALIZE LOŠIH PODATAKA U STATIČKOJ ESTIMACIJI STANJA	1940 – 1943
11. Milivoje Popovac, RAZVOJ VEB BAZIRANOG SISTEMA ZA PRORAČUN TOKOVA SNAGA DISTRIBUTIVNE MREŽE	1944 – 1947
12. Vladimir Popov, Marko Vekić, UPRAVLJANJE AKTIVNIM ISPRAVLJAČEM ZASNOVANIM NA P-REZONANTNIM REGULATORIMA	1948 – 1951
13. Jovo Vučetić, Marko Vekić, UPRAVLJANJE STATCOM-OM U SVRHU NADZORA NAPONA SABIRNICE U KOJOJ JE PRIKLJUČEN	1952 – 1955
14. Nada Ignjić, RAZVOJ NAMENSKOG JEZIKA ZA SPECIFIKACIJU MOBILNIH APLIKACIJA ZA DOMEN PRODAJE ODEĆE	1956 – 1959
15. Andrej Miletić, IZAZOVI U IMPLEMENTACIJI RAČUNARSKE IGRE	1960 – 1963
16. Aleksandra Sopka, UPOREDNA ANALIZA JAVASCRIPT BIBLIOTEKA I OKVIRA - REACT, ANGULAR, VUE	1964 – 1967
17. Lazar Bošnjaković, OPTIMALNO/AUTOMATSKO PUNJENJE I PRAŽNENJE SKLADIŠTA ELEKTRIČNE ENERGIJE	1968 – 1971
18. Aleksandar Radošević, Bojan Damjanović METODOLOŠKI OKVIR ZA PREDIKCIJU AKVATIČNE VEGETACIJE PRIMENOM VEŠTAČKIH NEURONSKIH MREŽA	1972 – 1975
19. Marko Vještica, POREĐENJE PERFORMANSI TEHNIKA INDEKSIRANJA U RELACIONIM BAZAMA PODATAKA	1976 – 1979
20. Teodora Mitrović, PRIMENA PAYPAL SISTEMA ZA INTERNET KUPOVINU POLISA OSIGURANJA	1980 – 1983
21. Mina Medić, IZRAČUNAVANJE CENA POLISA OSIGURANJA POMOĆU SISTEMA BAZIRANIH NA PRAVILIMA	1984 – 1987
22. Katarina Preradov, KEYCLOAK SISTEM ZA UPRAVLJANJE IDENTITETOM I KONTROLOM PRISTUPA	1988 – 1991
23. Tijana Savčić, Vladimir Katić, PREGLED SOFTVERSKIH ALATA ZA SIMULACIJU RADA FOTONAPONSKIH SISTEMA	1992 – 1994
24. Радован Туровић, ПРОНАЛАЗЕЊЕ РЕФЕРЕНТНИХ ТАЧАКА НА РЕНДГЕНСКИМ СНИМЦИМА КАРЛИЦЕ ПРИМЕНОМ КОНВОЛУЦИОНИХ НЕУРОНСКИХ МРЕЖА	1995 – 1998
25. Miloš Kajkut, AUTOMATSKO TESTIRANJE CVC DMS FUNKCIJE IMPELEMENTIRANE U ADMS SOFTVER	1999 – 2002
26. Jelena Janković, HIBRIDNA PARALELIZACIJA FILTRIRANJA 2D SLIKA	2003 – 2007

27.	Nenad Milić, GPU IMPLEMENTACIJA KONVOLUCIJE MREŽE ZA DETEKCIJU OBJEKATA U KONTINUALNIM MEDIJIMA	2008 – 2011
28.	Mirko Mikać, PROGRAMSKI JEZIK OBJECTIVE-C	2012 – 2015
29.	Stefan Varajić, OBJEKTNO-ORIJENTISANA PARADIGMA U PROGRAMSKOM JEZIKU PYTHON	2016 – 2019
30.	Olja Miljatović, Zorica Suvajdžin Rakić, IMPLEMENTACIJA APLIKACIJE ZA GENERISANJE TESTOVA ZA PROGRAM OTISAK UZ OSLONAC NA SPRINGBOOT I REACTJS	2020 – 2023
31.	Lazar Starčević, MODEL ZA SIMULACIJU RADA TOPLOTNE PODSTANICE SA INDIREKTNOM RAZMENOM TOPLOTE REALIZOVAN U PROGRAMSKOM PAKETU LABVIEW	2024 – 2027
32.	Čaba Šagi, INTEGRACIJA SCADA UREĐAJA	2028 – 2031
33.	Marko AntoniĆ, JEDNO REŠENJE SOFTVERKE ARHITEKTURE PRIJEMNIKA AUDIO I VIDEO SADRŽAJA ZASNOVANO NA DIGITALNOM SIGNAL PROCESORU	2032 – 2035
34.	Maksim Lalić, PRIMENA REGRESIONIH MODELA U PREDVIĐANJU ISHODA GLASANJA NA PESMI EVROVIJJE	2036 – 2039
35.	Dušan Radisavljević, NAMENSKI JEZIK I OKRUŽENJE ZA MODELOVANJE I GENERISANJE VIDEO IGARA SA NASUMIČNO GENERISANIM NIVOIMA	2040 – 2043
36.	Andrijana Stefanović, AGREGACIJA VREMENSKIH SERIJA U OBLASTI ELEKTROENERGETSKIH SISTEMA NA RAZLIČITIM TIPOVIMA OLAP SERVERA	2044 – 2047
37.	Filip Đorđević, 1POL PLOTER – VIZUELIZATOR JEDNOPOLNIH ŠEMA ELEKTRODISTRIBUTIVNE MREŽE	2048 – 2051
38.	Немања Жерајић, Немања Недић, АРХИТЕКТУРА ГЕНЕРИЧКОГ СЕРВИСА ЗА МАНИПУЛАЦИЈУ МОДЕЛИМА ПОДАТАКА ...	2052 – 2055
39.	Бранко Савић, РАЗВОЈ АПЛИКАЦИЈЕ ЗА ДЕТЕКЦИЈУ И АНАЛИЗУ ДУПЛИРАНОГ КОДА УПОТРЕБОМ ЈСЦПД АЛАТА	2056 – 2059
40.	Radislav Zekanović, МЕХАНИЗМИ ПРАЋЕЊА ПОТРОШЊЕ RESURSA PO ORGANIZACIJI PRI MIGRACIJI NA MULTI-TENANT SAAS	2060 – 2063
41.	Бојан Торбица, РАЗВОЈ ПЕРСОНАЛНОГ СИСТЕМА ЗА ПРАЋЕЊЕ ЖИВОТНИХ ПАРАМЕТАРА	2064 – 2067
42.	Miloš Marković, RELEJNA ZAŠTITA I PREDIKCIJA INCIDENATA U SISTEMU ZA UPRAVLJANJE OTKAZIMA	2068 – 2071

Radovi iz oblasti: Građevinarstvo

1.	Данијел Малешевић, Матија Стипић, ХИДРАУЛИЧКА АНАЛИЗА ОДВОЂЕЊА УПОТРЕБЉЕНИХ ВОДА НАСЕЉА ПЕЋИНЦИ, СУБОТИШТЕ И ПРХОВО СА ТЕХНО-ЕКОНОМСКИМ ЕЛЕМЕНТИМА	2072 – 2075
2.	Ивана Богнар, Небојша Радовић, ОЗАКОЊЕЊЕ ОБЈЕКТА У РЕПУБЛИЦИ СРБИЈИ – ПРИМЕРИ ОЗАКОЊЕЊА ОБЈЕКТА НА ПУТНОЈ МРЕЖИ	2076 – 2079

3. Милош Нешковић, МЕТОДОЛОШКЕ ОСНОВЕ ПЛАНИРАЊА И ПРОЈЕКТОВАЊА ОБИЛАЗНИЦА – ПРИМЕР ОБИЛАЗНИЦЕ ОКО КАЊИЖЕ	2080 – 2083
4. Siniša Sadžakov, HIDRAULIČKA ANALIZA VODOSNABDEVANJA OPŠTINE SRBOBRAN	2084 – 2087
5. Урош Ивковић, Небојша Радовић, РАЧУНАРСКИ ПОДРЖАНО (CAD) ПРОЈЕКТОВАЊЕ САОБРАЋАЈНИЦА- ПРИМЕР ПРИМЕНЕ СОФТВЕРА PLATEIA НА ПРОЈЕКТУ ПУТА ГРГУРЕВЦИ – БЕШЕНОВО	2088 – 2091
6. Milena Senjak, Vladimir Mučenski, ANALIZA KARAKTERISTIKA POVREDA NA RADU NA GRADILIŠTIMA REPUBLIKE SRBIJE	2092 – 2095
7. Nataša Rudakijević, PROCENA STANJA, DOGRADNJA I ENERGETSKA SANACIJA PORODIČNE STAMBENE ZGRADE	2096 – 2099
8. Ivana Gvero, PROJEKAT ARMIRANOBETONSKE VIŠESPRATNE STAMBENO-POSLOVNE ZGRADE PREMA EVROKODU I POREĐENJE EVROPSKIH I DOMAĆIH PROPISA	2100 – 2103
9. Sanja Lazić, PROJEKTOVANJE ČVORA GREDA – STUB U ARMIRANOBETONSKIM KONSTRUKCIJAMA USLED SEIZMIČKOG DEJSTVA	2104 – 2107
10. Bojan Milivojević, PROCENA STANJA I SANACIJA KONSTRUKCIJE MEHANIZOVANOG PODNOG SKLADIŠTA ZA ŽITARICE	2108 – 2111

Radovi iz oblasti: Saobraćaj

1. Nataša Čačić, SISTEMI ZAŠTITE PLATNIH KARTICA	2112 – 2115
2. Silard Istvan, Dejan Nemec, DISTRIBUIRANO REŠAVANJE SISTEMA LINEARNIH JEDNAČINA KORIŠĆENJEM RASPBERRY PI UREĐAJA PREKO AD-HOC MREŽE	2116 – 2119

АНАЛИЗА КАРАКТЕРИСТИКА ДИЗЕЛ МОТОРА КОРИШЋЕЊЕМ БИОДИЗЕЛ ГОРИВА

ANALYSIS OF CHARACTERISTICS DIESEL ENGINES USING BIODIESEL FUEL

Горан Јовичић, Јован Дорић, *Факултет техничких наука, Нови Сад*

Област – МАШИНСТВО

Кратак садржај – У овом раду задатак је био да се уради испитивање дизел мотора погоњеног дизел горивом и биодизел горивом. Циљ задатка је да се испитају следећи параметри и величине: број обртаја мотора, момент мотора, притисак у цилиндру мотора, емисија издувних гасова, температуре. Па на основу испитаних величина израчунати основне параметре мотора СУС.

Abstract – In this thesis task is to do test of diesel engine using diesel and biodiesel feul. The goal is examine next parameters and values: revolutions per minute, moment of engine, pressure in the cylinder of the engine, emission of exhaust gases, temperature. Then on the basis of the examined sizes, calculate the parameters internal combustion engine ICE

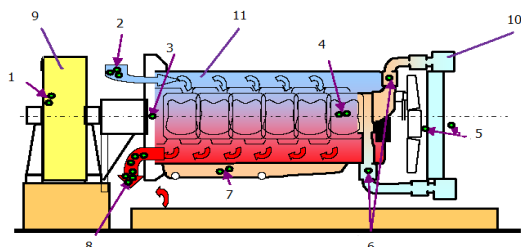
Кључне речи: индицирани притисак, ефективна снага

Кључне речи: indicated pressure, effective power

1. УВОД

Потрошња енергије у свету бележи стални раст упркос многим мерама које се предузимају ради рационализације потрошње енергије у свим сферама људске делатности. Потрошња у свету данас износи више од 9.500 [mTEN] (милиона тона еквивалента нафте)[1]. Залихе фосилних горива нестају великом брзином и у року једне до две деценије већине земаља биће приморане да користе обновљиве изворе енергије[1-3].

2. АНАЛИЗА КАРАКТЕРИСТИКА ДИЗЕЛ МОТОРА КОРИШЋЕЊЕМ БИОДИЗЕЛ ГОРИВА



Слика 1. Шематски приказ уређаја за испитивање мотора

Оптерећење брзине обртања (1), масени проток, температура, притисак улазног ваздуха (2), ТДЦ (3), температура горива, притисак убризгавања, ход игле, притисак у цилиндру (4), температура ваздуха на улазу и излазу (5), температура на улазу и излазу из мотора (6), температура и притисак уља (7), температура, притисак, емисија честица издувних гасова (O_2, CO, HC, NO_x) (8), динамометар (9), хладњак воде (10), мотор (11).

2.1. Подаци о мотору и систему на коме су вршена испитивања

Табела 1.

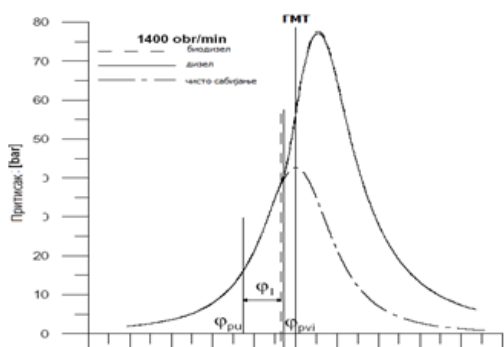
Модел мотора	MAN D 0826 LOH15
Тип мотора	Четверотактни, шестоцилиндрични, линијски, водом хлађени
Радна запремина	11 413 [cm ³]
Степен компресије	18
Пречник и ход клипа	108[mm] x 155 [mm]
Максимална снага	162 kW kod 2200 rpm
Начин убризгавања горива	директно убризгавање, (МАН) М-поступак сагоревања
Пумпа за убризгавање горива	Bosch PES 6A 95D 410 LS 2542
Клип пумпе	9.5 mm x 8 mm
Цев за гориво	1024 mm x 1.8 mm
Сапница (број и пречник отвора)	1 x 0.68 mm
Највећи ход игле	0.3 mm
Притисак отварања игле	175 [bar]

Осим карактеристика мотора који су приказани у табели 1. извршено је снимање и осталих релевантних карактеристика као што су: промена притиска у цилиндру мотора, максимални притисак у мотору, проток усисаног ваздуха, проток воде за хлађење, систем издувних гасова, потрошња горива и сл. Снимање индикаторских дијаграма извршено је задржавањем угла предубризгавања, за случај свих горива. Пример снимљеног дијаграма притиска у мотору при коришћењу дизела и биодизела приказан је на слици 2. На истој слици приказано је раздобље

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био др Јован Дорић, ван. проф.

притајеног сагоревања (φ_1), угао предубризгавања (φ_{pu}), почетак видљивог сагоревања (φ_{pvs}). На слици 2. приказан је дијаграм индицираног притиска за број обртаја који одговара максималном моменту при пуном оптерећењу мотора. Разлике притисака у два екстремна случаја, коришћење биодизела, нису велике, а исто тако могу се приметити мале, али одређене разлике у угаоном интервалу трајања притајеног сагоревања. Код биодизела угаони интервал притајеног сагоревања је нешто мањи, што се може повезати са нешто већим цетанским бројем биодизела у односу на дизел, као и већом вискозношћу биодизела те се самим тим и процес убризгавања горива одвија другачије у односу на дизел гориво.



Слика 2. Индицирани притисак мотора СУС коришћењем биодизел и дизел горива на режиму максималног момента

Повећањем броја обртаја мотора и приближавању режиму максималне снаге, разлике у индицираним притисцима у цилиндру мотора између случајева коришћења чистог биодизела, односно дизела су све мање.

2.2 Показатељи процеса сагоревања

Оптималан рад мотора првенствено зависи од убризгавања горива, његовог мешања са ваздухом и почетка паљења смеше, као и тока сагоревања[2]. Основни параметри сагоревања у мотору су : максимални притисак убризгавања p_{max} , максимална температура. Максимални притисак сагоревања као и почетак видљивог сагоревања одређују се на основу снимљеног дијаграма притиска и дијаграма притиска компресије. Раздобље трајања сагоревања не може се одредити директно на основу резултата мерења. Оно се дефинише преко карактеристика сагоревања које се добија обрадом индикаторског дијаграма притиска. Брзина ослобађања топлоте може се израчунати коришћењем следећих формула[4] :

$$\frac{dQ_{akt}}{d\varphi} = \frac{\frac{dQ_c}{d\varphi} + p \frac{dV}{d\varphi} + \frac{1}{R} \frac{\delta u}{\delta T} \frac{dV}{d\varphi} + V p \frac{dV}{d\varphi} - m T \frac{\delta R}{\delta p} \frac{dp}{d\varphi} + m \frac{\delta u}{\delta p} \frac{dp}{d\varphi}}{1 + \left[\frac{\delta u}{\delta T} \frac{T}{1 + \frac{T \delta R}{R \delta T}} u \right] \frac{1}{H_g} c_\alpha m \left[\frac{\delta u}{\delta \alpha} - \frac{\delta u}{\delta \alpha} \frac{\delta R T}{R} - \frac{1}{1 + \frac{T \delta R}{R \delta T}} \right]} \quad (1)$$

где је :

Q_c - губитак топлоте

u - специфична унутрашња енергија

R - гасна константа

T - температура

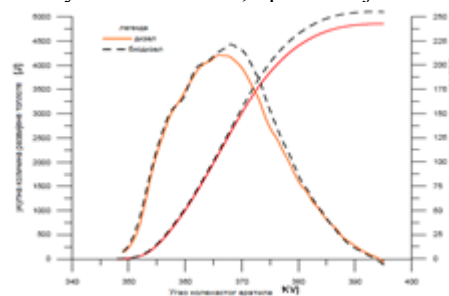
H_g - доња топлотна вредност

α - коефицијент вишка ваздуха

m - маса

p - притисак

С обзиром на карактер тока снимљеног притиска у мотору, било је потребно извршити његово изравњавање у зони појачаних осцилација, како би се добиле просечне вредности Q_{akt} и dQ_{akt}/d . Све остале вредности као што су: тренутна запремина у цилиндру $V(\varphi)$, маса радне смеше $m(\varphi)$, гасна константа $R(\varphi)$ итд., дефинисани су променом угла обртања коленастог вратила (φ) поштујући геометријске величине клипног механизма, састав радне смеше у цилиндру, састав гасова сагоревања у цилиндру и сл. Пример израчунатих карактеристичних величина, за оба разматрана горива при режиму називне снаге, приказан је на слици 3.



Слика 3. Карактеристика ослобађања топлоте и прираст укупно ослобођене топлоте

3. РАДНИ ПАРАМЕТРИ МОТОРА

Радни параметри мотора се још називају и интегрални параметри јер карактеришу процес мотора у целини. Они квантификовано (бројно) изражавају квалитет процеса рада мотора, на пример у погледу радног учинка, ефикасности, економичности и сл., и као такви служе за упоређење са конкурентним агрегатима, процену могућности апликације у појединим областима итд.

Могу се сврстати у три групе: 1. индицирани параметри који карактеришу оно што мотор развија у радном простору (цилиндру), 2. Ефективни параметри који карактеришу оно што мотор даје на излазном вратилу, 3. комплексни радни параметри који обично карактеришу форсираност мотора или неку другу особеност

3.1 Индикаторски параметри мотора

Показују учинак и економичност унутар радног простора, добијају се снимањем процеса рада, (индицирањем), они показују оно што даје стварни радни процес у цилиндру мотора.

Специфични индицирани рад циклуса или специфични индицирани притисак, сл.4. представља мјерило ефикасности стварног радног циклуса у цилиндру мотора. Показује колико се рада може добити по јединици радне запремине. Графички приказ специфичног радног притиска налази се на слици 4.

Индикаторски рад циклуса (рад развијен у једном радном циклусу)

$$W_i = \int p dV \text{ [J]} \quad (2)$$

Специфични индицирани рад или специфични индицирани притисак

$$p_i = \frac{W_i}{V_h} \left[\frac{\text{J}}{\text{m}^3} \right] \quad (3)$$

Индикаторска снага мотора

$$P_i = W_i \frac{2n}{\tau} = p_i V_h \frac{2n}{\tau} \text{ [W]} \quad (4)$$

Индикаторска снага мотора код вишцилиндричног мотора

$$P_i = z p_i V_h \frac{2n}{\tau} = p_i V_{hu} \frac{2n}{\tau} \quad (5)$$

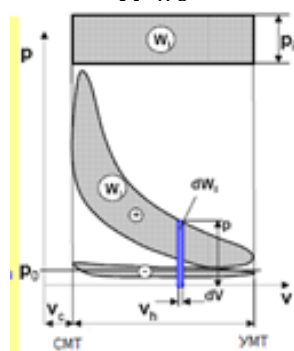
где је:

z – број цилиндара

V_{hu} – Укупна запремина целог мотора

Специфична индикаторска потрошња горива

$$g_i = \frac{m_g}{P_i} \left[\frac{\text{kg}}{\text{W s}} \right]$$



Слика 4. Средњи индицирани притисак

Потрошња горива у јединици времена сведена је на јединицу индициране снаге

$$g_i = \frac{G_h}{P_i} 10^3 \left[\frac{\text{g}}{\text{kWh}} \right] \quad (7)$$

где је :

G_h – проток горива $\left[\frac{\text{kg}}{\text{h}} \right]$

За дизел моторе $g_i = 160 - 240$

Индикаторски степен искориштења

Дефинисан је као однос енергије у облику механичког рада и уложене хемијске енергије горива.

Представља мерило економичности стварног циклуса у цилиндру мотора. Ако се посматра један радни циклус :

$$\eta_i = \frac{W_i}{Q_{gc}} \quad (8)$$

$$Q_{gc} = \frac{m_g H_d}{z \frac{2n}{\tau}} \left[\frac{\text{J}}{\text{cikl.}} \right] \quad (9)$$

где је :

m_g – укупни проток горива

H_d – доња топлотна моћ горива

z – број цилиндара

$$W_i = \frac{P_i}{z \frac{2n}{\tau}} \quad (10)$$

$$\eta_i = \frac{P_i}{m_g H_d} \quad (11)$$

$$\eta_i = \frac{1}{g_i H_d} \quad (12)$$

Степен добротe радног циклуса

Максимално што се може добити уколико мотор ради по одређеном циклусу пружа идеални термодинамички циклус. Што се стварни циклус више приближи овом „теоријском циклусу“ то ће и његов квалитет бити бољи. Ради процене колико се стварни циклус приближио теоријском дефинише се степен добротe радног циклуса η_d као однос индицираног рада стварног циклуса W_i и развијеног рада термодинамичког циклуса W_t

$$\eta_d = \frac{W_i}{W_t} \quad (13)$$

Како је по дефиницији $W_t = \eta_t Q_1$ и $W_i = \eta_i Q_{gc}$ и како количине доведене топлоте можемо сматрати еквивалентним

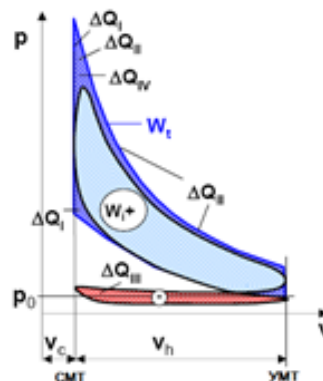
$$Q_1 = Q_{gc} \eta_d = \frac{\eta_i}{\eta_t}$$

(14)

Код идеалног термодинамичког циклуса је једини губитак еквивалентна количина топлоте Q_2 тако да је:

$$\eta_t = \frac{W_t}{Q_1} = \frac{Q_1 - Q_2}{Q_1} = 1 - \frac{Q_2}{Q_1} \quad (15)$$

Код стварног циклуса се осим Q_2 , што је у ствари топлота еквивалентна разлици енталпије издувних гасова који напуштају цилиндар и енталпије свржег пуњења, јавља се још читав низ губитака ΔQ_g . Због тога је индицирани рад стварног циклуса мањи што је приказано на p-v дијаграму, слика 5.



Слика 5. p - v дијаграм

Стварни термодинамички циклус

$$\eta_t = 1 - \frac{Q_2 + \Delta Q_g}{Q_1} \quad (16)$$

$$\Delta Q_g = Q_a + Q_b + Q_c + Q_d + Q_e \quad (17)$$

Q_a – Топлота еквивалентна губицима због реалности процеса сагоревања. Процес сагоревања код стварног циклуса почиње пре СМТ, а завршава после СМТ. И поред свих мера да се сагоревање убрза код неповољних услова догоревање се може знатно протезати на линију ширења.

Q_b – Губици топлоте због хлађења делова како би се њихове температуре одржале у дозвољеним границама. Прелаз топлоте на зидове радног простора је интензивнији што је: већи коефицијент прелаза топлоте, већа површина размене топлоте, већа разлика између температура гаса и зидова; зато су топлотни губици највећи за време сагоревања и ширења

Q_c – губици топлоте еквивалентни раду утрошеном на измену радне материје (пумпни рад)

Q_d – губици топлоте због непотпуног и недовршеног сагоревања

Q_e – остали топлотни и други губици које је тешко сагледати, нпр. зрачење мотора на околину, непотпуна заптивеност радног простора итд.

Поред наведених губитака на степен добре стварног циклуса утичу и други параметри као што су: регулација и услови рада.

4. ПРИМЕР

Прорачун индицираних параметара

Индициран је притисак у цилиндру на мотору и мерен је положај коленастог вратила.

Одредити: индикаторска рад, индикаторска снагу, средњи индикаторски притисак процеса, индикаторски степен искориштења

Обрада резултата добијених: мерењем силе кочења, брзине обртања и потрошње горива, индицирањем на мотору за радну тачку при брзини 1200 [s⁻¹] и силом кочења 102 [N]

$$\text{Снага} : P = \frac{Fn}{1360} \text{ [kW]}$$

$$\text{Фактор корекције} : k = \frac{p}{p_0} \sqrt{\frac{T}{T_0}}$$

$$\text{Коригована снага} : P_{kor} = kP \text{ [kW]}$$

$$\text{Обртни момент} : M = 9550 \frac{P_{kor}}{n} \text{ [Nm]}$$

$$\text{Укупни радни волумен мотора} : \Sigma V_h = \frac{D^2}{4} \pi S z \text{ [cm}^3\text{]}$$

$$\text{Средњи ефективни притисак} : p_e = \frac{1200 P_{kor}}{n \Sigma V_h} \text{ [bar]}$$

$$\text{Специфична потрошња горива} : g = \frac{3600 V_0 \rho}{\tau P_{kor}} \text{ [g/kWh]}$$

$$\text{Потрошња горива по процесу} : \beta = \frac{V}{\frac{n}{2 \cdot 60} \tau} \text{ [mm}^3\text{/proc.]}$$

Табела бр. 2

n [min ⁻¹]	F [kp]	τ [s]	P [kW]	k	P_{kor} [kW]	M [Nm]	ΣV_h [ccm]	p_e [bar]	g [g/kWh]	β [mm ³ /proc]
2300	41.5	18	70.18	1.10	76.97	319.58	3.92	10.24	216.88	72.87
2200	44.5	18	71.99	1.10	78.94	342.68	3.92	10.98	212.64	76.61
2000	46.2	17	87.94	1.10	74.51	355.77	3.92	11.40	234.52	87.72
1800	48.4	24	64.06	1.10	70.25	372.71	3.92	11.94	179.47	70.32
1600	47	23	55.29	1.10	60.64	361.93	3.92	11.60	212.40	80.82
1500	47	25	51.84	1.10	56.85	361.93	3.92	11.60	214.54	81.63
1400	46.7	26	48.07	1.10	52.72	359.62	3.92	11.52	218.83	82.74
1200	44.6	30	39.35	1.10	43.16	343.45	3.92	11.00	233.91	84.46
1000	39.6	36	29.12	1.10	31.93	304.95	3.92	9.77	262.12	84.03

Резултати из претходних операција су у табели бр.2

Индикаторски рад :

$$W_i = f p dV = \Sigma p dV = 217,945 \text{ [J]}$$

Индикаторска снага :

$$P_i = W_i \cdot z \frac{n}{30000 \tau} = 217,945 \cdot 6 \frac{1200}{30000 \cdot 4} = 13,077 \text{ [kW]}$$

Средњи индикаторски притисак :

$$p_{sr,i} = \frac{W_i}{V_s} = \frac{217,945}{0,00114511 \cdot 100000} = 1,903267 \text{ [bar]}$$

Доведена енергија горивом:

$$Q_{gc} = \frac{m_g H_d \tau}{z \cdot n \cdot 2 \cdot 60} = \frac{3,47 \cdot 42700000 \cdot 4}{6 \cdot 1200 \cdot 60 \cdot 2} = 685,97 \text{ [J]}$$

Индикаторски степен искориштења:

$$\eta_i = \frac{W_i}{Q_{dov}} = \frac{217,945}{685,97} = 0,317719$$

5. ЗАКЉУЧАК

Уз мање преправке на мотору, као што су замена неметалних делова деловима од адекватног материјала, подешавања убризгавања, подешавања односа смеше гориво – ваздух и др. Биодизел се може користити за погонско гориво. Проучавајући карактеристике рада дизел мотора погоњеног дизел и биодизел горивом видећемо да нема великих разлика у раду мотора. Радни параметри истог мотора погоњеним са ове две врсте горива разликују се при мањим бројевима обртаја, са повећањем броја обртаја те разлике се смањују. До различитих особина при истим условима рада, долази због разлике у карактеристикама горива. Најважнија од тих карактеристика је топлотна моћ горива, биодизел има нешто нижу топлотну моћ за око 10 [%] у односу на дизел па због тога остварује мању сангу мотора. Део те снаге надокнађује тиме што има већу вискозност па приликом убризгавања усисава се већа количина горива тако да је снага мотора који за гориво користи биодизел 5 – 7 [%] мања.

6. ЛИТЕРАТУРА

- [1] Џевад Бибић, Алеш Хриберник, Иван Филиповић, Бреда Кегл, Утјецај алтернативних горива на показатеље изгарања код дизелових мотора – научни рад, 2009
- [2] Желимир Добовишек, Блаж Вајда, Станислав пехан, Бреда Кегл, Утјецај својстава горива на значајке мотора и триболошки параметри -*научни рад, 2009
- [3] Ивица Стаменковић, Ивана Банковић, Оливера Стаменковић, Влада Вељковић, Континуални поступци добијања биодизела – прегледни рад, 2009
- [4] Краљевић, Д., Емерт, Р., Хеффер, Г., Плашчак, И., Анализа карактеристика дизел мотора кориштењем дизелског и биодизелског горива – научни рад, 2009

Kratka biografija:



Горан Јовичић је рођен у Ханау на Мајни, Њемачка 1973 године Дипломски рад на Саобраћајном факултету у Добоју из области Саобраћаја–Транспортно инжињерство одбранио је 2011 год.



Јован Дорич је рођен у Новом Саду 1983 год. Дипломирао, магистрирао и докторирао на Факултету техничких наука, гдје је тренутно запослен у звању ванредног професора.

**SIMULACIJA PROCESA INJEKCIONOG BRIZGANJA VIŠENAMENSKE POSUDE
ZAPREMINE 5 LITARA****NUMERICAL SIMULATION OF INJECTION MOLDING OF MULTI-PURPOSE 5L
CONTAINER**

Miloš Stepanović, Mladomir Milutinović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj – Cilj ovog rada jeste analiza procesa injektionog presovanja primenom numeričkih simulacija. Studija slučaja izvedena je na primeru injektionog brizganja višenamenska posuda zapremine 5 litara korišćenjem programskog paketa Moldex3D. Na početku rada prvo su date opšte karakteristike procesa injektionog presovanja, a zatim prikaz same posude. U skladu sa strukturom Moldex3D softvera i dinamikom procesa numeričke simulacije sprovedene su za četiri karakteristične faze procesa injektionog presovanja na bazi prethodno definisanih parametara procesa.

Abstract – The goal of this paper is to analyze an injection molding process using numerical simulation. Case study is performed for injection molding of a multi-purpose container of 5 liters by employing Moldex3D software. At the beginning of the paper, the general characteristics of both injection molding process and container itself are given first. In accordance to the structure of Moldex3D software and the circle of injection molding process numerical simulations are carried out for four different process stages based on previously chosen parameters.

Cljučne reči: Injekciono presovanje, numerička simulacija, posuda.

1. UVOD

Injekciono presovanje (brizganje) je najvažniji ciklični postupak prerade polimera, a prema dostignutoj tehnologiji i najsavršeniji. Postupak spada u primarnu preradu polimera (praoblikovanje), jer se oblik otpreska koji se dobija posle presovanja izrađuje od materijala koji nema određenu formu nego je u vidu komadića, granula, praha itd.

Proces se izvodi tako što se rastopljena masa određene viskoznosti iz injektione jedinice ubrizgava u temperiran kalup, nakon čega obradak u kalupu očvršćava hlađenjem. Prema prof. Čatiću [1] tehnologijom injektionim presovanjem može se vršiti oblikovanje: termoplasta, termoseta i elastomera.

Osnovne prednosti izrade plastičnih proizvoda ovom tehnologijom su:

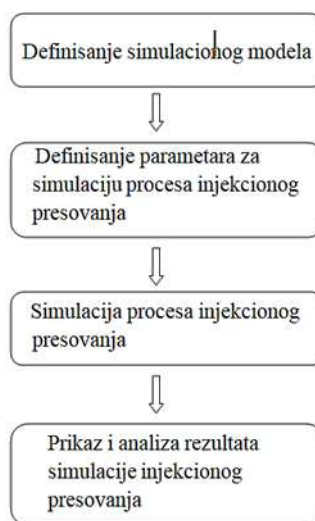
- veoma velika raznolikost delova,
- visoka ponovljivost dobijenih proizvoda,
- estetski proizvod u željenim bojama,
- kratko vreme ciklusa proizvodnje,
- ušteda materijala,
- dobijanje krajnje oblikovanog proizvoda, bez dodatnih obrada,
- mogućnost automatizacije tehnologije.

U današnje vreme, tržište proizvoda dobijenih injektionim presovanjem od otpresaka zahtevaju sve kompleksniju geometriju, kraće vreme isporuke, visoki kvalitet uz što jeftinije troškove proizvodnje itd. Kao odgovor na sve strožije zahteve tržišta javlja se područje upotrebe računara za simulaciju procesa injektionog presovanja (CEA) [2].

Većina softvera specijalizovanih za simulaciju injektionog presovanja se baziraju na korišćenju metode konačnih elemenata (MKE, engl. FEM).

Korišćenjem softvera za simulaciju brizganja plastike može se povećati kvalitet dobijenih plastičnih proizvoda, smanjiti vreme potrebno za razvoj novog proizvoda i troškove koji nastaju kao posledica nepravilne konstrukcije proizvoda [3].

Da bi simulacija injektionog presovanja bila uspešna, potrebno je ispratiti korake date na slici 1 [2].



Slika 1. Koraci pri simulaciji injektionog presovanja

U nastavku radu izvršena je analiza i dati su rezultati numeričke simulacije procesa injektionog presovanja multi-funkcionalne posude zapremine 5l, dobijenih korišćenjem programskog paketa Moldex3D.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Mladomir Milutinović.

2. VIŠENAMENSKA POSUDA ZAPREMINE 5L

Kao primer polimernog otpreska koji se izrađuje procesom injektionog presovanja uzeta je višenamenska posuda zapremine 5 litara.

Sama konstrukcija posude treba da zadovolji dva kriterijuma:

- funkcionalne kriterijume,
- estetske kriterijume.

Višenamenska posuda namenjena je za upotrebu u domaćinstvu, odnosno za čuvanje i skladištenje hrane za ljudsku upotrebu na kraći i duži period. Takođe je predviđena za čuvanje hrane u zamrzivaču, na temperaturama ispod 0 °C, za čuvanje hrane u frižideru, kao i odlaganje tople hrane, što znači da materijal mora biti postojan na niskim i visokim temperaturama. Materijal za ovaj proizvod mora da ima veću krutost, kako ne bi došlo do izvijanja i krivljenja prilikom manipulisanja sa napunjenom posudom.

Na slici 2 dat je izgled višenamenske posude.



Slika 2. Višenamenska posuda zapremine 5l

2.1 Materijal za izradu posude

Za izradu datog proizvoda usvojen je materijal *polipropilen*. Polipropilen spada u polimere iz grupe termoplasta koji ima izuzetnu kombinaciju mehaničkih, toplotnih, i električnih osobina tako da ima široku primenu u praksi. U prirodnom stanju je proziran i može se bojiti u razne nijanse. Gustina ovog materijala je od 900 do 907 kg/m³, i spada u termoplaste sa najnižom gustinom. PP se odlikuje dobrim mehaničkim osobinama: visokom čvrstoćom istezanja, visokom površinskom tvrdoćom, visokom savojnom čvrstoćom [4].

Jedna od najvažnijih osobina polipropilena je njegova postojanost na visokim temperaturama. Izdržava temperature i do 120°C bez deformacije i promene mehaničkih osobina. Takođe ima dobru toplotnu stabilnost i na temperaturama ispod 0°C [5].

PP nije toksičan, pa se može koristiti za proizvode koji su u kontaktu sa hranom. Hemijska postojanost mu je odlična, čak i na temperaturama gde se drugi termoplasti ne mogu upotrebiti. Postojan je prema vodi, praktično da i ne apsorbuje vodu.

3. SIMULACIJA PROCESA INJEKCIONOG PRESOVANJA VIŠENAMENSKE POSUDE 5L

Numerička simulacija procesa injektionog presovanja višenamenske posude izvršena je u programu Moldex 3D, koji omogućava potpunu vizuelizaciju i analizu svih faza procesa injektionog presovanja, uključujući i detekciju potencijalnih odstupanja fizičko-mehaničkih i geometrijskih karakteristika delova od projektovanih. Softver omogućava proveru valjanosti dizajna alata, izbora materijala,

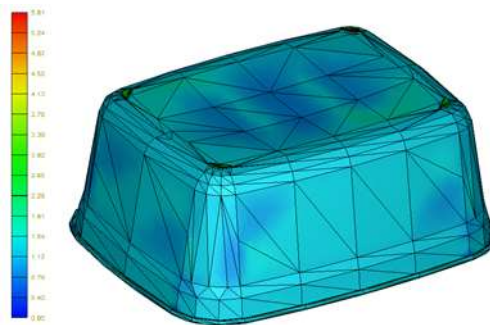
mašine i parametara procesa, kao i vrlo jednostavnu i brzu korekciju dizajna alata i optimizaciju procesa kako bi se eliminisale eventualne greške u procesu.

3.1 Definisane simulacionog modela

Pre izvođenja numeričke analize vrši se definisanje simulacionog modela, odnosno pretprocesiranje. Pretprocesiranje se vrši u pet koraka:

1. Uvoz CAD modela u .STL fajlu, provera mreže modela, prikaz i modifikovanje debljine zidova modela.
2. Definisane mesta ulivanja i izgled ulivnog sistema.
3. Kreiranje ili uvoz pokretnog i nepokretnog dela alata, kreiranje ili uvoz kanala za hlađenje, definisanje ulaza i izlaza rashladnog sredstva, kao i provera istog.
4. Generisanje mreže.
5. Izvoz mreže modela.

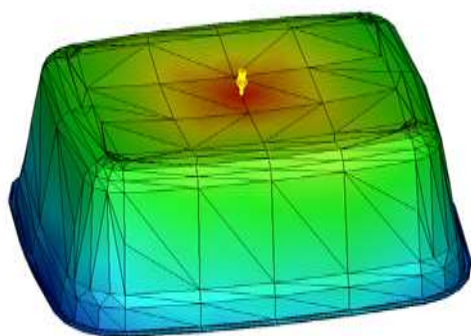
U prvom koraku pretprocesiranja, nakon uvoza CAD modela vrši se analiza debljine zida modela. Na prikazu modela debljine zidova mreže konačnih elemenata kalupnih šupljina mogu se uočiti eventualni nedostaci vezani za debljinu dizajniranog dela. Slika 3 pokazuje da je debljina zidova otpreska dosta ujednačena, i sa tog stanovišta geometrija obratka je povoljna za izradu procesom injektionog presovanja.



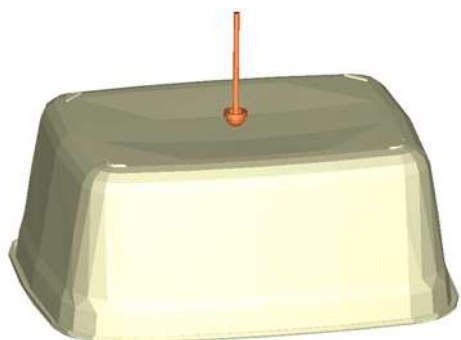
Slika 3. Analiza debljine zida dela

Za brizganje posude predviđen je alat sa jednom kalupnom šupljinom. Kod ovakvih alata za brizganje jednog otpreska najčešće se ubrizgavanje vrši direktno, iz jednog ulivnog mesta koje se postavlja u središte otpreska. Softver preko odgovarajućih algoritama automatski sugeriše najpovoljniju lokaciju za postavljanje ušća. Na slici 4 prikazano je najpovoljnije mesto za postavljanje jednog ušća.

Ubrizgavanje rastopljenog polimera se vrši toplom mlaznicom preko toplog ulivnog sistema. Ovakav ulivni sistem obezbeđuje besprekorno napajanje kalupa rastopljenim polimerom, bez gubitka materijala [6]. Softver nudi mogućnost definisanja različitih tipova ulivnog sistema ili uvoz prethodno kreiranog ulivnog sistema u nekom od CAD softvera. Uvezeni ulivni sistem je kružnog poprečnog preseka i ima oblik kanala tople mlaznice, slika 5.



Slika 4. Prikaz najpovoljnijeg mesta za postavljanje jednog ušća

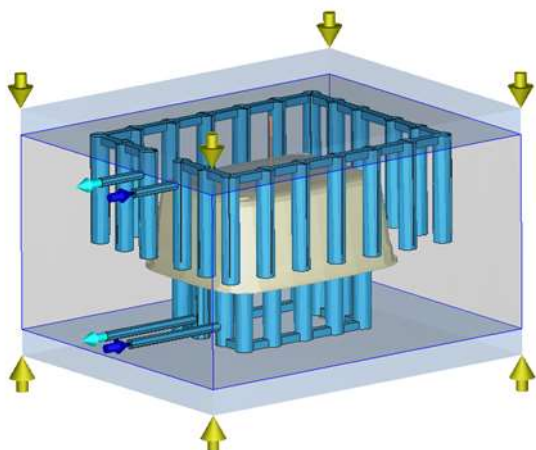


Slika 5. Topli ulivni sistem

U trećem koraku pretprocesiranja vrši se definisanje:

- dimenzija ploča alata,
- dizajn kanala za hlađenje,
- ulaza i izlaza rashladne tečnosti,
- provera kanala za hlađenje, slika 6.

Kao i kod ulivnog sistema, za dizajn rashladnog sistema nije korišćen alat koji nudi gotove oblike kanala za hlađenje čije se dimenzije parametarski definišu, nego je uvežen kao CAD model i definisan kao rashladni sistem.



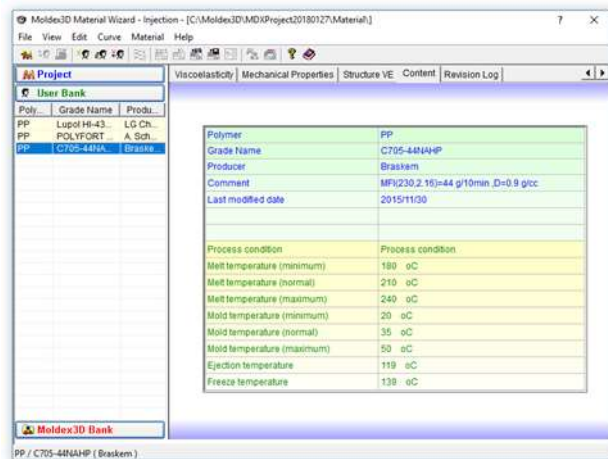
Slika 6. Definisanje dimenzija ploča alata i položaja kanala za hlađenje

3.2 Definisanje parametara injektionog presovanja

Pre definisanja parametara injektionog presovanja, pritisak i brzinu ubrizgavanja, vreme hlađenja itd, potrebno je definisati materijal otpreska. Iz bogate baze polimernih materijala koje nudi ovaj softver izabran je polipropilen C705-44NAHP. Uslovi prerade ovog polipropilena dati su na slici 7.

Nakon izbora materijala vrši se definisanja parametri injektionog presovanja. Ovi parametri se mogu usvojiti na osnovu preporuke softvera ili se mogu zadati. Parametri injektionog presovanja su sledeći:

- temperatura rastopa: 240 °C,
- temperatura kalupa: 50 °C,
- maksimalni pritisak ubrizgavanja: 200 MPa.



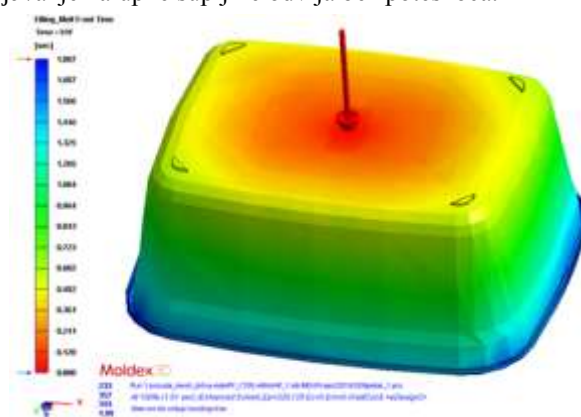
Slika 7. Uslovi prerade materijala PP C705-44NAHP

3.3 Rezultati simulacije

Simulacija injektionog presovanja posude izvršena je u četiri modula softverskog paketa Moldex 3D:

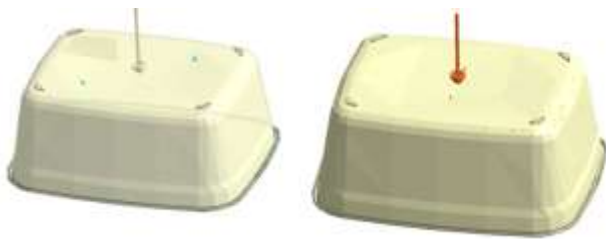
- popunjavanje kalupne šupljine (Flow),
- delovanje naknadnog pritiska (Pack),
- očvršćavanje (Cool),
- deformisanje otpreska u kalupu (Warp).

Vreme popunjavanja kalupa i distribucija rastopljenog materijala na kraju faze popunjavanja kalupa prikazano je na slici 8. Vreme popunjavanja kalupa je 1.8 s. Analizom ovog rezultata primećuje da se tečenje materijala i popunjavanje kalupne šupljine odvija bez poteškoća.

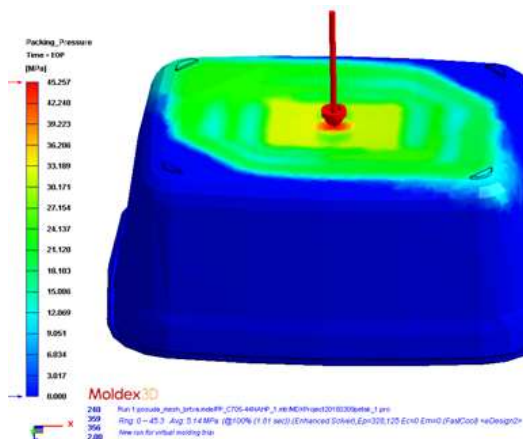


Slika 8. Vreme popunjavanja kalupa

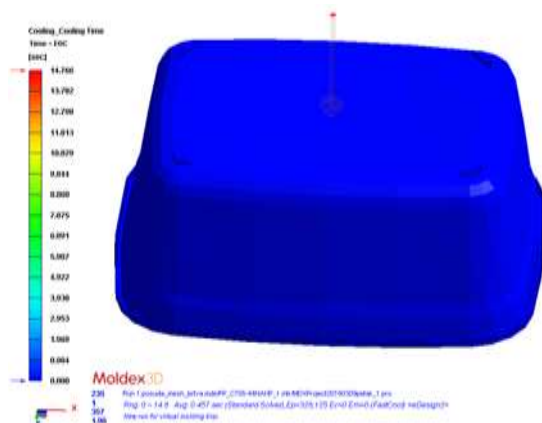
Iako tok materijala nije podeljen, primećuje se pojava linija spajanja po obodu dna otpreska. Linije spajanja su kratke, sa nešto nižom temperaturom od temperature rastopa, tako da neće uticati na mehaničke osobine i vizuelan izgled otpreska, slika 9. Pojava gasnih uključaka (sl.10) je izražena po obodu posude, gde materijal najkasnije popunjava kalup, a smanjenje istih se može izvršiti postavljanjem segmenata za odvođenje gasova. Vreme potrebno da se u svakoj oblasti dostigne temperature izbacivanja je 14.76 s, slika 11.



Slika 9. Gasni uključci i linije spajanja

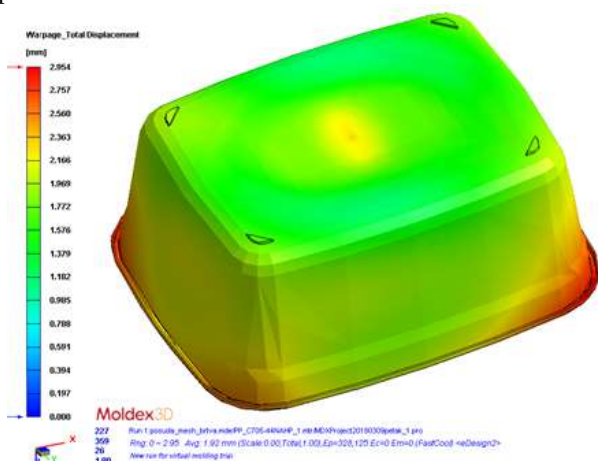


Slika 10. Distribucija naknadnog pritiska



Slika 11. Vreme hlađenja otpreska

Rezultati analizom rezultata deformacija (Warp) pokazuju da može doći do blagog krivljenja (vitoperenja) posude, što proizilazi iz nešto bržeg hlađenja dva dijagonalna čoska otpreska, Slika 12. Vrednost vitoperenja nije velika, tako da neće uticati na vizuelni izgled i funkcionalnost posude.



Slika 12. Ukupno krivljenje otpreska

4. ZAKLJUČAK

Uvođenjem numeričkih simulacija u proces projektovanja alata znatno se može povećati kvalitet otpresaka, smanjiti vreme koje je potrebno za razvoj novog proizvoda i smanjiti troškovi vezani za modifikaciju alata usled nepravilne konstrukcije proizvoda.

U radu je predstavljen prikaz numeričke simulacije injektionog presovanja na praktičnom primeru višenamenske posude zapremine 5 litara. Predviđeno mesto ulivanja i ulivni sistem omogućava brzo i lako popunjavanje kalupne šupljine. Pojava linija spajanja neće imati poseban uticaj na krajni izgled otpreska. Gasni uključci koji se mogu javiti po obodu posude mogu se otkloniti smanjenjem brzine rastopa, smanjenjem temperature rastopa i postavljanjem segmenata za odvođenje vazduha. Nakon hlađenja otpreska moguća je pojava vitoperenja otpreska, čija vrednost nije velika, tako da neće imati veliki uticaj na izgled i funkcionalnost otpreska.

5. LITERATURA

- [1] Čatić I.: *Proizvodnja polimernih tvorevina*, BIBLIOTEKA POLIMERSTVO–SERIJA ZELENA, Zagreb, 2006.
- [2] Godec, D.: *Računalna simulacija procesa injeksijskoga prešanja*, Polimeri, Vol.27, No.1, 2006. pp 13-19
- [3] Trifunović, B., Nedić, B., Erić, M., Janković, M.: *Introduction to digital production of plastic products*, FQ2015 Festival kvaliteta, Kragujevac, 2015., pp A-227-A-234
- [4] Perošević B.: *Kalupi za injektiono presovanje plastomera (termoplasta)*, Naučna knjiga, Beograd, 1995.
- [5] Nađ M.: *Termoplastične mase, prerada postupkom injektionog prešanja*, Publicitas, Zagreb 1974.
- [6] Vilotić D.: *Tehnologija injektionog presovanja polimera*, Univerzitet u Novom sadu, Fakultet tehničkih nauka, Novi Sad, 2007.

Kratka biografija:



Miloš Stepanović rođen je u Beogradu 1989. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Mašinstva – Mašine za obradu deformisanjem odbranio je 2014.god.



Mladomir Milutinović rođen je u Arilju 1967. Doktorirao je na Fakultetu tehničkih nauka u Novom Sadu 2013. god. i iste godine izabran u zvanje docenta. Zaposlen je na katedri za tehnologije oblikovanjem i inženjerstvo površina.

**ZNAČAJ PRIMENE DIREKTIVA I STANDARDA EU
ZA IZVOZNO ORIJENTISANA PROIZVODNA PREDUZEĆA****IMPORTANCE OF THE IMPLEMENTATION OF EU DIRECTIVES AND STANDARDS
FOR EXPORT-ORIENTED MANUFACTURING ENTERPRISES**

Ana Šavija, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj – Osnovni cilj ovog rada se odnosi na prikaz značaja direktiva i standarda EU za izvozno orijentisana preduzeća, sa aspekta certifikovanja proizvoda CE znakom. U radu su prikazani moduli za ocenu usaglašenosti proizvoda, odgovornosti učesnika, sadržaj tehničkog dosijea proizvoda, kao i opis osnovnih koraka koji su potrebni za dolazak do CE znaka.

Ključne reči: Direktive i standardi EU, Izvozno orijentisana preduzeća, CE znak za proizvode

Abstract – The main subject and objective of this paper is the presentation of the importance of EU directives and standards for export oriented enterprises, primarily for certification of products with CE mark. The paper presents the product conformity assessment modules, the content of the technical dossier of the product, the responsibilities of the participants, and a description of the basic steps required to reach the CE mark.

Keywords: EU Directives and Standards, Export oriented companies, CE mark for products

1. UVOD

Osnovni cilj standardizacije je stvaranje potrebnih uslova za postizanje visokog kvaliteta proizvoda, procesa i usluga, kao i ekonomičnosti u razvoju, proizvodnji, prometu i primeni proizvoda. Pored toga, značajni ciljevi standardizacije su eliminisanje prepreka u protoku roba i kapitala i omogućavanje lakšeg i efikasnijeg sporazumevanja učesnika procesa, različitih organizacija na nacionalnom i međunarodnom nivou [1]. Na evropskom nivou stvorene su i rade tri organizacije za standardizaciju:

- CENELEC (Evropski komitet za standardizaciju u elektrotehnici) u području elektrotehnologije,
- ETSI (Evropski komitet za telekomunikacione standarde) je kompetentan za telekomunikacije i
- CEN (Evropski komitet za standardizaciju) u svim ostalim područjima.

EU ima jedinstveno tržište čija je osnovna karakteristika slobodno kretanje ljudi, roba, usluga i kapitala. Da bi unutrašnje tržište moglo da funkcioniše po principu jednakosti i fer konkurencije, moraju se poštovati određena pravila. Princip slobode kretanja je obuhvaćen Ugovorom o Evropskoj zajednici, a specifična pravila su definisana propisima i direktivama.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dejan Lukić, vanr. prof.

Veliki broj proizvoda široke potrošnje i industrijskih proizvoda koji su obuhvaćeni direktivama Novog pristupa, moraju da imaju CE znak. Njime proizvođač garantuje da proizvod ispunjava sve zahteve date u direktivama koje se odnose na taj proizvod i da je obavljena procedura ocenjivanja usaglašenosti. Pored toga, proizvođač izdaje Izjavu o usaglašenosti (EC deklaracija).

U ovom radu su prikazani moduli za ocenu usaglašenosti proizvoda, odgovornosti učesnika, sadržaj tehničkog dosijea proizvoda, kao i opis osnovnih koraka koji su potrebni za dolazak do CE znaka.

2. EVROPSKE DIREKTIVE I STANDARDI ZA IZVOZNO ORIJENTISANA PREDUZEĆA

Zajedničko tržište EU je jedna od najvećih dostignuća, gde roba, usluge, kapital i radna snaga mogu slobodno da se kreću, što obezbeđuje osnovu za prosperitet EU i njenim članicama u 21. veku. Radi ostvarivanja ovog cilja EU je razvila originalne modele i mehanizme za uklanjanje prepreka za slobodno kretanje roba, za čega se koriste sledeći prilazi: *Novi pristup koji definiše propise za proizvode* i *Globalni pristup koji definiše model za ocenjivanje usaglašenosti* [2, 3].

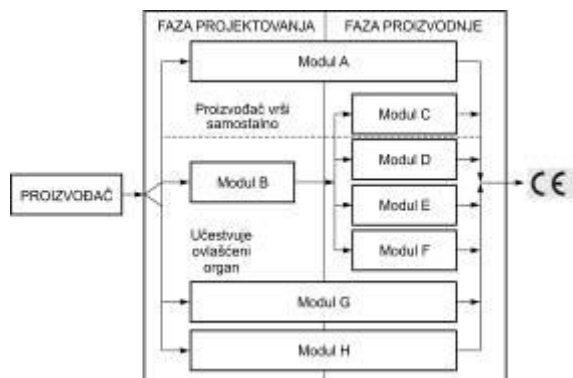
Direktive ostavljaju određeni stepen slobode nacionalnim nadležnim organima u odlučivanju na koji način će provesti ciljeve EU u okviru nacionalnog pravnog sistema. Ovi dokumenti, zajedno sa uredbama smatraju se najvažnijim zakonskim instrumentima. Svrha im je usaglašavanje različitih ciljeva EU i njenih država članica. Isto tako, smatraju se sredstvima za harmonizaciju, što podrazumeva eliminaciju nacionalnih kontradikcija i konflikata među nacionalnim zakonima.

Direktive su najvažniji zakonski instrument u uspostavljanju i jačanju unutrašnjeg tržišta EU. Direktive Novog pristupa i njihova primena se zasnivaju na sledećim principima [2]:

1. Harmonizacija je ograničena na bitne zahteve,
2. Samo proizvodi koji ispunjavaju bitne zahteve mogu se plasirati na tržište i pustiti u upotrebu,
3. Smatra se da su harmonizovani standardi objavljeni u službenom listu i preneseni u nacionalne standarde, usaglašeni sa odgovarajućim najvažnijim zahtevima,
4. Primena harmonizovanih standarda ili drugih tehničkih specifikacija ostaje dobrovoljna i proizvođači mogu slobodno da odaberu bilo koje tehničko rešenje koje obezbeđuje usaglašenost sa najvažnijim zahtevima i
5. Proizvođači biraju različite postupke ocenjivanja usaglašenosti prema direktivi koja se primenjuje.

Osnovne postavke modularnog pristupa za ocenjivanje usaglašenosti proizvoda prema direktivama novog pristupa su (zaključak Saveta 93/46/EEC):

- Utvrđivanje usaglašenosti je podeljena na module sa ograničenim brojem procedura, koje su primenljive na široki spektar proizvoda (Slika 1),
- Moduli se odnose na fazu projektovanja proizvoda i na fazu proizvodnje ili na obe faze. Postoji osam osnovnih modula (A do H), tako da kombinacijom na različite načine dobivamo postupke za ocenjivanje usaglašenosti proizvoda, prema zahtevu direktive,
- Za proizvod je potrebno izvršiti procenu usaglašenosti prema odgovarajućem modulu i
- Svaka direktiva novog pristupa opisuje vrstu i sadržaj mogućih postupaka za utvrđivanje usaglašenosti, da bi se postigao potrební stepen zaštite.



Moduli za ocenu usaglašenosti	
A – Interna kontrola proizvodnje	E – Obezbed. kvaliteta proizvoda
B – EC ispitivanje tipa	F – Verifikacija proizvoda
C – Usaglašenost sa tipom	G – Jedinična verifikacija
D – Obezbed. kvaliteta proizvodnje	H – Potpuno obezbed. kvaliteta

Slika 1. Osnovni moduli globalnog pristupa [2, 3]

3.0 TEHNIČKA DOKUMENTACIJA ZA PROIZVOD

Proizvođači moraju pripremiti odgovarajuću tehničku dokumentaciju za proizvode, koja ima značajnu ulogu kod ispunjavanja zahteva predviđenih Direktivama novog pristupa. Tehničkom dokumentacijom se obezbeđuju informacije o fazama projektovanja i proizvodnje proizvoda. Sadržaj tehničke dokumentacije za proizvode naveden je u direktivama Novog pristupa. Tehnička dokumentacija obično ima dva dela:

- Opšti deo (podaci o proizvođaču, opis proizvoda, spisak korišćenih standarda, EC deklaraciju, itd.) i
- Deo sa poverljivim informacijama (postupci ispitivanja i rezultati, analiza rizika i rešenja za uklanjanje opasnosti, sertifikati korišćenih materijala, sertifikati Ovlašćenih organa, itd.).

Tehničke podloge ili dokumentacija moraju sadržavati sve informacije relevantne za ocenjivanje i to:

- opšti opis proizvoda,
- koncepciju konstrukcije i proizvodne crteže i šeme komponenti, podsklopova, sklopova itd.,
- opis i neophodna objašnjenja za razumevanje navedenih crteža i šema i funkcionisanja proizvoda,
- spisak standarda koji se primenjuju u celosti ili delom i opise rešenja usvojenih da bi se ispunili bitni zahtevi direktive, ukoliko navedeni standardi nisu primenjeni,

- rezultate obavljenih proračuna konstrukcije, izvršenih ispitivanja, itd., kao i izveštaje o ispitivanju.

Tehnički dosije čuva proizvođač ili njegov ovlašćeni zastupnik u EU i na zahtev stavlja na raspolaganje organima za nadzor nad tržištem. Ukoliko direktivom nije definisano drugačije, ova dokumentacija se čuva najmanje 10 godina nakon proizvodnje poslednjeg proizvoda.

Uputstvo za upotrebu proizvoda je važan deo tehničke dokumentacije. Ono se podešava prema vrsti proizvoda, vrstama opasnosti i znanju korisnika, a obuhvata:

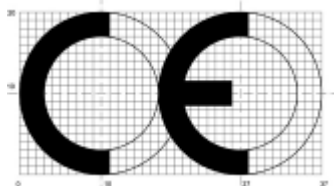
- Puštanje u rad i upotrebu proizvoda,
- Montažu, rastavljanje i podešavanje,
- Održavanje i popravku i
- Obuku, instrukcije gde je potrebno.

Većina direktiva Novog pristupa zahteva da se tehnička dokumentacija piše na službenom jeziku države članice EU. Da bi se postupci utvrđivanja usaglašenosti odvijali na pravilan način, tehnička dokumentacija mora uvek biti napisana na jeziku koji je razumljiv Ovlašćenom organu. Nadzorni organ može tražiti prevod tehničke dokumentacije i EC-deklaracije o usaglašenosti na svoj nacionalni jezik.

Uputstva za upotrebu koja su namenjena potrošačima moraju biti na službenim jezicima država gde je proizvod plasiran. EC-deklaracija o usaglašenosti, mora biti napisana na jednom od službenih jezika država EU. Za proizvode koje prati EC-deklaracija o usaglašenosti, ona mora biti na jeziku države u kojoj je proizvod na tržištu.

4.0 OZNAČAVANJE SA CE ZNAKOM

CE označavanje je jedino označavanje kojim se potvrđuje da su proizvodi usaglašeni sa direktivama koje se zasnivaju na globalnom pristupu. CE znak je sada obavezan za veliki broj proizvoda koji se prodaju u EU. Evropska komisija opisuje **CE znak kao pasoš**, koji dozvoljava proizvođačima da slobodno prodaju proizvode širom internog tržišta EU. CE znak nije znak kvaliteta i ne pokazuje usaglašenost sa standardom, već usaglašenost sa zakonskim bitnim zahtevima direktiva EU. Oznaka CE se mora sastojati od inicijala „CE“ čiji je oblik opisan u odluci Veća 93/465/EEC ili primenljivoj direktivi, slika 2.



Slika 2. CE oznaka prema odluci 93/465/EEC [3]

CE oznaka je obavezna za proizvode obuhvaćene direktivama Novog pristupa: mora je postaviti proizvođač/ovlašćeni predstavnik/uvoznik, usmerena je prema institucijama sistema nadzora nad tržištem, nije znak kvaliteta, nije oznaka za poreklo, mora biti postavljena na proizvod/ natpisnu pločicu /pakovanje / prateće dokumente, mora biti vidljiva, čitka, neizbrisiva, ne može se kombinovati sa oznakama kvaliteta, mora se postaviti nakon sastavljanja EC deklaracije o usaglašenosti, može deklaracija sadržavati identifikacioni broj ovlašćenog organa (ukoliko je uključena u proceduru ocenjivanja usaglašenosti u fazi proizvodnje).

4.1 Odgovornost proizvođača i tržišni nadzor

U procesu projektovanja, proizvodnje, ispitivanja i prodaje proizvoda uključeni su različiti učesnici, dok na tržištu EU glavnu odgovornost za proizvod ima proizvođač. On mora da obezbedi da se proizvod koji je namenjen za tržište EU, projektuje i izgradi u saglasnosti sa zahtevima direktiva Novog pristupa. To znači da se postigne usaglašenost proizvoda sa bitnim zahtevima.

Odgovornost ostalih učesnika: ovlašćenih zastupnika, uvoznika, distributera, monter, instalatera, korisnika, laboratorija i ovlašćenih organa je takođe definisana u zakonodavstvu EU.

Kontrola proizvoda na tržištu ili nadzor nad tržištem je nerazdvojivi deo sistema primene zakona u oblasti legislative proizvoda. U državama članicama EU nadzor nad tržištem je u nadležnosti države ili njenih lokalnih organa vlasti. Strane koje provode nadzor nad tržištem u Evropskoj uniji su [2]:

- Proizvođači – pre plasiranja proizvoda na tržište,
- Carinski službenici – za proizvode poreklom izvan EU,
- Ovlašćene organizacije za provođenje nadzora – nakon plasiranja proizvoda na tržište i
- Sudstvo – pri pojavi incidenta.

Tržišni nadzor omogućuje da se efikasnom kontrolom postigne bezbednost proizvoda na tržištu. Države članice EU su obavezne da organizuju i vrše tržišni nadzor, na način koji omogućuje da se identifikuju neusaglašeni proizvodi. Na taj način se vrši zaštita interesa potrošača, radnika, korisnika i proizvođača od nelojalne konkurencije. Tržišne inspekcije proveravaju proizvode: redovnim obilascima trgovačkih industrijskih i skladišnih prostora; redovnim obilascima radnih mesta i ostalih prostora gde su proizvodi dati na korišćenje; organizovanjem slučajnih nezakonskih kontrola i uzimanjem uzroka proizvoda koje daje na kontrolu.

4.2 Koraci do CE znaka

Postupak CE označavanja u velikoj meri zavisi od proizvoda i direktiva koje se na predmetni proizvod primenjuju i za proizvođače ne predstavlja nimalo jednostavan i lak put do CE znaka. Osnovni koraci koje u većini slučajeva moraju da prođu proizvođači koji na svoje proizvode žele da stave CE znak, bez obzira na vrstu proizvoda i direktive, su dati u nastavku [4, 5, 6, 7]:

1. Pokretanje projekta i formiranje tima za CE znak

Preduzeće donosi odluku da želi da pristupi pripremi za dobijanje CE znaka za svoje odabrane proizvode i odabira način realizacije ovog projekta (samostalno-sopstveni tim ili uz stručnu pomoć-konsultantska organizacija/tim, notifikovano telo).

Projektni tim sačinjava plan realizacije projekta pri čemu definiše sledeće glavne grupe aktivnosti [5]: sagledavanje relevantnih direktiva i harmonizovanih standarda; procena mogućnosti realizacije cilja (asortiman potrebnih ispitivanja, procena troškova i vremena) – I kontrolna tačka; definitivni asortiman proizvoda namenjenih za CE označavanje; definitivni izbor direktiva, standarda i ispitivanja sa ocenom troškova i vremena – II kontrolna tačka; aktivnosti obuke zaposlenih preduzeća (opšta i posebna obuka); kontakt sa notifikovanim telom;

ocenjivanja (ocena rizika, ocena upravljanja kvalitetom); priprema tehničke dokumentacije i prevod; dobijanje atesta o usaglašenosti od Notifikovanog tela i izrada EC deklaracije o usaglašenosti i Označavanje CE znakom.

2. Sagledavanje proizvoda i procesa proizvodnje

Potrebno je sagledati kako se vrši projektovanje proizvoda i tehnoloških procesa (tehnička dokumentacija), odnosno kako se vrši proces proizvodnje proizvoda (dijagram toka, osnovni proizvodni procesi, logistički procesi) u saglasnosti sa ISO 9001:2008.

Preduzeća koja imaju primenjen standard ISO 9001 i definisane i dokumentovane proizvodne procese kojima se upravlja, mogu samo da provere njihovu usklađenost sa praksom i da pređu na sledeći korak. Za preduzeća koja nemaju ISO 9001 se preporučuje (nije obaveza) da ga implementiraju i iskoriste njegove prednosti za lakše dokumentovanje procesa [2, 4].

3. Izbor asortimana proizvoda za CE znak

Izbor asortimana najčešće zavisi od tržišta na koja se plasiraju proizvodi, kao i od prioriteta koje ima preduzeće. Obično se biraju proizvodi za koje postoji mogućnost plasmana na tržište EU, a kasnije se postepeno uvode i ostali proizvodi. Bitno je naglasiti da CE označavanje treba sprovesti pre izvoza, kako bi se izbegli problemi na carini, nemogućnost ugovorene isporuke, raskid ugovora, kašnjenja i sl., jer postupak ocenjivanja usaglašenosti za CE označavanje zahteva određeno vreme [4].

4. Identifikacija direktiva

Identifikacija direktiva se vrši na osnovu: vrste i namene proizvoda, načina funkcionisanja, vrste pogona i tipa sastavnih komponenti, sklopova, delova. Na jedan proizvod se može odnositi jedna ili više direktiva, s tim da je za CE označavanje potrebno primeniti sve direktive koje se odnose na predmetni proizvod [5].

5. Identifikovanje harmonizovanih standarda

U skladu sa identifikovanim direktivama, vrši se identifikacija harmonizovanih standarda koji se odnose na predmetni proizvod. Na jedan proizvod se može odnositi jedan ili više harmonizovanih standarda, s tim da je za CE označavanje potrebno primeniti sve standarde koji se odnose na predmetni proizvod [4]. Na osnovu proučavanja standarda ocenjuju se osnovne performanse koje proizvod treba da zadovolji, koja ispitivanja treba obaviti na materijalima, delovima, kao i na samom proizvodu.

6. Izbor procedure za ocenu usaglašenosti proizvoda

Ocenjivanje usaglašenosti vrši proizvođač ili notifikovano telo, u zavisnosti od modula koji se primenjuje (A-H), prema slici 1. Ocenjivanje usaglašenosti obuhvata fazu projektovanja, fazu proizvodnje proizvoda ili obe faze (u skladu sa karakteristikama proizvod, rizikom pri upotrebi i dr.).

7. Analiza rizika

Ako ne postoje harmonizovani standardi koje pozivaju direktive, ili oni ne pokrivaju sve rizike za sve bitne zahteve, onda je neophodno sprovesti postupak analize i ocene rizika korišćenja proizvoda.

Analizom rizika se identifikuje nivo opasnosti i procenjuje verovatnoća da proizvod postane opasan. Kada se utvrdi da postoje karakteristike koje posebno utiču na bezbednost proizvoda, one se moraju ispitati kako bi se

utvrdilo da li se nalaze u granicama dozvoljenog odstupanja. Treba uzeti u razmatranje i mogućnosti upotrebe proizvoda pre svega ako proizvod može doći u kontakt sa neobučenim osobama, starim licima ili decom.

8. Priprema tehničkog fajla

Tehnički fajl treba da sadrži tehničku dokumentaciju kojom se dokazuje usaglašenost proizvoda sa zahtevima evropskih direktiva. Tehnički fajl treba da sadrži projektnu, proizvodnu i korisničku dokumentaciju [2, 4].

U projektnu dokumentaciju najčešće spadaju projektni crteži proizvoda, tehnički opis proizvoda, detalji o obliku, izgledu, sklopovima i sastavnim delovima proizvoda i dr.

Proizvodna dokumentacija sadrži tehničke detalje o procesu proizvodnje, odgovarajuću prateću dokumentaciju za sastavne delove i dr., izveštaje o ocenjivanjima usaglašenosti, dokaze o kvalitetu, ateste i drugu potrebnu dokumentaciju u skladu sa prirodom proizvoda.

Korisnička dokumentacija najčešće obuhvata uputstva za upotrebu i uputstva za sastavljanje/montažu. Za složenije proizvode je potrebna dokumentacija za montažera (ako proizvod zahteva stručnu montažu pre prvog puštanja u rad), kao i dokumentacija za servisera (ako proizvod zahteva stručno održavanje i popravke), i dr.

9. Imenovanje ovlašćenog predstavnika

S obzirom na to da Srbija još uvek nije članica EU, srpski proizvođači treba da imaju ovlašćenog predstavnika koji se nalazi na teritoriji Evropske unije. Predstavnik i proizvođač moraju imati ugovorno regulisane odnose sa jasnim obavezama i odgovornostima. Ovlašćeni predstavnik ne može da vrši bilo kakve modifikacije na finalnom proizvodu proizvođača, sa namerom da isti izmeni i uskladi sa direktivama. Ovlašćeni predstavnik u EU mora da bude upoznat sa tehničkom dokumentacijom proizvoda za CE označavanje, koja dokazuje usaglašenost proizvoda sa evropskim direktivama [4].

10. Izbor notifikovanog tela

Izbor notifikovanog tela vrši proizvođač u skladu sa komercijalnim uslovima i odobrenjima koja imaju notifikovana tela. Notifikovana tela se nalaze na teritoriji EU, a odobrenja za rad dobijaju direktno od Evropske Komisije [2, 3]. Kada notifikovano telo sprovede ocenu usaglašenosti, ono izdaje atest o usaglašenosti.

11. Izrada EC deklaracije o usaglašenosti

Na osnovu Atesta o usaglašenosti izrađuje se EC deklaracija o usaglašenosti na osnovu koje se vrši CE označavanje. Proizvođač sam izrađuje EC deklaraciju o usaglašenosti. Sadržaj EC deklaracije o usaglašenosti propisan je svakom pojedinačnom direktivom.

CE atest (sertifikat) o saobraznosti mora da sadrži: naziv i adresu sertifikacionog tela, naziv i adresu proizvođača ili ovlašćenog predstavnika u EU, opis proizvoda i odredbe kojima je proizvod saglasan [5].

12. Označavanje proizvoda CE znakom

Stavljanje CE znaka vrši proizvođač u skladu sa primenljivim direktivama. CE znak mora imati oblik koji je propisan u direktivi i mora biti vidljiv, čitljiv i postojan da se ne bi lako brisao. Projektovana identifikaciona tablica za familiju Univerzalnih međurednih kultivatora UMK, koji su bili predmet razmatranja u master radu [7], prikazana je na slici 3.

 JOŽEF HUĐIK PR HUĐIK Pap Pala 127 21235 Temerin Srbija http://www.hudik.com 			
Naziv mašine Machine name		Univerzalni međuredni kultivator Universal inter row cultivator	
Tip Type		UMK-8K	
Standard Standard		EN ISO 4254-1:2013; EN ISO 12100:2010	
Serijski broj Serial No.	21	Radna širina [cm]/Broj redova Working width / No. of rows	630 / 8
Godina proizvodnje Year of manufacturing	2016	Približna masa mašine [kg] Machine weight	1020

Slika 3. Identifikaciona tablica familije proizvoda UMK

5.0 ZAKLJUČAK

U svetu je trenutno prisutna tendencija globalizacije kako tržišta, tako i standardizacije na regionalnom i svetskom nivou. Evropska unija ima jedinstveno tržište čija je osnovna karakteristika slobodno kretanje ljudi, roba, usluga i kapitala. Tržište EU se bazira na principu jednakosti i ravnopravnosti konkurencije, pri čemu se moraju poštovati određena pravila.

Veliki broj proizvoda široke potrošnje i industrijskih proizvoda koji su obuhvaćeni direktivama Novog pristupa, moraju da imaju CE znak ukoliko žele da budu plasirani na tržište EU. U radu je predstavljen značaj direktiva i standarda EU za izvozno orijentisana domaća preduzeća koja žele da izvoze u EU, dok je akcenat dat na prikaz načina dolaska do CE znaka.

6.0 LITERATURA

- [1] V. Miladinović: Ciljevi, principi i struktura procesa standardizacije, *Naučnotehnički pregled*, vol. L, br. 6, pp. 83-93, 2000.
- [2] V. Jovišević, S. Borojević: *Standardizacija i industrijska legislativa*, Mašinski fakultet, Banja Luka, 2010.
- [3] D. Bijelić: *Vodič - Pitanja i odgovori u vezi sa CE označavanjem proizvoda*, Republički zavod za standardizaciju i metrologiju, Banja Luka, 2012.
- [4] <https://www.icertgroup.org/kako-do-ce-znaka/>, 5.2.2018.
- [5] G. Jakovljević i dr.: *Deset koraka do CE znaka*, <http://adqm.rs/assets/uploads/Publikacije/2465b-10-koraka-ce-znak-2007.pdf>, 5.4.2018.
- [6] *CE znak – pasoš za proizvode*, Privredna komora Srbije, Beograd, 2014.
http://www.pks.rs/SADRZAJ/Files/WEB_CE_brosura.pdf, 6.4.2018.
- [7] A. Šavija: *Značaj primene direktiva i standarda EU za izvozno orijentisana proizvodna preduzeća*, Master rad, Fakultet tehničkih nauka, Novi Sad, 2018.

Kratka Biografija



Ana Šavija rođena je u Novom Sadu, 1993. god. Master rad na Fakultetu tehničkih nauka iz oblasti Mašinstva – Proizvodno mašinstvo, smer Računarom podržane tehnologije odbranila je 2018.god.

ГЛАВНА МЕРНО-РЕГУЛАЦИОНА СТАНИЦА СА ДВОСТЕПЕНОМ РЕДУКЦИЈОМ ПРИТИСКА ГАСА

MAIN REGULATION-METERING STATION WITH TWO-STAGE PRESSURE REDUCTION OF NATURAL GAS

Милош Пешић, Факултет техничких наука, Нови Сад

Област – МАШИНСТВО

Кратак садржај – Задатак овог рада се заснива на изради решења за редукције притиска природног гаса са магистралног на дистрибутивни ниво. Као решење је одабрана главна мерно-регулациона станица са двостепеном редукцијом притиска природног гаса. У овом раду је приказан комплетан поступак прорачуна и избора опреме за изградњу потребне гасне станице. Током израде рада се тежило ка што већој енергетској ефикасности и рационалном избору потребне опреме.

Abstract – The task of this thesis is based on developing a solution for reducing the pressure of natural gas from the main stream to the distribution stream level. As a solution has chosen the main metering-regulation station with two-stages reduction of natural gas pressure. This thesis presents a complete procedure for calculation and selection of equipment for the construction of the necessary gas station. During the development of the thesis, the aim was to increase the energy efficiency and rational choice of the chosen equipment.

Кључне речи: природни гас, главна мерно-регулациона станица, притисак гаса, регулатор притиска, мерење протока гаса...

1. УВОД

Предмет рада је главна мерно-регулациона станица (ГМРС). ГМРС чини основни део сваког гасовода. Задатак ГМРС је обезбеђивање константног излазног притиска, независно од промене улазног притиска и потрошње гаса, као и мерење протекле количине гаса. Једна од основних разлика између ГМРС и МРС је то што је пад притиска код ГМРС доста већи што води до Цул-Томпсоновог ефекта (потхлађење гаса). Услед тога свака ГМРС садржи догрејач гаса, а у највећем броју случајева и своју котларницу на природни гас. ГМРС се постављају између магистралног гасовода високог притиска (20 – 50 bar за наше услове) и гасних мрежа средњег притиска (6 – 12 bar). Одоризација природног гаса је још једна функција која се обавља на ГМРС. МРС се обавезно постављају између гасних мрежа средњег притиска, односно дистрибутивних гасних мрежа и потрошача. Притисак гаса који се предаје потрошачима се креће у границама од 100 mbar

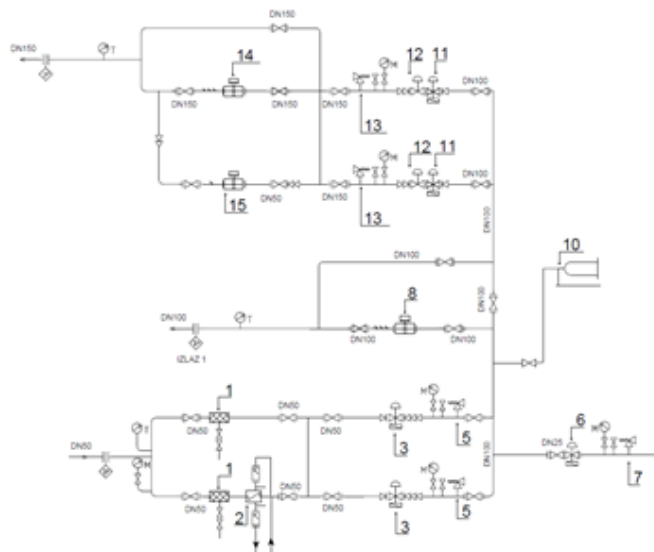
(за мање потрошаче, као што су домаћинства, мање мануфактурне радионице итд), па до 4 bar (за котларнице у главно мерно регулационој станици (ГМРС)).

Циљ рада је да се предложи техничко решење, димензионише и усвоји опрема за главну мерно-регулациону гасну станицу са двостепеном редукцијом гаса. Димензионисање и усвајање опреме ГМРС потребно је урадити за следеће услове:

Радни притисак гаса на улазу (20÷50 bar), радни притисак гаса на линији средњег притиска (9 bar), радни притисак на излазној линији ниског притиска (2 bar), радни притисак на линији према котларници (22 mbar), максимални проток природног гаса (4000 Sm³/h)

Током димензионисања се обратила пажња на то да се добије што већа ефикасност постројења са што мањим инвестиционим трошковима. Основни делови сваке ГМРС за гас треба да буду: филтер за гас, сигурносни (блок) вентил, регулатор притиска гаса, сигурносни одушни вентил, мерило протока гаса, догрејачи, одоризатори, манометри, термометри, потребни запорни уређаји. На ГМРС се поставља и телеметрија за даљински надзор и управљање радом ГМРС (овај део ГМРС неће бити предмет овог рада).

Ради лакшег праћења рада, препоручује читање уз праћење технолошке шеме која се налази у графичком прилогу мастер рада тј. на *Слици 1*.



Слика 1. Технолошка шема главне мерно-регулационе станице

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био доц. др Синиша Бикић.

2. ПРИРОДНИ ГАС

Гасови се обично деле на природне и вештачке. Природни гасови добијају се на један од следећа три начина [1]:

- из бушотина гасних лежишта:
 - гас се састоји углавном од метана (82-98 %) и
 - мањег дела тежих угљоводоника.
- из бушотина кондензатних лежишта:
 - гас се састоји од метана (80-95 %) и
 - кондензата керозина и бензина.
- из бушотина лежишта сирове нафте:
 - гас се јавља у виду мешавине метана (30-70 %) и
 - пропана и бутана.

Главни потрошачи природног гаса су индустријске котларнице, градске топлане, металуршка и цементна индустрија, домаћинства итд. У новије време, када је питање најекономичнијег искоришћења горива веома актуелно, природни гас добија све више на значају, захваљујући изузетним физичко-хемијским особинама које му омогућају најбоље искоришћење енергије акумулисане у гориву [1].

Табела 1. Састав узорка природног гаса са карактеристикама [1]

Метан, CH ₄	87,82%
Етан, C ₂ H ₆	7,96 %
Пропан, C ₃ H ₈	0,58 %
Тежи угљоводоници, C _m H _n	0,03%
Азот, N ₂	2,36%
Угљен-диоксид, CO ₂	1,25%
Доња топлотна моћ при нормалним условима (0°C, 101325 Pa), Hd [kJ/m ³]	33340
Густина гаса при нормалним условима (0°C, 101325Pa), ρ _N [kg/m ³]	0,802
Просечна вредност границе запаљивости (експлозивности) у ваздуху, [%]	4 - 16
Максимална брзина паљења, [m/s]	0,35
Температура паљења у ваздуху, T [K (°C)]	943 (670)
Температура сагоревања у стехиометријској количини ваздуха λ=1, T [K (°C)]	T = 2273 K (2000 °C)

3. ОСНОВНЕ КАРАКТЕРИСТИКЕ ГМРС

Главна мерно регулациона станица (ГМРС) је објект у коме се, притисак гаса из магистралног гасовода (20 до 50 bar), редукује на притисак градске мреже (6 до 12 bar) и истовремено врши мерење протока ради обрачуна предатих количина. ГМРС се постављају, гледајући по смеру протока гаса, на улазу у насеља [1].

Мерно регулациона станица (МРС) чини основни део сваког гасовода. Задатак МРС је обезбеђење константног излазног притиска, независно од промене улазног притиска и потрошње гаса, као и мерење протекле количине гаса [2].

Количина гаса, коју испоручује извор снабдевања, односно, коју прима транспортер, дистрибутер или потрошач представља други значајан параметар. Познавање вредности протеклих количина гаса је неопходно за обрачун између продавца и купца, а затим и за правилно вођење технолошког процеса производње, транспорта, дистрибуције, као и процеса производње код потрошача.

3.1 Основна концепција главне мерно-регулационе станице

Свака ГМРС мора имати следеће целине [1]:

- део за припрему гаса;
- регулационо-сигурносни део и
- део за мерење.

Елементи који чине и који се обавезно морају уградити у ГМРС су:

- изолациони елементи;
- цеви, спојнице итд;
- запорни органи;
- филтри за гас;
- сигурносни прекидни (блок) вентили;
- сигурносни одушни (испусни) вентили;
- одоризатор;
- догрејач гаса
- мерила протока гаса;
- коректори запремине гаса и
- контролно-мерни инструменти

4. ХИДРАУЛИЧКИ ПРОРАЧУН ЧЕЛИЧНОГ ГАСОВОДА

Прорачун пада притиска у цевоводу рачунат је према формули Реноара [1]:

$$p_1^2 - p_2^2 = 46742 \cdot \rho_R \cdot L \cdot Q^{1,82} \cdot d_u^{-4,82}$$

где су:

p₁ - притисак на почетку цевовода (апсолутни), p₁ = 51 bar;

p₂ - притисак на крају цевовода (апсолутни), p₂ = 10 bar;

ρ_R - релативна густина гаса ρ_R = 6,59;

L - дужина цевовода (од прикључка са магистралног гасовода до ГМРС), L=0,04 km;

Q - проток гаса кроз цевовод при стандардним условима, Q=4000 Sm³/h и

d_u - унутрашњи пречник цевовода d_u = 54,5 mm.

Резултати прорачуна улазног цевовода у ГМРС дати су у табели 2.

Табела 2. Резултати хидрауличног прорачуна улазног цевовода у МРС

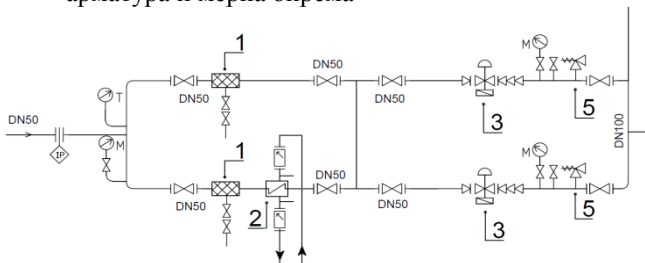
Ред број.	Деоница	L (km)	d _u (mm)	Q (m ³ /h)	p ₁ (bar)	p ₂ (bar)
1.	Прикључни шахт	0,04	54,5	4000	50	49,11

5. ДИМЕНЗИОНИСАЊЕ ЦЕВОВОДА И ИЗБОР ОПРЕМЕ

5.1. Улазни цевовод са првом регулационом линијом, DN 50/DN 100

На основу полазних података усвојена је следећа опрема, (гасна линија је приказана на Сlici 2):

- Филтери за гас са диференцијалним манометром (GASTEN модел 414/2)- сакупљачи кондензата, (позиција 1),
- Загрејач гаса (Тип ES 1040, DN 50, ANSI 300, 50 kW), (позиција 2),
- Регулатори притиска гаса („ENERGO-SISTEM тип 430-SPV, са блокадним вентилом, DN 50, ANSI 300, подешен на излазни притисак од 9 bar), (позиција 3),
- Сигурносно одушни вентил (Тип ES RVS 510, R1", PN 25), (позиција 5)
- Цеви, запорни органи (славине), потребна арматура и мерна опрема

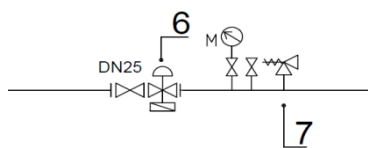


Слика 2. Технолошка шема улазне регулационе гасне линије

5.2. Линија малог капацитета према котларници

На основу полазних података усвојена је следећа опрема, (Гасна линија је приказана на Сlici 3.)

- Регулатор притиска са блокадним вентилом тип ES 225-SPV, DN 25, PN 16 (позиција 6),
- Сигурносно одушни вентил тип ES 570, R1/2", PN 2,5 (позиција 7),

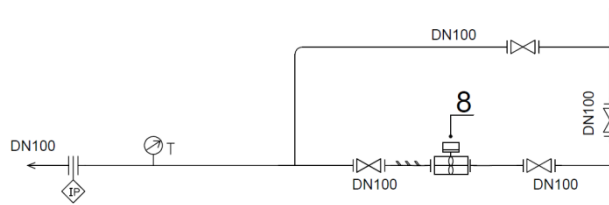


Слика 3. Технолошка шема регулационе гасне линије према котларници

5.3. Прва мерна линија, DN 100

На основу полазних података усвојена је следећа опрема, (мерна линија је приказана на Сlici 4):

- Ротационо мерило протока гаса G-400, DN 100, PN 16 са електронским коректором запремине (позиција 8),
- Цеви, запорни органи (славине), потребна арматура и мерна опрема

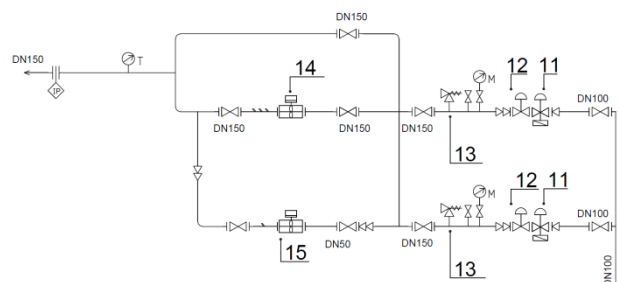


Слика 4. Технолошка шема прве мерне линије

5.4. Други излазни цевовод са мерно-регулационом линијом DN 100/DN 150

На основу полазних података усвојена је следећа опрема, (мерно-регулациона линија је приказана на Сlici 5):

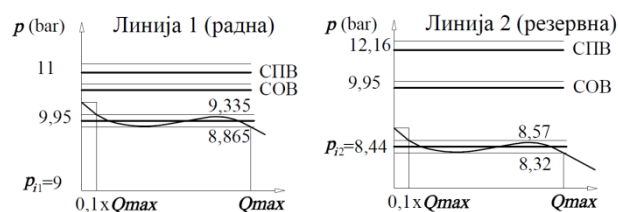
- Регулатор притиска гаса са сигурносно прекидним вентилом тип ES 330-SPV, DN 40, PN 16, (позиција 11),
- Регулатор притиска гаса тип ES 330-SPV, DN 40, PN 16, монитор регулатор, (позиција 12)
- Вентил сигурности тип ES 560 R3/4", PN 6, (позиција 13)
- Турбинско мерило протока гаса "INSTROMENT" G-650, DN 150, ANSI 150, (позиција 14)
- Мерило протока гаса са ротационим клиповима "DELTA" tip 2050-100, G40, DN 50, (позиција 15)



Слика 5. Технолошка шема друге мерно-регулационе линије

6. ПОДЕШАВАЊЕ ПАРАМЕТАРА ГАСНЕ СТАНИЦЕ

Дијаграм подешавања прве регулационе линије је приказан на слици 6. док су у табели 3. приказани карактеристични притисци регулационе линије.

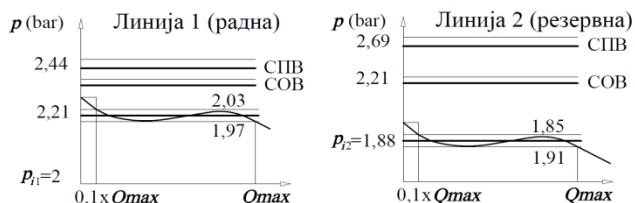


Слика 6. Дијаграм подешавања прве регулационе линије

Дијаграм подешавања друге регулационе линије је приказан на слици 7. док су у табели 4. приказани карактеристични притисци регулационе линије.

Табела 3. Карактеристични притисци прве регулационе линије

МАКСИМАЛНИ ПРОТОК ГАСА	4000	Sm ³ /h
УЛАЗНИ ПРИТИСАК	9	bar
РЕГУЛАЦИОНА ЛИНИЈА 1 (РАДНА)		
ИЗЛАЗНИ ПРИТИСАК	2	bar
ПРИТИСАК ОТВАРАЊА СОВ	2,21	bar
ПРИТИСАК ЗАТВАРАЊА СПВ	2,44	bar
РЕГУЛАЦИОНА ЛИНИЈА 2 (РЕЗЕРВНА)		
ИЗЛАЗНИ ПРИТИСАК	1,88	bar
ПРИТИСАК ОТВАРАЊА СОВ	2,21	bar
ПРИТИСАК ЗАТВАРАЊА СПВ	2,69	bar

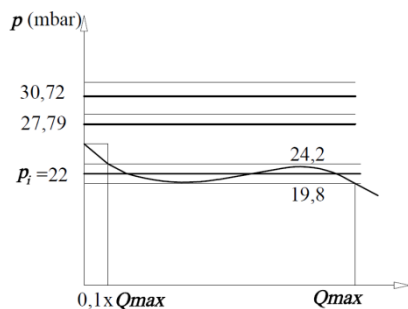


Слика 7. Дијаграм подешавања друге регулационе линије

Табела 4. Карактеристични притисци друге регулационе линије

МАКСИМАЛНИ ПРОТОК ГАСА	4000	Sm ³ /h
УЛАЗНИ ПРИТИСАК	9	bar
РЕГУЛАЦИОНА ЛИНИЈА 1 (РАДНА)		
ИЗЛАЗНИ ПРИТИСАК	2	bar
ПРИТИСАК ОТВАРАЊА СОВ	2,21	bar
ПРИТИСАК ЗАТВАРАЊА СПВ	2,44	bar
РЕГУЛАЦИОНА ЛИНИЈА 2 (РЕЗЕРВНА)		
ИЗЛАЗНИ ПРИТИСАК	1,88	bar
ПРИТИСАК ОТВАРАЊА СОВ	2,21	bar
ПРИТИСАК ЗАТВАРАЊА СПВ	2,69	bar

Дијаграм подешавања регулационе линије према котларници је приказан на слици 8. док су у табели 5. приказани карактеристични притисци регулационе линије.



Слика 8. Дијаграм подешавања регулационе линије према котларници

Табела 5. Карактеристични притисци регулационе линије према котларници

МАКСИМАЛНИ ПРОТОК ГАСА	10	Sm ³ /h
УЛАЗНИ ПРИТИСАК	9	bar
РЕГУЛАЦИОНА ЛИНИЈА		
ИЗЛАЗНИ ПРИТИСАК	22	mbar
ПРИТИСАК ОТВАРАЊА СОВ	27,79	mbar
ПРИТИСАК ЗАТВАРАЊА СПВ	30,72	mbar

7. ЗАКЉУЧАК

Главна мерно-регулациона станица чини основни део сваког гасовода и служи за обезбеђивање константног излазног притиска, независно од промене улазног притиска и потрошња гаса, као и мерење протекле количине гаса. Предмет рада је била ГМРС са два излаза (различитих излазних притисака, 9 bar и 2 bar). Циљ рада је био да се димензионише и усвоји опрема и подесе параметри предметне главно-мерно регулационе станице за гас. Усвојено је техничко решење које чини: фини филтер за гас (DN 50, PN 64), Догрејач гаса, 50 kW (DN 50, ANSI 300), Регулатор притиска са пилотом и блок вентилом (DN 40, ANSI 300), Сигурносно испусни вентил (R1", PN 25), Ротационо мерило протока гаса G400 (DN 100, PN 16), Сигурносно испусни вентил (R 1/2", PN 2,5), Регулатор притиска гаса са блок вентилом (DN 25, PN 16), Сигурносно испусни вентил (R 3/4", PN 6), Турбинско мерило протока гаса G-650 (DN 150, PN 16), Регулатор притиска гаса са пилотом и блок вентилом (DN 40, PN 16), Регулатор притиска са пилотом (DN 40, PN 16), Ротационо мерило протока гаса G-40 (DN 50, PN 16)

8. ЛИТЕРАТУРА

- [1] Душан Узелац, (2001.), Пумпе и компресорске станице, скрипта, Факултет техничких наука
- [2] ИНТЕРНА ТЕХНИЧКА ПРАВИЛА за пројектовање и изградњу гасовода и гасоводних објеката на систему ЈП СРБИЈА-ГАС, ЈП Србија-гас Народног фронта 12 Нови Сад, 2009.
- [3] Срђа Поповић (1999.): Приручник за пројектовање и израду МРС на природни гас, ЈУГАС, Београд

Кратка биографија:



Милош Пешић, рођен је у Новом Саду 1993 год. Мастер рад на Факултету техничких наука из области Машинства – Гасна техника одбранио је 2018. год.



Синиша Бикић рођен је у Кикинди 1979 Докторирао је 2013. год. са темом „Развој методе мерења протока ваздуха помоћу регулационе жалужине. Област интересовања су примењена механика флуида, рачунарска динамика флуида и гасна техника.

АНАЛИЗА УТИЦАЈНИХ ПАРАМЕТАРА НА ТАЧНОСТ МЕРЕЊА ТРАКАСТИМ ДОЗАТОРОМ**ANALYSIS OF INFLUENTIAL PARAMETERS OF MEASUREMENT ACCURACY WITH THE BELT FEEDER**Никола Иланковић, Драган Живанић, *Факултет техничких наука, Нови Сад***Област – МАШИНСТВО**

Кратак садржај – Тема рада јесте анализа утицаја затегнутости и брзине траке на тачност мерења тракастим дозатором. На почетку су дате теоријске поставке у вези са тракастим дозаторима, као и опис опреме која је коришћена приликом испитивања. Након тога је дат прорачун тракастог дозатора, анализа кретања траке, калибрација мерних давача силе, као и дискусија о утицају поменутих параметара на тачност мерења.

Кључне речи: тракасти дозатор, мерни давачи силе, калибрација

Abstract – The subject of this paper is the analysis of influential parameters of precise measurement with the belt feeder. The first part gives theoretical backgrounds of belt feeders, description of the equipment used during testing. The second part gives the calculation of the belt feeder, analysis of the belt movement, calibration of force sensors and conclusions of the influence of the mentioned parameters on the precision of measurements.

Keywords: belt feeder, force sensors, calibration

1. УВОД

Дозирање је процес који се састоји из три основне активности: мерења, транспорта и регулације жељене задате вредности протока. Дозирање зрнастих и прашкастих материјала је специфично због особина материјала тог типа. Наиме, неке особине тих материјала, попут запреминске густине и способности течења, могу да варирају у широком опсегу. Због тога је могуће применити различите поступке дозирања при чему се најчешће примењује дозирање са мерењем масе (тежинско) и без мерења масе (запреминско). Да би се измерила претранспортована количина материјала у неком одређеном временском интервалу, користи се дозирна тракаста вага. Дозирна тракаста вага пружа могућност да се транспортује жељена количина материјала за време задатог временског периода, тј. пружа могућност континуалне регулације протока материјала. Принцип рада се заснива на одржавању константног протока.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чије ментор је био др Драган Живанић, доц.

Тракасти дозатор који је испитиван има мерни ваљчани слог који замењује један ослони ваљак на радној страни транспортера. На крају дозатора се налази мерна посуда у коју упада материјал. Затезање траке дозатора се врши преко навојних вретена на којима су налепљене мерне траке. Предмет експеримента је био утицај затегнутости и брзине траке на тачност дозирања.

2. ОПИС ПРОБЛЕМАТИКЕ

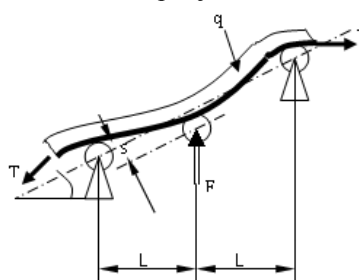
Уређаји за дозирање веома су подложни поремећајима који ометају њихов правилан рад. Долази до одступања приказаних величина протока у односу на стварне што изискује утврђивање тачности информација. Могу се пронаћи различита упутства, тј. конвенције о начину дефинисања величине грешака и метода прорачуна тих грешака. Да би се дефинисање правилно извршило, потребно је утврдити погонско стање и време праћења дозирног уређаја. У стандардима постоје прописи о разграничењу (класа тачности, толеранције итд.) и захтевима градње који подлежу законској овери [1].

Дозирна тачност је максимално дозвољено одступање задате вредности дозирног тока или количине.

Постоји пет врста поремећаја: грешке код одређивања оптерећења траке, грешке код одређивања брзине траке, грешке калибрације, грешке обраде мерних податка, грешке утицаја околине.

2.1. Грешке код одређивања оптерећења траке

Утицаји материјала транспорта су главни узроци грешака. Многи модели траку третирају као еластичну греду при чему се дефинише фактор крутости траке k . Један од утврђених модела је по Колијину, који је дефинисао фактор крутости траке узевши у обзир затезање траке, модул еластичности и површински момент инерције [2].



Слика 1. Приказ оптерећења траке и силе на ваљку

Сила на ваљку се добија према следећем изразу:

$$F = \frac{nqL}{12 \cos \varphi} \pm \frac{2ksT \cos \varphi}{L} \quad (1)$$

Фактор крутости траке:

$$k = \frac{1}{1 - \frac{2}{L \sqrt{\frac{T \cos \varphi}{EI}}} \tan \frac{L}{2} \sqrt{\frac{T \cos \varphi}{EI}}} \quad (2)$$

где су:

q – линијско опт. траке; T – сила затез. у траци;
 L – корак ваљака; n – број мерних ваљака;
 φ – угао нагиба траке; E – модул еластичности;
 s – одступ. мерне ролне; I – повр. момент инерц.

Израчуната грешка (други део израза (1) представља грешку кроз утицај траке) се не додаје увек грешкама ваге. Због одступања (s) може имати делимичан утицај, па служи за оцењивање постојећих или могућих грешака система. Модул еластичности и површински момент инерције се могу приближно одредити према препорукама [2].

Из израза (1) се види како сила затезања у траци има веома значајан утицај на тачност мерења.

2.2. Грешке код одређивања брзине траке

Ова врста грешке се јавља уколико постоји проклизавање траке по погонском бубњу. Да би се остварио прописани проток, дозатор мора да функционише са прописаном устаљеном брзином траке. Међутим, често долази до проклизавања траке услед неадекватне затегнутости, накупљања прљавштине на траци, бубњевима и ваљцима [3].

Због горе наведеног је важно да се у сваком тренутку прати да ли долази до проклизавања траке или не.

3. КОРИШЋЕНА МЕРНА ОПРЕМА

У оквиру ове тачке следи кратак опис мерне опреме која је коришћена током експеримента.

3.1. Аквизициони мерни уређај SPIDER 8 –HBM

SPIDER 8 је електронски мерни уређај који се користи за електрична мерења механичких варијабли (статичких, динамичких) попут силе, притиска, убрзања, температуре, али је у стању да уради и друге задатке.

Уређај је осмишљен као једна целина где су уграђени улази канали, дигитални I/O, комуникациони портови, дигитализација, рачунарски интерфејс.

Рачунарски интерфејс базиран је на CATMAN PRO HBM софтверу, нуди више могућности за прихват, обраду, приказивање, управљање, архивирање података.

3.2. Индуктивни сензор

Индуктивни сензор за мерење броја обртаја погонског бубња, постављен је непосредно изнад ланчаника и сигнал се добија детекцијом наилаaska зуба ланчаника. Повезује се на SPIDER 8, који број обртаја користи као параметар за проверу да ли долази до проклизавања траке.

3.3. Ротациони енкодер

Енкодер је електромеханички уређај који може да мери померај и позицију. Већина енкодера користи оптичке сензоре који дају електричне сигнале у виду импулса који се могу конвертовати у померај, правац или позицију.

Помоћу енкодера се мери брзина траке. Затим се вредности добијене преко енкодера и индуктивног сензора упореде и тако се прати да ли долази до проклизавања траке.

3.4. Давачи силе

Давачи силе или мерне ћелије најбоље се уклапају у многе мерне задатке где су најчешће примењивани у индустрији, јер су издржљиви, имају добре резонантне вредности и дуготрајан циклус примене. Састоје се од мерних трака које су чврсто залепљене за опружни елемент, при чему се најчешће четири мерне траке повезују у пун Винстонов мост.

На тракастом дозатору су постављена четири давача силе и то:

- први је постављен на проточну вагу и ознаке је Z6 FC3/100kg, чије је максимално дозвољено оптерећење 100 kg;
- други се налази испод прихватног суда и мери тежину материјала који се налази у прихватном суду, ознака RSC, S-tipa, 5t;
- трећи и четврти су исти, направљени су дорадом навојних вретена, тако што је део највоног вретена преструган и на том месту су налепљене мерне траке; преко њих се затеже трака транспортера и мере силу у истој.

3.5. Ласерски давач позиције траке SPSR-115/230 Self-Powered Sensor

Ласерски давач позиције траке се користи како би се утврдило колико је потребно времена траци да направи пун круг, у зависности од подешене фреквенције електромотора.

Такође, коришћен је како би мерење увек почињало у истој тачки. Функционише на принципу „једног окидања по обртају“.

3.6. Фреквентни претварач Danfoss VLT 5000 Series-type 5004

Помоћу фреквентног претварача се врши управљање брзином електромотора, а тиме се врши регулација брзине траке дозатора.

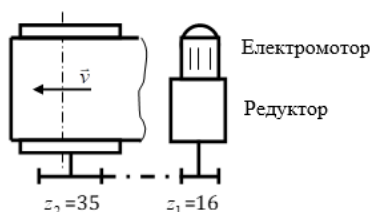
4. КАРАКТЕРИСТИКЕ ТРАКАСТОГ ДОЗАТОРА И ИЗВОДИ ИЗ ПРОРАЧУНА

У оквиру ове тачке су дате основне карактеристике дозатора коришћеног током експеримента, као и основни изводи из прорачуна истог.

4.1. Основни параметри дозатора

Коришћена је трака ширине 540 mm са вођицама висине 70 mm, дужина траке је 6000 mm, коефицијент крутости траке је $k = 125 \text{ N/m}$, број слојева траке је 3, дебљина траке је 8 mm.

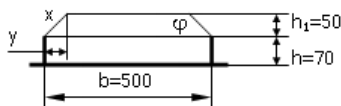
Снага погонског електромотора је $P = 0,75 \text{ kW}$, број обртаја на излазу из редуктора при 50 Hz је $n_{izl} = 28 \text{ o/min}$, док је преносни однос ланчаног преносника $i = 2,1875$.



Слика 2. Шема погона

4.2. Капацитет дозатора

Дефинисан је запремински капацитет за минимални и максимални положај засуна на кошу.



Слика 3. Пресек траке

Проток се рачуна према следећем изразу:

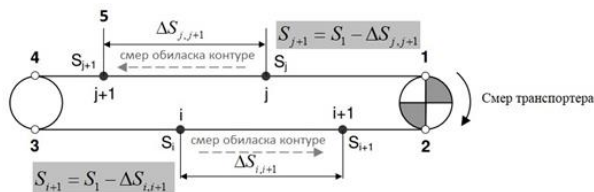
$$Q \left[\frac{\text{m}^3}{\text{h}} \right] = 3600 \cdot A [\text{m}^2] \cdot v \left[\frac{\text{m}}{\text{s}} \right] \quad (3)$$

Добијен је максимални и минимални проток:

$$Q = 3,78 \div 26,54 \frac{\text{m}^3}{\text{h}} \quad (4)$$

4.3. Одређивање потребне силе затезања

Да би се одредила потребна сила затезања, користи се метода обиласка контуре.



Слика 4. Метода обиласка контуре

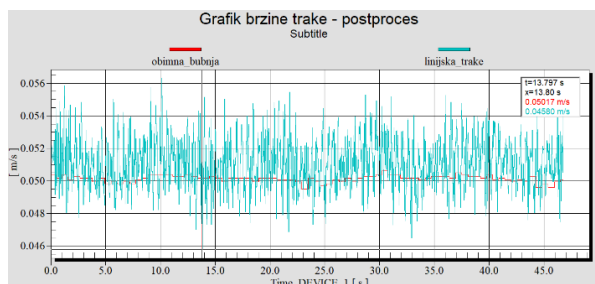
Основна идеја прорачуна снаге погонског механизма транспортера на бази методе обиласка контуре састоји се у утврђивању затезних сила у карактеристичним тачкама транспортера, при чему је пораст затезне силе последица отпора који се појављују између две узастопне карактеристичне тачке. Суштина примене овог приступа заправо је у познавању карактеристичних отпора које уносе поједини уређаји на траси транспортера, познавању утицаја конфигурације трасе транспортера, односно савлађивању висинских разлика, вертикалних и хоризонталних кривина транспортера и слично.

Добијено је да је потребна сила затезања $F_z = 1059,3 \text{ N}$, што је детаљније приказано у мастер раду.

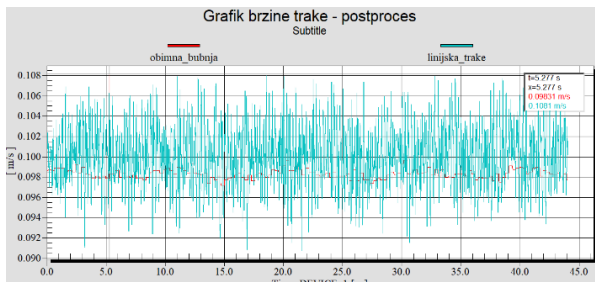
5. АНАЛИЗА КРЕТАЊА ТРАКЕ И КАЛИБРАЦИЈА МЕРНИХ ДАВАЧА СИЛЕ

5.1. Анализа кретања траке

На почетку се вршила провера да ли при раду празне траке долази до проклизавања при 10 и 20 Hz електромотора помоћу индуктивног сензора и енкодера.

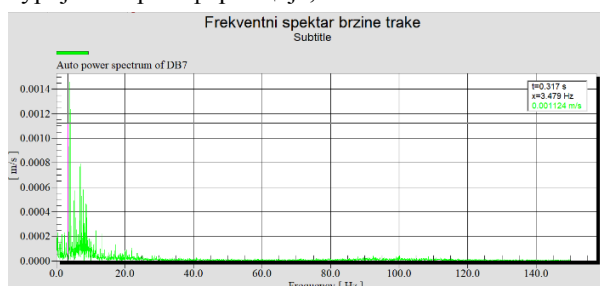


Слика 5. Упоредни приказ линијске брзине траке и обимне брзине бубња на 10 Hz електромотора



Слика 6. Упоредни приказ линијске брзине траке и обимне брзине бубња на 20 Hz електромотора

Утврђено је да се при 10 Hz електромотора трака креће просечном брзином од 0,051 m/s, слика 5, што је 1,5% одступање од теоријске брзине од 0,05025 m/s. При 20 Hz електромотора, трака се креће просечном брзином од 0,1 m/s, слика 6, што је исто 1,5% одступање од теоријске брзине од 0,0985 m/s. Уочена одступања су релативно мала и могу се приписати потенцијално већој вредности пречника погонског бубња од оне која је назначена у техничкој документацији истог или одређених заокруживања вредности при математичкој калкулацији. Због тога је утврђено да не долази до проклизавања при поменутим брзинама. Међутим, уочено је да брзина траке осцилује у облику синусоиде при обе брзине електромотора. Због тога је урађена анализа сигнала у фреквенцијском домену применом алата брзе Фуријеове трансформације, слика 7.



Слика 7. Фреквенцијски спектар временског записа брзине траке при 20 Hz електромотора

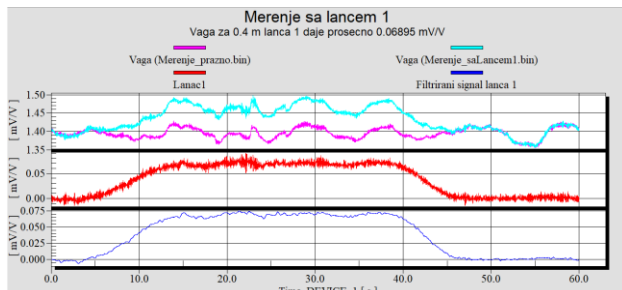
Примећено је да се при 20 Hz рада електромотора, јавља пик на 3,479 Hz, што одговара фреквенцији узубљавања ланца и погонског ланчаника која има исту вредност. Поклапање се јавило и при 10 Hz. Закључено је да до осцилација линијске брзине траке долази због узубљавања ланца и погонског ланчаника.

5.1. Калибрација мерних давача силе

На почетку је обављена калибрација мерне посуде тако што је прво измерен (тариран) сигнал празне посуде.

Затим су у посуду стављани предмети познате масе, добијене вредности су унесене у софтвер и тиме је завршена калибрација мерне посуде.

Следеће је калибрисан мерни давач силе на проточној ваги. Коришћена су два ланца познате масе и дужине. Прво је снимана карактеристика празне траке на мерном давачу, а затим карактеристика са ланцима. Од карактеристике са ланцима је одузет сигнал празне траке и добијен је чист сигнал који је директно пропорционалан маси ланца, слика 8. Тиме су уклоњени утицаји имперфекције траке.

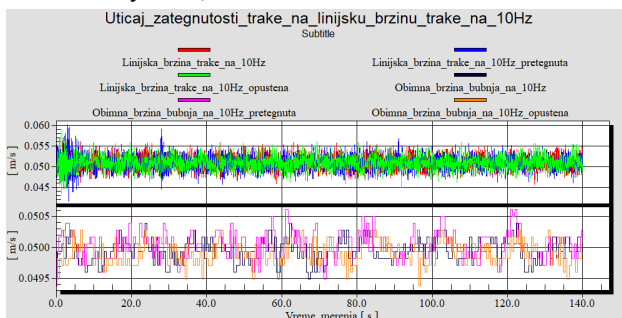


Слика 8. Резултати мерења празне траке и првог ланца

На крају су калибрисани мерни давачи силе на затезним навојним вретенима. Калибрација је вршена на два начина: на кидалици *Toyoseiki AT-L-118B* са мерним распоном од 0 до 100 kg и вешањем познате масе на затезаче.

6. РЕЗУЛТАТИ ЕКСПЕРИМЕНТА

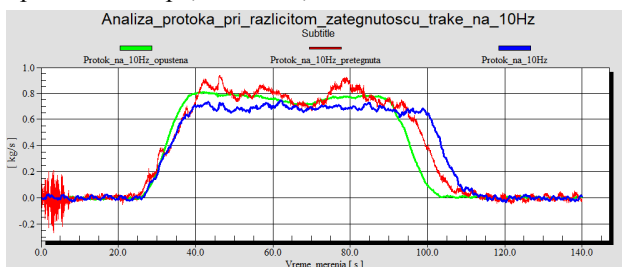
На почетку су упоређени сигнали линијске брзине траке при 10 Hz на различитим нивоима затегнутости траке како би се видело да ли би ту могло да дође до неких одступања, слика 9.



Слика 9. Утицај затегнутости траке на линијску брзину траке на 10 Hz ЕМ

Утврђено је да нема одступања и да може да се настави са испитивањем.

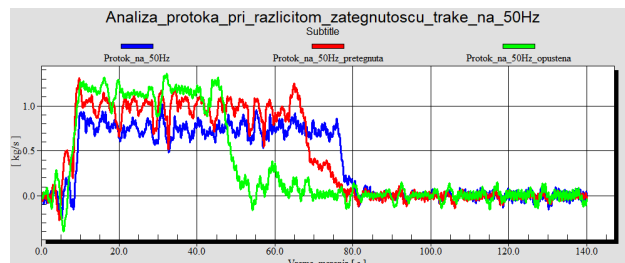
Анализиран је утицај брзине и затегнутости траке на проток дозатора, слике 10, 11 и 12.



Слика 10. Анализа протока при различитом затегнутошћу траке на 10 Hz



Слика 11. Анализа протока при различитом затегнутошћу траке на 20 Hz



Слика 12. Анализа протока при различитом затегнутошћу траке на 50 Hz

7. ЗАКЉУЧАК

Испитивање тракастог дозатора је показало да са променом затегнутости и брзине траке долази до промене протока. Посебно је интересантно како се са порастом наведених параметара мења карактеристика протока, тј. долази до осциловања чија периода и амплитуда директно зависе од испитиваних параметара.

8. ЛИТЕРАТУРА

- [1] В. Васић, „Систематизација машина за паковање, транспортни системи и испитивање тракастог дозатора“, дипломски рад, Факултет техничких наука у Новом Саду, 1999.
- [2] N.G.Müller, „Fehlereinflüsse bei der Massebestimmung eines stetigen Materialstroms auf Förderbandwaagen“, 1988.
- [3] Н. Я. Гроссман и Г. Д. Шнырев, „Автоматизированные системы взвешивания и дозирования“, Машиностроение Москва, 1988.

Кратка биографија:



Никола Иланковић рођен у Суботици 1994. год. Дипломирао 2017. године на Факултету техничких наука, смер машинство, на којем исте године уписује мастер студије на усмерењу за машинске конструкције, транспортне системе и логистику. Запослен као сарадник у настави на Факултету техничких наука у Новом Саду.



Драган Живанић рођен у Сремској Митровици 1972. год. Докторирао је 2012. год, а од 2014. ради као доцент на Факултету техничких наука у Новом Саду.

ВИШЕНАМЕНСКИ ПРЕНОСНИК СНАГЕ ЕКСПЕРИМЕНТАЛНОГ ХИБРИДНО-ЕЛЕКТРИЧНОГ ВОЗИЛА**POWER TRANSFER BOX FOR AN EXPERIMENTAL HYBRID-ELECTRIC VEHICLE**

Милош Голић, Факултет техничких наука, Нови Сад

Област - МАШИНСТВО

Кратак садржај – Овај рад представља предлог за конструисање вишенаменског преносника снаге који је намењен за експериментално хибридно – електрично возило. Првобитно, возило је намењено да се оствари као паралелни хибрид са мотором СУС који погони задње точкове а на предњој осовини точкове погоне два електромотора смештена у самом точку.

Возило има могућност да буде опремљено са трофазним синхроним мотор/ генератором који би пунио тракционе батерије за електро погон предње осовине и тиме би имао особине серијског хибрида а уједно би се повећао број пређених километара по једном пуњењу батерија. Задатак преносника, у овом раду, је да повеже мотор СУС са мотор/ генератором, преко нпр.

ПТО извода са мењача мотора СУС. Један од најважнијих циљева овог рада је развити преносник што мањих димензија и масе са што мањим изворима губитком снаге.

Кључне речи: Серијско–паралелни хибрид, Хибридно–електрично возило, мотор/генератор, Механички преносник, прорачун машинских елемената.

Abstract – This paper presents design proposal for power transfer box intended for an experimental light hybrid-electric vehicle. This vehicle is originally configured as parallel hybrid with mechanical power at the rear axle and two electric hub-motors at the front axle.

Vehicle should subsequently be also equipped with additional electric generator enabling charging of traction batteries for front axle drive, enabling also configuration of series hybrid or range extender. Task of the transfer box described in this paper is to connect IC engine with either rear driving axle or PTO for electric generator, or both.

The goal of design process is to design the transfer box by minimizing the mass and power losses.

Keywords: Serial–parallel hybrid, hybrid–electric vehicle, motor/generator, transmission, machine elements calculation.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био др Борис Стојић, доцент.

1. УВОД

Праћењем савремених начина конструисања погонских склопова хибридних–електричних возила, у Лабораторији за моторна возила Факултета техничких наука у Новом Саду развијено је лако хибридно–електрично возило. Ово возило представља платформу за учење и развој студентских пројеката. Тренутни изглед возила је приказан на слици 1.



Слика 1. Изглед хибридног – електричног возила “Хермес”.

Возило је конструисано по тренутно широко распрострањеним начином изведбе [4] узевши у обзир захтеве које морају испуњавати хибридно–електрична возила [1], [2]. Главна одлика овог возила је његов хибридно–електрични погон. Осмишљен је као паралелан са два електромотора на предњој осовини који су смештени у точковима, док задње точкове погони мотор СУС преко припадајућег мењача, ланчаног пара и диференцијала, изглед предње и задње осовине приказан је на сликама 2 и 3. Паралелна врста погона хибридно–електричног возила означава да предња и задња осовина могу бити погоњене независно једна од друге или упарене зависно од одабира режима. У овом раду описан је развој механичког преносника који би читав систем пренаčio у серијску верзију хибрида али са могућношћу да се користи и као паралелни, као што је првобитно и замишљен. Задатак овог рада је да се развије механички преносник који мора да испуни следеће услове:

- Минималне димензије, мала маса, мали отпори момената инерције обртних маса;
- Минимални извори губитака снаге;

- Минималне измене на возилу ради уградње преносника;
- Могућност уградње актуатора за одабир режима рада на преносник;
- Једноставност израде, олакшано склапање, минимално и лако одржавање.



Слика 2. Предња осовина са електромоторима у точковима и батеријама на предњем препусту



Слика 3. Задња осовина – погон мотор СУС

2. ОДАБИР КОНЦЕПЦИЈЕ ПРЕНОСНИКА

Према задатим критеријумима, преносник мора да повеже и да прекине везу између излазног вратила мењача мотора СУС, ланчаног преноса и мотор/генератора. Због тога, преносник мора да омогући следеће режиме рада:

- Мотор СУС погони диференцијал и точкове;
- Мотор СУС погони мотор/генератор без да погони диференцијал и точкове;
- Мотор/генератор погони диференцијал и точкове;
- Допуна батерија путем регенеративног кочења;
- Мотор СУС погони диференцијал и точкове и истовремено погонимотор/генератор.

3. КОНСТРУИСАЊЕ

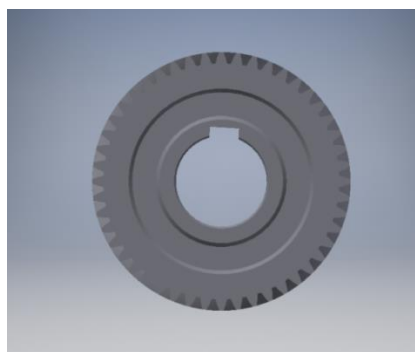
После усвајања концепције преносника, у наредној фази дефинисани су његов облик и димензије путем одговарајућих прорачуна машинских елемената [3]. Прорачуни и цртежи су изведени употребом софтверског пакета "Autodesk Inventor 2016 – Student version".

3.1. Прорачун зупчаника

Прорачун зупчастог пара је изведен по важећем светском стандарду из те области ISO 6336:1996. Претпостављено је да генератор допуњује батерије са 1500o/min и да је на улазном вратилу преносника расположива снага од 10kW, док је број обртаја који је на улазном вратилу израчунат, и износи 1825o/min. Подаци о зупчастом пару дати су у табели 1.

Табела 1. Параметри зупчаника

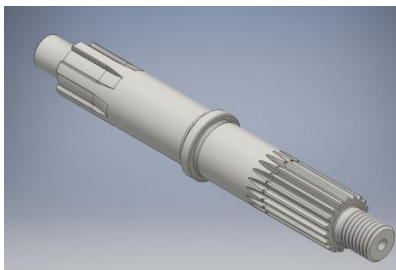
	Улаз	Излаз
Број зуба	40	48
Ширина [mm]	22	20
Пречник [mm]	59.690	71.078
Модул [mm]	1.375	
Угао закошења зуба [deg]	15	
Материјал	37Cr4	



Слика 4. Приказ гоњеног зупчаника

3.2. Прорачун вратила

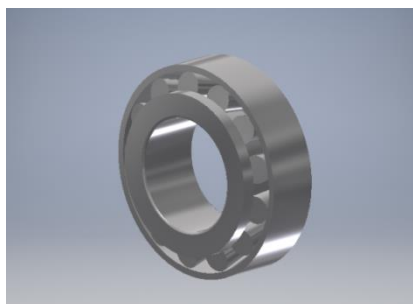
Одабир материјала вратила се не разликује много од одабира материјала за зупчасти пар. Вратила на зупчастим паровима су оптерећена радијалном силом, аксијалном силом и обртним моментом, док је вратило које погони ланчаник, оптерећено само обртним моментом. За потребе прорачуна узета су сва оптерећења на вратилу као улазни подаци. Изглед једног од вратила дат је на слици 5.



Слика 5. Погонско, улазно, вратило преносника

3.3. Прорачун лежајева

Прорачун лежајева је такође изведен у наведеном софтверском пакету. Пошто су усвојени прорачунати лежајеви производ немачког предузећа “SKF”, прорачун тих лежајева је проверен на интернет сајту произвођача. Лежајеви који су усвојени су коничног типа који се уграђују у преноснике са косим зубима на зупчаницима, отпорни су на дејство аксијалне силе, и захтевају прецизну уградњу.



Слика 6. Конични лежак “DIN 720 SKF- Precision (SKF 32005 X/Q)”

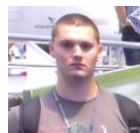
4. ЗАКЉУЧАК

У раду је приказан поступак развоја вишенаменског преносника за погон хибридно-електричног возила. Основне функције разводника су повезивање мотор/генератора са погонском јединицом мотора СУС и тиме су остварени сви потребни режими рада хибридно-електричног возила. Коришћени су поступци инжењерских прорачуна зупчастог пара, лежајева и вратила по тренутно важећим стандардима коришћењем софтверског пакета “Autodesk Inventor 2016 - Student Version”. Идеја је била да се конструише преносник компактних димензија како би се избегле велике измене на возилу. Усвојена концепција конструисаног преносника је оптимална са аспекта димензија, масе и једноставности и осим хибридно-електричног возила може бити употребљена и за конструисање разводника погона за теренска возила са погоном на две осовине. Повезивање два саосна вратила је изведено по угледу на аутомобилски мењач са три вратила.

5. ЛИТЕРАТУРА

- [1] Fuhs A 2009 *Hybrid Vehicles and the Future of Personal Transportation*, CRC Press Taylor & Francis Group
- [2] Hodkinson R and Fenton J 2001 *Lightweight Electric/Hybrid Vehicle Design*, Butterworth-Heinemann
- [3] Kutz M (editor) 2005 *Mechanical Engineers' Handbook*, John Wiley & Sons
- [4] Weber J 2009 *Automotive Development Processes*, Springer

Кратка биографија



Милош Голић -рођен је у Новом Саду 1993. године. Мастер рад на Факултету техничких наука из области Машинства – Аутомобилско инжењерства одбранио је 2018. године. Контакт: milosgolic1@gmail.com

SOFTVER ZA POKRETANJE, ZAUSTAVLJANJE I OPORAVAK DISTRIBUIRANIH SERVISA**SOFTWARE FOR DISTRIBUTED SERVICES STARTUP, SHUTDOWN AND RECOVERY**

Stanko Polić, *Fakultet Tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Ovaj rad se bavi problematikom distribuiranih operativnih sistema. Rad pokriva pitanja sistemskog “startup”-a i “shutdown”-a redundantnog HA (High Availability) klastera. Detaljno je objašnjen sistemski “startup/shutdown” servisa, pisanih “Python” programskim jezikom.

Abstract – This paper deals with the problems of distributed operating systems. The thesis covers the notion of system startup and shutdown redundant HA (High Availability) clusters. A system startup/shutdown of services, written in the Python programming language, is explained in detail.

Ključne reči: Softversko inženjerstvo, Distribuirani operativni sistemi, HA klasteri, Server-klijent model, Sistemski startup/shutdown.

1. UVOD

U distribuiranim računarskim sistemima, odnosno njihovim nodovima, potrebna je kontrola i nadzor nad servisima, njihovim stanjem i osnovnim funkcionalnošću. Radi što boljeg praćenja statusa i održavanja bazičnih stanja servisa, potrebno je razviti skripte, koje omogućavaju pokretanje i zaustavljanje servisa sistema po želji i potrebi korisnika. Unutar HA (High Availability) klastera, najvažnije bazične servisne osobine su *startup/shutdown* i *failover*, kao osnovne operacije, koje obezbeđuju punu operativnu funkcionalnost celog sistema.

U poglavlju 2. dat je opis osnova operativnih sistema, kao i njihovog razvoja. Navedene su osobine distribuiranih operativnih sistema i njihove podele, a dat je i opis klasterne strukture distribuiranih sistema, sa akcentom na mikrokernelski klijent-server model. Opisana je i bazična funkcionalnost HA klastera, koja se ogleda u pravilnom manipulisanju operacijama pokretanja i zaustavljanja servera na nodovima klastera.

U poglavlju 3. dat je pogled na jedan HA dualni redundantni kluster. Opisana je njegova arhitektura, naznačeni su glavni delovi i ključni serveri. Detaljno je opisan Inicijalni sistemski *startup/shutdown*, kao i glavni Sistemski *startup/shutdown*.

Poglavlja 4. i 5. sadrže zaključna razmatranja i pregled korišćene literature.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Miroslav Hajduković, r.prof.

2. OPIS REŠAVANOG PROBLEMA**2.1 Pojam operativnog sistema**

Glavni cilj operativnih sistema je da obezbede apstrakciju mašine (proširena ili virtuelna mašina) predstavljenu kao skup servisa. Servisi su implementirani kao skup softverskih modula, koji čine interfejs između programa i sistemskog hardvera. Funkcije operativnog sistema kao upravljača resursima su sledeće [1]:

- raspoređivanje procesa,
- sinhronizacija procesa i međuprocena komunikacija,
- upravljanje uređajima i komunikacijama,
- upravljanje memorijom,
- upravljanje datotekama.

2.2 Razvoj operativnih sistema

Razvojem i velikom ekspanzijom umrežavanja, javljaju se Mrežni operativni sistemi i Distribuirani operativni sistemi. U mrežnim operativnim sistemima korisnici su svesni postojanja više mašina (računara), ali ovakav vid umrežavanja omogućava međusobnu razmenu programa i podataka među mašinama. Distribuirani operativni sistem menja pogled na sistem sastavljen od više računarskih jedinica (računari su spojeni u mrežu, i korisnik stiče privid uniformnosti). Ovakav sistem nazivamo **transparentnim** sistemom.

2.3 Distribuirani operativni sistemi

Distribuirani računarski sistem (*Distributed Computer System*) nastaje zbog potrebe razmene podataka komunikacionim linijama među međusobno udaljenim računarima. Svaki od računara, povezan u distribuirani računarski sistem, je neophodno da sadrži: procesor, radnu memoriju i mrežni kontroler. Operativni sistem, prisutan na svakom računaru u distribuiranom računarskom sistemu, nije obavezan da podržava masovnu memoriju, jer ona nije prisutna u svakom od računara. Ovakav operativni sistem se naziva mikrokernel (*microkernel*), jer ima smanjenu funkcionalnost u odnosu na “običan” operativni sistem. Distribuirani operativni sistemi treba da obezbede istu uniformnu sliku umreženih mašina, koje mogu biti prostorno udaljene. Posledica je da se identičan kernel izvršava na svim mašinama u sistemu, i da se poštuje isti komunikacioni protokol. Distribuirani operativni sistem od više jeftinih i malih računara obrazuje multiračunarski sistem, koji je jeftiniji, ali opet moćniji od jednog velikog i skupog računara.

Radi najbržeg usluživanja korisnika (najkraćeg vremena odziva), važno je da se obezbedi automatsko upravljanje svim resursima distribuiranog računarskog sistema, kao i optimalno raspoređivanje procesa po procesorima.

2.4 Koristi od distribuiranih sistema

Cilj distribuiranih operativnih sistema je da se što efikasnije koriste računarski resursi i to na sledeće načine:

- deljenjem resursa među pojedinim mašinama distribuiranog sistema,
- raspoređivanjem procesorskog opterećenja između više različitih mašina,
- povećanjem pouzdanosti, odnosno, poslovi koji su se izvršavali na pokvarenoj mašini mogu biti preuzeti od strane druge mašine [1].

2.5 Cluster

Klaster računar (*Cluster*) je skup više računara (*nod-ova*) povezanih brzom mrežom, koji predstavljaju jedinstvenu celinu. Klaster računar može biti jednostavni dvo-čvorni system, koji samo povezuje dva personalna računara, ili može biti veoma brz superkompjuter.

Klasteri mogu ostvariti osobinu jedinstvenosti na tri nivoa:

1. hardvera,
2. operativnog sistema,
3. aplikacije (klasterski *middleware*).

Visoko-dostupni klasteri (poznati i kao *FAILOVER* klasteri, ili HA klasteri) poboljšavaju dostupnost klaster pristupa. Oni imaju redundantne čvorove (računare), koji se koriste kada osnovne komponente sistema otkazu. HA klaster koristi redundantnost tako što umesto pokvarenog čvora u sistemu, koristi rezervni čvor.

Redundantnost je dupliranje kritičnih komponenti ili funkcija sistema, sa namerom povećanja pouzdanosti i uvećanja performansi sistema, uobičajeno u formi *backup-a* ili *failover-a* [2].

2.6 Servisi

Postoje dva osnovna pristupa proširenja servisa konvencionalnih operativnih sistema, da bi se oni mogli koristiti za potrebe distribuiranih operativnih sistema: **Monolitni kernel** (baziran je na centralizovanom operativnom sistemu) i **Mikrokernelski pristup** (zasniva na ideji da se obezbede samo neophodni servisi kernela).

2.7 Klijent – Server model

Osnovna ideja klijent - server modela je da se napravi kooperacija procesa, koji obezbeđuju servise (serveri), i procesa (klijenti), koji servise zahtevaju. Server se može ponašati i kao klijent, ukoliko zahteva servis od drugog servera, da bi izvršio svoj zadatak. Komunikacija klijenata i servera obuhvata razmenu zahteva i odgovora:

- klijent šalje zahtev serveru i potom se blokira,
- server prima zahtev, obavlja traženu radnju, i vraća odgovor,
- klijent prima odgovor i nastavlja izvršavanje.

Problem kako klijent locira server, ako on promeni lokaciju, rešava se tako što se serveri identifikuju preko imena. Posebni serveri, nazvani vezujući serveri (*binder servers*) spajaju klijente i servere. Procedura, koja izvršava ovo spajanje, je nazvana dinamičko povezivanje (*dynamic binding*).

2.8 Startup/Shutdown koncept

U redundantnim HA klusterima, glavna osobina sistema je postojanje više čvorova. Kada čvor u klasteru otkaze,

koristi se strategija kao što je *fencing* da bi ostatak sistema ostao operativan. *Fencing* je proces izolovanja čvora ili zaštite deljenih resursa, kada se pojavi pokvaren čvor. Klasteri Visoke-dostupnosti (HA klasteri) se često koriste za balansiranje opterećenja sistema, pravljenja rezervne kopije, kao i slučaju pada dela sistema ili kvara. U slučaju ispada jednog noda, HA kluster je dužan da održi aktivnost sistema, kao i održi bazičnu funkcionalnost celokupnog sistema. U slučajevima redovnog održavanja sistema, kvara u sistemu, izmene u arhitekturi klastera, potrebno je servise sistema uspešno zaustaviti i zatim ih nakon završenih predviđenih operacija, ponovo uspešno pokrenuti. Stoga, *startup* i *shutdown* procedure imaju veoma važnu ulogu u ispravnom funkcionisanju servisa klastera.

3. OPIS REŠENJA PROBLEMA

Sistemske startovanje je proces startovanja/prelaženja RVDS (Redundantni Visoko Dostupni Sistem) servera u operativno stanje, odnosno, u stanje gde klijent može pristupiti i zatim upotrebiti RVDS servise (oni mogu biti u *HOT* i *STANDBY* stanju).

Postoje dva tipa *startup* procedure: Inicijalni *startup* (pokretanje inicijalnih servera na serverskoj mašini unosa podataka) i potpuni RVDS sistemski *startup* (pokretanje svih instaliranih servera na redundantnoj serverskoj mašini). Inicijalni serveri su komponente RVDS Sistema, potrebne za postavljanje inicijalnih podataka i fajlova na redundantnim serverskim mašinama RVDS-a.

3.1 US (Upravljaajući Server)

US (Upravljaajući Server) je odgovoran za pripremanje RVDS sistema za pokretanje (*startup*), zatim je zadužen da izvrši proveru svih RVDS servera u paraleli, i nakon toga da ih pokrene u određenom redu.

3.2 NS (Nadgledajući Server)

NS (Nadgledajući Server) je odgovoran za arbitraciju sistemskih stanja servera. NS postavlja servere u određenom nodu u sledeća stanja: *HOT* ili *STANDBY*.

3.3 Koraci pri Inicijalnom startup-u

Inicijalni *Startup* je prva faza RVDS sistemskog *startup-a*. On se izvršava na mašini za unos podataka klastera.

Tokom ove faze treba da budu izvršeni sledeći koraci:

- Inicijalni *Startup* proverava da li su već startovani RVDS servisi.
- Nakon toga startuje se US proces i poziva se startovanje CORBA interfejsa.
- US server ulazi u određeno i unapred predviđeno stanje (*READY* stanje), i potom US startuje SO (Server Obrasca).
- SO ulazi nakon toga u određeno i unapred predviđeno stanje (*READY* stanje).
- US loguje da su Inicijalni serveri uspešno startovani, ali ukoliko nisu zadovoljeni svi uslovi, poziva se Inicijalni *Shutdown*.

3.4 RVDS sistemski startup

Tokom ove faze, svi sistemski RVDS serveri treba da budu startovani. RVDS sistemski *startup* može se predstaviti sledećim koracima:

- RVDS sistemski *startup* proverava da li su podignuti već RVDS sistemski serveri
- US proces se startuje i poziva se startovanje CORBA interfejsa.
- US na startovanom nodu proverava da li je startovan US proces na drugom nodu:
 - Ukoliko je US startovan na dugom nodu, proverava se da li se modeli na oba noda podudaraju u verzijama: modela šeme, instalacionim verzijama i verzijama *Update-a*.
 - US takođe proverava da li se modeli na oba noda poklapaju u sadržajima: *checksums* binarne šeme, *checksums RSConf.xml* fajla, *checksums Options.xml* fajla i drugih važnih konfiguracionih fajlova.
 - US prekida poretanje i prijavljuje (loguje) grešku, ukoliko neki od navedenih koraka ne prođe proveru.
- US server ulazi u određeno i unapred predviđeno stanje (*READY* stanje).
- Poziva se US “*system startup*” interfejs.
- US čeka da svi RVDS serveri uđu u inicijalno stanje.
- US poziva startovanje CORBA interfejsa i čeka da svi server uđu u *READY* stanje
- NS startuje proces arbitracije. Nakon arbitracije, RVDS serveri su ili u **HOT**, ili u **STANDBY**.
- NS nakon arbitracije nastavlja monitoring svih servera.
- US prijavljuje (loguje) da su svi serveri uspešno pokrenuti, ili se poziva *shutdown* procedura.

3.5 RVDS sistemski shutdown

RVDS sistemski *shutdown* je proces zaustavljanja RVDS servera. Postoje dva tipa RVDS sistemskog *shutdown-a*, jedan na UP Server mašini – Inicijalni *shutdown* i RVDS sistemski *shutdown* svih servisa. Inicijalni *shutdown* stopira US i SO server, kao i *omniNames*.

RVDS sistemski *shutdown* može biti iniciran jedino ukoliko je sistem ili u **HOT** ili u **STANDBY** stanju.

3.5.1 RVDS sistemski shutdown koraci

Tokom ove procedure, svi RVDS sistemski servisi biće stopirani u sledećem redosledu:

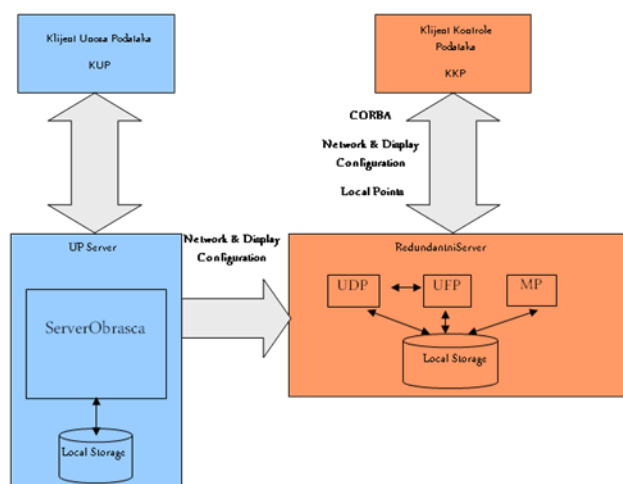
- Ukoliko mašina pripada redundantnom serverskom **HOT** nodu, izvršiće se prvo *FAILOVER* procedura i nakon toga će sistem biti ispravno zaustavljen. RVDS sistemski *shutdown* proverava da li je US pokrenut, i ukoliko nije, prekida procedure.
- Poziva se “*system shutdown*” CORBA interfejs US-a.
- Nakon toga, US prijavljuje (loguje) da je *shutdown* iniciran.
- US poziva u određenom i unapred predviđenom redosledu *shutdown* interfejs svih RVDS servera, i nakon toga proverava da li su svi RVDS serveri zaustavljeni i prijavljuje (loguje) grešku ukoliko nisu.

RVDS Sistem-ArHITEKTURA

RVDS sistem se sastoji od dva podsistema:

1. podsistem **Klijent Unosa Podataka (KUP)**
2. podsistem **Klijent Kontrole Podataka (KKP)**.

ArHITEKTURA RVDS sistema se može videti na Slici 1.



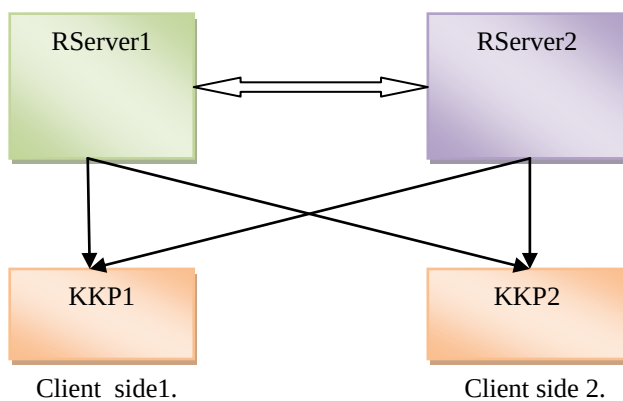
Slika 1. RVDS sistem

3.6 Klijent Kontrole Podataka RVDS sistema

Sledeće komponente čine Klijenta Kontrole Podataka: Nadgledajući Server, Upravljučujući Server, kao i drugi serveri.

Podsistem Dualne Kontrole Podataka se sastoji iz **HOT** i **STANDBY** dela (*node-a*). Klijent Kontrole Podataka može biti operativan i sa aktivnim samo jednim *nod-om*, međutim tako se narušavaju ključne osobine i prednosti dualnih serverskih sistema. Podsistem Dualne Kontrole Podataka se može videti na Slici 2.

Podsistem Dualne Kontrole Podataka STANDBY HOT



Slika 2. Dualni KKP podsistem

3.7 Startup inicijalnih servera na UP mašini

Inicijalni *startup* baznih servera se odigrava na UP mašini RVDS sistema. Sledeći koraci predstavljaju Inicijalni *startup*:

1. Inicijalni *Startup* proverava da li su već startovani RVDS serveri, i ako su serveri startovani, treba da obustavi *startup* i da prijavi grešku.
2. Inicijalni *Startup* startuje US proces i poziva startovanje CORBA interfejsa.
3. Inicijalni *startup* čeka da US server uđe u određeno i unapred predviđeno stanje (*Ready* stanje).
4. US startuje SO.
5. Inicijalni *startup* čeka da SO uđe u određeno i unapred predviđeno stanje (*READY* stanje).
6. Ukoliko su svi uslovi zadovoljeni, US loguje da su Inicijalni serveri uspešno startovani.

3.8 Sistemski *Startup* RVDS sistem

1. RVDS sistemski *startup* proverava da li su podignuti već RVDS sistemski serveri, i ukoliko je bilo koji servis startovan, prekida *startup* i prijavljuje (loguje) grešku. RVDS sistemski *startup* prvo startuje US proces i poziva startovanje CORBA interfejsa.
2. US proverava da li je startovan US proces na drugom nodu. Ukoliko jeste startovan, US proverava da li se modeli na oba noda podudaraju u verzijama modela šeme, instalacionoj verziji i verziji *Update-a*. US proverava i da li se modeli na oba noda poklapaju u sadržajima *checksum-a*: binarne šeme, *RSConf.xml* fajla i *Options.xml* fajla. Ukoliko bilo koji od navedenih koraka ne prođe proveru, US prekida startovanje i prijavljuje (loguje) grešku.
3. RVDS sistemski *startup* čeka da US server uđe u predviđeno (*READY*) stanje.
4. US izvršava sve RVDS serverske procese simultano i čeka da svi serveri uđu u inicijalno stanje.
5. Kada su svi serveri u inicijalnom stanju, US poziva startovanje CORBA interfejsa u unapred predviđenom redosledu i čeka da svi serveri uđu u *READY* stanje.
6. Nakon što je startovan NS, on startuje proces arbitracije. Posle završenog procesa arbitracije, RVDS serveri su ili u *HOT* stanju ili u *STANDBY* stanju, i NS počinje njihov monitoring.

3.9 Inicijalni RVDS *shutdown*

Inicijalni *shutdown* stopira US i SO kao i *omniNames*. Inicijalni *shutdown* se startuje na UP mašini RVDS sistema.

3.10 Sistemski RVDS *shutdown*

RVDS sistemski *shutdown* može se predstaviti sledećim logičkim koracima:

1. Ukoliko je sistem dualan, i ukoliko mašina pripada RS serverskom *HOT* nodu, tada nastupa *failover* procedura, nakon toga će sistem biti ispravno zaustavljen.
2. RVDS sistemski *shutdown* proverava da li je US pokrenut, i prijavljuje (loguje) grešku i prekida procedure ukoliko nije.
3. RVDS sistemski *shutdown* poziva "system shutdown" CORBA interfejs US-a.
4. US poziva *shutdown* interfejs svih RVDS servera (uključujući NS, koji je prvi na listi) u određenom i unapred predviđenom redosledu.
5. US proverava da li su svi RVDS serveri stopirani i prijavljuje (loguje) grešku ukoliko nisu (situacija se može dogoditi ukoliko serveri ne odgovaraju na pozive).

4. ZAKLJUČAK

U ovom radu opisano je rešenje koje povećava efikasnost, pouzdanost i robusnost distribuiranog računarskog sistema. Ono se zasniva na pristupnosti paralelnih struktura, kako softverskih tako i hardverskih kapaciteta sistema. Primenjena je *HOT and STANBY* metoda redundancije u kojoj primarni i sekundarni (npr. *backup*) sistemi rade simultano. Podaci se prebacuju u sekundarni server u *real time-u*, tako da oba sistema sadrže identične podatke.

Problem sistemskog *startup-a* i *shutdown-a* je predmet pažnje ovoga rada. Rešavanje ovoga problema se zasniva na pravilnom i redoslednom pokretanju, odnosno zaustavljanju, servera na nodovima klastera. Pravilan *startup* i *shutdown* doprinose povećanoj bezbednosti, očuvanju podataka i produžetku operativnog rada celokupnog klastera u periodu eksploatacije i održavanja.

Postupci *startup-a* i *shutdown-a* servisa na nodovima klastera su veoma važni procesi zbog očuvanja podataka, njihove bezbednosti i njihove distribucije kroz ceo sistem. Veoma su važni i prilikom redovnog održavanja sistema, *update-a* sistema (kako softverskog tako i hardverskog), selidbe celog ili delova sistema, jer je potrebno u realnom vremenu, ne ometajući funkcionalnost sistema, ili držeći funkcionalnost sistema prekidu u što kraćem vremenskom roku, održati sve potrebne sistemske procese aktivnim, održati celokupan sistem operativnim što duže, a pritom što jednostavnije i što kvalitetnije.

Pravci daljeg razvoja

Dalji pravci razvoja i unapređenja realizovanih sistemskih *startup/shutdown* procedura su *remotely* upravljanje klasterom. To znači, da se celokupno upravljanje obavlja sa jednog mesta (*control room*), odakle se može sistem podizati i spuštati po potrebi, kako zahtevaju predviđeni koraci korišćenja sistema.

Potrebno je unaprediti *user friendly* okruženje, tako da se može startovati GUI aplikacija koja će imati pobrojane sve mašine klastera (sve servere, RS i UP mašine sistema).

5. LITERATURA

- [1] B. Radenković; "Distribuirani operativni sistemi"; FON
- [2] Michel Daydé, "High Performance Computing for Computational Science - VECPAR 2004"; Jack Dongarra 2005 ISBN 3-540-25424- 2

Kratka biografija:



Stanko Polić rođen je 1983. godine u Novom Sadu, Republika Srbija. Završio je srednju školu Gimnazija Jovan Jovanović Zmaj u Novom Sadu. Akademske studije, na Fakultetu tehničkih nauka u Novom Sadu, je upisao 2002. godine. Ispunio je sve obaveze i položio je sve ispite predviđene studijskim programom.

PODRŠKA VIZUALIZACIJI STRUKTURE I BOJENJU KODA U VISUAL STUDIO CODE EDITORU ZA JEZIKE BAZIRANE NA TEXTX ALATU**A SUPPORT FOR CODE OUTLINE AND HIGHLIGHTING IN VISUAL STUDIO CODE EDITOR FOR LANGUAGES BASED ON TEXTX TOOL**

Nemanja Starčev, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu su implementirani jezici za podršku vizualizaciji strukture i bojenju koda u Visual Studio Code alatu za jezike napisane u textX-u.

Abstract – This work presents implementation of a languages for support for code outline and highlighting in Visual Studio Code editor for languages based on textX tool.

Ključne reči: bojenje koda, struktura koda, textX, JSD, JON, skener, parser, Visual Studio Code

1. UVOD

U ovom radu prikazani su jezici za bojenje koda i prikaza strukture koda (eng. outline) u Visual Studio Code editoru za jezike napisane u textX-u. Bojenje koda je osobina editora da prikaže programski kod u boji da bi programer mogao lakše napisati i analizirati kod. Najčešće se boje ključne reči, brojevi, stringovi i operatori. Struktura koda predstavlja prikaz najvažnijih delova koda u obliku stabla unutar zasebnog prozora. Stablo se sastoji od čvorova. Svaki čvor ima naziv, ikonicu i eventualno decu (druge čvorove). Klikom na čvor, biće selektovan deo koda koji odgovara tom čvoru. TextX je softver otvorenog koda koji omogućava kreiranje jezika specifičnih za domen (JSD). Proces konstrukcije JSD prolazi kroz nekoliko faza: kreiranje sintakse, semantička analiza, transformacije, generisanje koda i integracija jezika sa drugim alatima [1]. Sintaksa textX-a je pogodna za kreiranje sintakse drugih jezika, semantička analiza je moguća u kombinaciji sa Python-om, textX obavlja različite vrste transformacija, generisanje koda je moguće pomoću nekog od obrađivača šablona u Python-u.

Međutim integracija jezika sa drugim alatima je veoma teška ili nemoguća. Jezici razvijeni u ovom master radu omogućavaju laku integraciju jezika napisanih u textX-u i Visual Studio Code alata. Integracija se ogleda u vidu podrške za bojenje koda i prikaza strukture koda. Podrška se obavlja generisanjem koda i konfiguracionih fajlova koji predstavljaju proširenje za Visual Studio Code.

Prema istraživanjima sajta „StackOverFlow“, Visual Studio Code je najpopularnije okruženje za razvoj softvera [2]. Svoju popularnost duguje veoma jednostavnom načinu za kreiranje proširenja. Zbog toga smo se odlučili da novo-kreirane jezike integrišemo sa Visual Studio Code alatom.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Igor Dejanović, vanredni prof.

2. TEORIJSKE OSNOVE

Programske jezike možemo podeliti u 5 generacija: mašinski, asemblerske, proceduralne, domenski specifične i deklarativne. Prve dve generacije programskih jezika predstavljaju programske jezike nižeg nivoa, dok poslednje tri programske jezike višeg nivoa. Kod programskih jezika nižeg nivoa neophodno je da programer bude upoznat sa arhitekturom mašine na kojoj programira. Tokom programiranja, programer razmišlja na nivou arhitekture naredbi, što odvlači pažnju od suštine problema i tako nastaju greške. Ovaj nedostatak nižih programskih jezika prevazilaze viši programski jezici. Kod njih programer tokom programiranja može u potpunosti da se fokusira na programski kod zanemarujući arhitekturu računara [3].

Jezici specifični za domen (JSD) omogućavaju rešavanje problema unutar domena. Njihova sintaksa je veoma bliska ljudskom načinu razmišljanja. Zbog toga, često ljudi koji nisu programerski obučeni mogu programirati. Njihova jednostavnost odgleda se u tome što korisnik ovih jezika treba samo da kaže šta treba da se obavi ali ne i kako. JSD imaju prednosti ali i nedostatke [4]. Prednosti su: povećana produktivnost programera, bolji kvalitet softvera, validacija i verifikacija su izražene konceptima iz domena, integritet modela se ne narušava prilikom prelaska na novu tehnologiju, razdvajanje bitnog od nebitnog, moguća angažovanost domenskog eksperta tokom razvoja, platformska nezavisnost. Nedostaci su: izgradnja JSD zahteva napor i vreme, neophodno iskustvo programera, nedovoljno znanje iz oblasti domena od strane tvorca JSD, održavanje, zbog velikog broja JSD moguće dupliranje funkcionalnosti ili čitavog JSD, razlike kulturne prirode.

JSD se sastoje od: konkretne sintakse, apstraktne sintakse i semantike [5]. Konkretna sintaksa predstavlja sam izgled jezika. Koristi je čovek kako bi kreirao programski kod. Apstraktne sintaksa definiše strukturu jezika. Semantika definiše značenje jezičkih iskaza. Najčešće se semantika zadaje prevodenjem na jezike opšte namene (JON) koji se kasnije prevode na niže programske jezike sve do mašinskog koji može da se izvrši na računaru.

Prema načinu implementacije, JSD mogu biti interni ili eksterni [5]. Interni JSD preuzimaju kompletnu infrastrukturu domaćina. Pošto su iskazani konceptima opšte namene, koriste se isključivo za potrebe programera. Može se reći da je biblioteka svakog programskog jezika interni JSD. Prilikom razvoja eksternih JSD potrebno je dizajnirati sintaksu jezika. Za njihov razvoj najčešće koristimo alate kao što su Arpeggio ili textX. Zbog

potrebe konstruisanja sintakse eksternog JSD, njihova izgradnja zahteva više truda i vremena, ali zato su lakši za korišćenje.

Prema vrsti konkretne sintakse, JSD možemo podeliti na tekstualne, grafičke, tabelarne, itd. [5]. Kod tekstualnih JSD, program se razvija tako što korisnik piše tekst koji predstavlja programski kod u editoru. Karakteriše ih mali troškovi za konstrukciju, jer ne zahtevaju razvoj novog editora, niti sistema za praćenje verzija. Konstrukcija programa kod grafičkih JSD odvija se tako što korisnik bira određen entitet ili strukturu iz palete, i zatim je prevlači na radnu površinu. Za ove jezike je karakterističan veoma moćan sistem validacije, jer onemogućava korisnike da kreiraju nevalidne strukture. Konstrukcija grafičkih JSD zahteva konstrukciju specijalizovanih editora prilagođenih određenom JSD. Konstrukcija grafičkih JSD zahteva više vremena za razvoj, a time i veće troškove. Ako su korisnici eksperti koji nemaju dodira sa programiranjem, ili nisu bliski sa domenom, pa im je potrebno ponuditi skup koncepata koje mogu koristiti, tada je najbolja grafička reprezentacija. Rad sa tabelarnim JSD određen je vršenjem različitih operacija nad tabelom. Najbolji su izbor ako nam je potreban prikaz sortiranih podataka.

3. TEHNOLOGIJE

Kreiranje novog DSL obavlja se pomoću *textX* alata i *Python* programskog jezika. Nakon što programer kreira jezik, moguće je integrisati novo-kreirani jezik sa *Visual Studio Code* alatom. Integracija se obavlja generisanjem odgovarajućih fajlova pomoću *jinja2* obrađivača šablona.

Visual Studio Code je editor programskog koda razvijen od strane *Microsoft*-a za *Windows*, *Linux* i *MacOS*. Za razliku od *Visual Studia* koji je baziran na „*solution*“ principu organizacije fajlova, *Visual Studio Code* je baziran na principu foldera i fajlova. Napisan je u *typescript* jeziku. Prva verzija objavljena je u aprilu 2015. Razvoj alata je i dalje aktivan. Sadrži podršku za debagovanje, git kontrolu, bojenje, prikaz strukture koda, dopuna koda, refaktorisanje, navigaciju, itd. Njegova veoma važna osobina je mogućnost kreiranja proširenja. Sva proširenja se smeštaju u *.vscode* fajl koji se kreira prilikom instalacije alata. Proširenje se prepoznaje na osnovu *package.json* fajla. Ovaj fajl sadrži meta-podatke o ekstenziji poput autora, izdavača, dodatnih prozora, putanje do fajla koji predstavlja ulaz u *typescript* kod i putanje do fajla koji sadrži *JSON* objekat koji predstavlja uputstvo kako izvršiti bojenje koda. *JSON* objekat se sastoji od šablona (eng. *pattern*) i klase. Šablon predstavlja regularni izraz. Svakom šablonu dodeljujemo jednu klasu iz *TextMate* specifikacije. Svaka klasa predstavlja jednu boju za jednu temu. Svaka pojava regularnog izraza biće obojena odgovarajućom bojom. Boja će se odrediti na osnovu *TextMate* klase i trenutno podešene teme unutar alata. Kreiranje stabla koji predstavlja strukturu koda obavlja se pomoću *Visual Studio Code namespace API* biblioteke. Ova biblioteka sadrži klase koje omogućavaju prikazivanje čvorova stabla unutar zasebnog prozora, registrovanje osluškivača (eng. *listener*-a), selektovanje delova koda, itd. *Visual Studio Code* poziva komandu na *language serveru*-u, a kao rezultat komande dobija *JSON*

objekat koji predstavlja stablo. Prolaskom kroz ovaj objekat i instanciranjem klasa pomoću *Visual Studio Code namespace API* biblioteke prikazujemo stablo u prozoru.

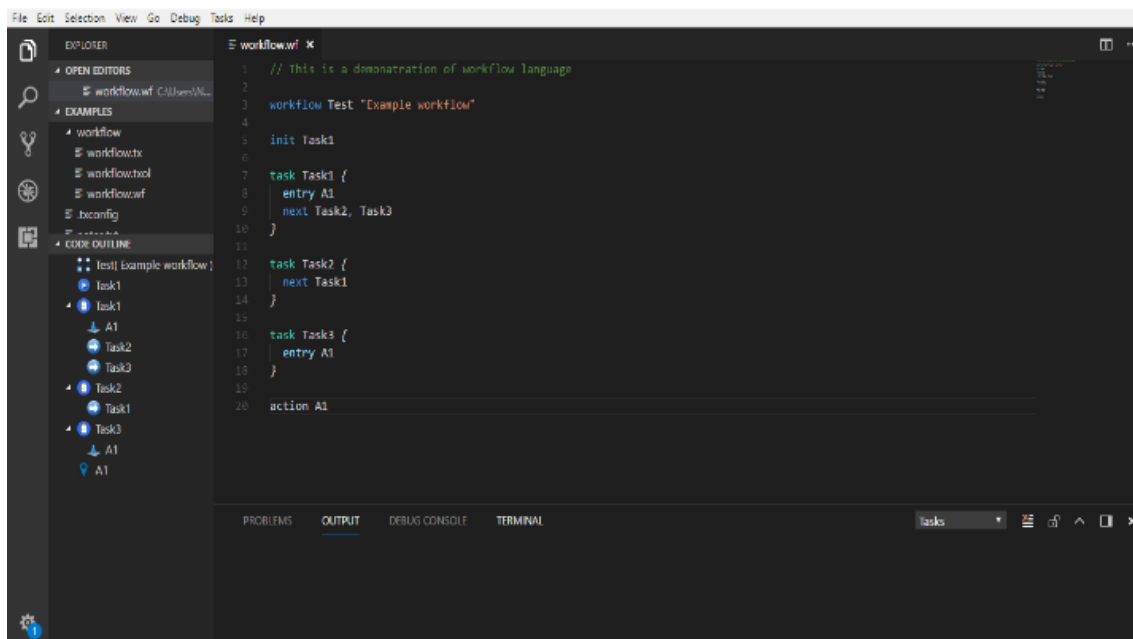
TextX je eksterni tekstualni JSD namenjen za brzu i jednostavnu izgradnju drugih eksternih tekstualnih JSD. Razvijen je pomoću *Arpeggio* parsera na Fakultetu tehničkih nauka u Novom Sadu. Gramatika jezika opisuje se u fajlu sa ekstenzijom *tx*. Kreiranje gramatike u *textX*-u svodi se na kreiranje skupa povezanih pravila. Definisanje pravila započinje zadavanjem njegovog naziva. Nakon naziva pravila navodi se telo. Telo počinje dvotačkom („:“), a završava se tačkom-zarez („;“). Unutar tela navode se konstante, referenciraju se druga pravila i definišu se promenjive. Promenljivoj se dodeljuje tip pomoću operatora dodele. Tip promenjive može biti tipa nekog od osnovnih pravila ili nekog drugog pravila u gramatici. Nakon parsiranja gramatike, kreira se *Python* objekat, a sve promenjive u gramatici su deo tog objekta. Prvo pravilo u gramatici, predstavlja početno pravilo.

4. IMPLEMENTACIJA SISTEMA

U ovom poglavlju ćemo prikazati implementaciju sistema koji omogućava kreiranje podrške u *Visual Studio Code* alatu za jezike napisane pomoću *textX*-a. Jezik za bojenje teksta smeštamo u fajl sa ekstenzijom *.txcl*, dok jezik za prikaz strukture koda smeštamo u fajl sa ekstenzijom *.txol*. Funkcionalnosti za kreiranje navedenih funkcionalnosti se realizuju kreiranjem pravila u navedenim jezicima. Na listingu 1 prikazana je gramatika jezika za *workflow* napisana u *textX*-u.

Jezik za bojenje omogućava korisniku da kreira pravila za bojenje. Gramatikom jezika određujemo šta želimo obojiti, dok *TextMate* specifikacijom navodimo boju kojom treba obojiti pri odgovarajućoj temi. Moguće je bojenje ključnih reči i operatora. Ključne reči su stringovi koji se sastoji od slova, dok su operatori stringovi koji se sastoje samo od specijalnih karaktera.

Definisanje pravila za bojenje pomoću jezika za bojenje odvija se pisanjem programskog koda. Gramatika jezika definiše: konfiguraciju (*configuration*), pravila (*rules*), parove (*matches*) i regularne izraze (*regular expressions*). Unutar konfiguracije navodi se proizvoljna (eng. *default*-na) klasa za ključne reči i operatore, znak koji predstavlja početak i kraj blok komentara i znak za linijski komentar. Bojenje brojeva i stringova (jednostrukih i dvostrukih) je podrazumevano. Unutar odeljka za pravila navode se nazivi pravila iz gramatike i njima odgovarajuće klase. Sve ključne reči i operatori koji se nalaze unutar definisanog pravila pripadaće klasi koja je dodeljena pravilu. Odeljak za parove služi za dodeljivanje klase ključnim rečima i operatorima. Ako je neka ključna reč definisana unutar odeljka za parove, a i nalazi se unutar pravila koje je definisano unutar odeljka za pravila, tada će pripadati onoj klasi iz odeljka za parove (odeljka za parove ima prednost u odnosu na odeljak za pravila). Odeljak za regularne izraze služi za definisanje regularnih izraza i njima odgovarajućih *TextMate* klasa. Na listingu 2 prikazana su pravila za bojenje koda za jezik *workflow*.



Slika 1: Jezik workflow u Visual Studio Code alatu.

```

1  /* Simple workflow language. */
2
3  Program:
4    workflow = Workflow
5    init = Init
6    tasks *= Task
7    actions *= Action
8  ;
9
10 Workflow:
11   'workflow' name=ID desc=STRING?
12 ;
13
14 Init:
15   'init' init=[Task]
16 ;
17
18 Task:
19   'task' name=ID '{'
20   ( entry=Entry | leave=Leave | next=Next ) *
21   '}'
22 ;
23
24 Entry:
25   'entry' entry=[Action]
26 ;
27
28 Leave:
29   'leave' leave=[Action]
30 ;
31
32 Next:
33   'next' next+=[Task][' , ' ]
34 ;
35
36 Action:
37   'action' name=ID
38 ;
39
40 Comment:
41   /\./.*$/
42 ;

```

Listing 1: Gramatika jezika u textX-u (fajl .tx)

```

1  configuration {
2    default:
3      keyword = keyword.control;
4      operation = keyword.operator;
5    comment:
6      line = "//";
7    block:
8      start = "/*";
9      end = "*/";
10 }
11
12 rules {
13   keyword:
14     Workflow, Init = storage.type;
15     Task = entity.name.class;
16     Entry, Leave = support.variable;
17     Next, Action = storage.modifier;
18   operation:
19     Task = emphasis;
20     Next = storage.type;
21 }
22
23 matches {
24   "action" = meta.tag;
25 }

```

Listing 2: Jezik za bojenje teksta (fajl .tcl)

Nakon parsiranja gramatike jezika i jezika za bojenje teksta, moguće je dobiti Python objekte koji reprezentuju model gramatike i model jezika za bojenje. Upoređivanjem ova dva objekta moguće je pronaći sve ključne reči i operatore i odrediti kojim klasama pripadaju. Nakon toga moguće je generisati fajl koji predstavlja uputstvu Visual Studio Code alatu kako da izvrši bojenje. Navedene su default-ne klase za ključne reči i operatore, kao i klase za ključne reči i operatore koje se nalaze unutar odgovarajućeg pravila. Za reč „action“ direktno je navedena klasa.

Cilj jezika za prikaz strukture koda je omogućiti prikaz najvažnijih delova koda u Visual Studio Code alatu za jezike koji su napisani u textX-u. Korisnik navodi nazive pravila čije će vrednosti biti prikazane u prozoru za prikaz

strukture u formi stabla. Jedno pravilo, predstavlja jedan tip čvora. Svaka pojava pravila predstavljena je čvorom u stablu. Za svaki tip pravila potrebno je navesti na koji način će se formirati naziv čvora (labela) i putanju do ikonice koja će ga predstavljati. Svaki čvor ima opseg, tj. liniju i kolonu početka i liniju i kolonu završetka. Klikom na čvor u stablu, biće selektovan deo koda koji odgovara čvora, tj. sve što se nalazi od početka do kraja opsega. Deca (*children*) su čvorovi, čiji opseg je unutar opsega nekog drugog čvora, koji predstavlja njegov roditelj. Na listingu 3 prikazana su pravila za prikaz strukture koda za jezik *workflow*.

```

1 Workflow {
2     label = name + ' ( ' + desc + ' ) '
3     icon = "examples\\workflow\\icons\\workflow.png"
4 }
5
6 Init {
7     label = init
8     icon = "examples\\workflow\\icons\\init.png"
9 }
10
11 Task {
12     label = name
13     icon = "examples\\workflow\\icons\\task.png"
14 }
15
16 Entry {
17     label = entry
18     icon = "examples\\workflow\\icons\\entry.png"
19 }
20
21 Leave {
22     label = leave
23     icon = "examples\\workflow\\icons\\leave.png"
24 }
25
26 Next {
27     label = next
28     icon = "examples\\workflow\\icons\\next.png"
29 }
30
31 Action {
32     label = name
33     icon = "examples\\workflow\\icons\\action.png"
34 }

```

Listing 3: Jezik za prikaz strukture koda (fajl .txol)

Prikaz strukture koda započinje slanjem komande *Language Serveru*. Unutar komande nalazi se gramatika jezika, model jezika za prikaz strukture koda i programski kod. Parsiranjem modela jezika za prikaz strukture koda i programskog koda na osnovu gramatike jezika moguće je dobiti objekte koji reprezentuju model jezika za prikaz strukture koda i meta-model programskog koda. Upoređujući ova dva objekta možemo zaključiti kako izgleda stablo. Ovo stablo se vraća *Visual Studio Code* editoru u obliku *JSON* objekta. Prolazeći kroz ovaj objekat *Visual Studio Code* uz pomoć *Visual Studio Code namespace API* biblioteke prikazuje strukturu koda u zasebnom prozoru. Svako pravilo reprezentuje jedan tip čvora i preuzima ikonicu sa odgovarajuće putanje.

Labelu predstavlja odgovarajuća vrednost koja se dobija iz objekta koji je rezultat parsiranja modela. Na slici 1 se može videti da je programski kod za jezik *workflow* obojen odgovarajućim bojama. Sa leve strane se nalazi struktura koda u formi stabla. Klikom na odgovarajući čvor u stablu biće selektovan odgovarajući deo koda.

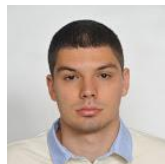
5. ZAKLJUČAK

U ovom radu bavili smo se podrškom za jezike napisane u *textX*-u. U prvom delu rada objasnili smo šta su JSD, zbog čeka ih koristimo, šta su im prednosti i nedostaci. Predstavili smo *textX* kao JSD koji nam pomaže da konstruišemo sopstvene JSD. Međutim, nakon što kreiramo JSD pomoću *textX*-a, za njega ne postoji adekvatno programsko okruženje. Kreiranje okruženja za konstruisani JSD je teško i mukotrpno. Ponuđena su dva nova JSD koja omogućavaju kreiranje podrške u *Visual Studio Code* alatu. Danas postoji pregršt JSD koji su razvijeni uz pomoć *textX*-a. Mnogi od njih nemaju podršku ni u jednom programskom okruženju, pa je korisnicima tih JSD konstrukcija programa otežana. Nadamo se da će ponuđena rešenja pomoći tvorcima JSD da za njih izgrade i odgovarajuće programsko okruženje, i tako olakšaju korišćenje njihovih JSD krajnjim korisnicima.

6. LITERATURA

- [1] Leonard Elezi "Spoofax vs Xtext", http://msdl.cs.mcgill.ca/people/hv/teaching/MSBDesi gn/201314/projects/Leonard.Elezi/report/reading_repo rt.pdf, poslednji pristup: 27. Maj 2018.
- [2] Stackoverflow: "Most popular development enviroments", <https://insights.stackoverflow.com/survey/2018/#devel opment-environments-and-tools>, poslednji pristup: 27. Maj 2018.
- [3] Miroslav Hajduković: „Arhitektura računara“, Novi Sad, 2013
- [4] Markus Voelter: "DSL Engineering", 2010-2013, dslbook.org
- [5] Igor Dejanović: "Uvod u jezike specifične za domen", <http://www.igordejanovic.net/courses/jsd/uvod.html>, poslednji pristup: 27. Maj 2018.
- [6] Igor Dejanović: "Tekstualne sintakse", <http://www.igordejanovic.net/courses/jsd/tekstualne-sintakse.html>, poslednji pristup: 27. Maj 2018.

Kratka biografija:



Nemanja Starčev rođen je u Novom Sadu 1993. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva - softversko inženjstvo odbranio je 2018. godine.

**ELEKTRONSKI SISTEM MERENJA RAZDALJINE I GENERISANJA UPOZORENJA
NA PREPREKE KAO POMOĆ SLEPIM I SLABOVIDIM OSOBAMA****ELECTRONIC SYSTEM FOR DISTANCE MEASUREMENT WITH OBSTACLE ALERT
GENERATION AS ASSISTANCE TO THE VISUALLY IMPAIRED**

Dejan Krstić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je razmatran problem detekcije prepreke na putanji kretanja slepih i slabovidnih osoba. Akcenat je stavljen na prepreke koje se nalaze na određenoj visini iznad tla i koje su samim time teže za detekciju upotrebom belog štapa. Predstavljeno je rešenje za proračunavanje razdaljine prepreke, principom generisanja ultrazvučnih talasa. Kritičan parametar za precizno proračunavanje razdaljine predstavljaju temperatura i vlažnost vazduha. Ova dva parametra unose se u proračun brzine zvuka kroz vazduh, obzirom da je elektronski sistem namenjen za upotrebu na otvorenom prostoru. Prikazani su i upoređeni rezultati merenja sa dva različita senzora.

Abstract – The paper reviews the problem of an obstacle detection on the path when moving the visually impaired. The accent is placed on obstacles that are at a certain height above the ground level, which are therefore more difficult to detect using a white cane. A solution for calculating distance of an obstacle was presented based on the principle of generating ultrasonic waves. A critical parameter for precise calculation of the distance is an air temperature and relative humidity. These two parameters are entered into the ultrasonic air velocity calculation, since the electronic system is intended for outdoor use. Temperature and relative humidity measurement results by two of the sensors are presented.

Cljučne reči: primenjena elektronika, medicinska elektronika, ultrazvuk, temperatura vazduha, relativna vlažnost vazduha, mikrokontroler, senzor, Arduino

1. UVOD

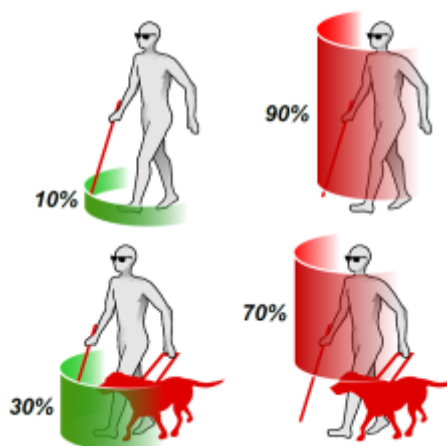
Slepe i slabovide osobe nailaze na brojne probleme prilikom kretanja u otvorenom prostoru (ulica, park, trg, plato, pothodnik, sajamski paviljon) na kojem se može nalaziti veliki broj predmeta i potencijalnih prepreka.

Najviše problema odnosi se na sam pristup takvom prostoru (uspon, spust, stepenice, rampe), ali i na prepreke koje tom prilikom treba zaobići (stubovi, klupe, ulični mobilijar, reklamni panoi, ograde). Prepreke na tlu do visine cca 10-20 cm poput ivičnjaka i stepenika mogu biti uspešno detektovani upotrebom belog štapa, pa zato slepim i slabovidnim osobama ne predstavljaju veći problem.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Goran Stojanović, red. prof.

U najvećem broju slučajeva oni savladavaju takve prepreke bez većih poteškoća, što je rezultat navike, koja se vremenom stiče i razvija kod ovih osoba. Međutim veći procenat gornje površine tela (do 90%) ostaje nedovoljno zaštićen (Slika 1).



Slika 1. Površina tela izložena opasnosti od povrede, beli štap (gore), beli štap i pas vodič (dole)

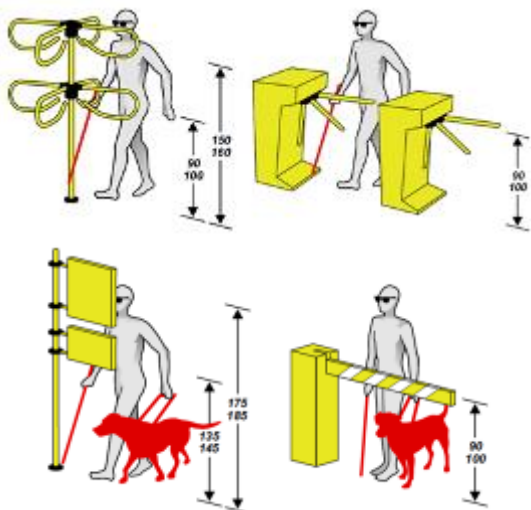
Upotreba specijalno dresiranih pasa vodiča, kao pomoć slepim i slabovidnim osobama, može da poveća sigurnost u kretanju do određenog nivoa (do 30-50 cm od nivoa tla) i zaštititi veću površinu tela (do 30%). Međutim, ne mogu se sve osobe osloniti na takvu pomoć, obzirom na mogućnost postojanja urođenog straha od životinje ili alergijskih reakcija na životinjsku dlaku.

**2. PROBLEMI PRI KRETANJU SLEPIH I
SLABOVIDIH OSOBA NA OTVORENOM**

Prepreke na visinama iznad 50 cm od nivoa tla, predstavljaju daleko veći problem za pravovremenu detekciju. Nažalost, u takvim slučajevima beli štap ne predstavlja dovoljno efikasno i fleksibilno sredstvo za detekciju prepreka. Isto se može reći i za pse vodiče.

Najkritičnije prepreke, koje se mogu naći na putanji kretanja, a da ne budu pravovremeno detektovane, jesu one u nivou kolena, natkolenice, karličnog pojasa, abdomena, grudnog koša, vrata i glave (Slika 2).

Povrede uzrokovane nailaskom na takve prepreke mogu izazvati, kako lakše telesne povrede, tako i povrede sa daleko težim posledicama po zdravlje slepe i slabovide osobe. Povrede sa težim posledicama mogu kod slepih i slabovidnih osoba doprineti dodatnom strahu i nelagodi od svakog daljeg kretanja u otvorenom prostoru, što je uzrokovano upravo lošim prethodnim iskustvom.

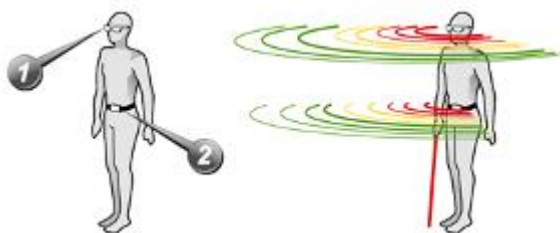


Slika 2. Potencijalne prepreke i problemi pri kretanju slepih i slabovidih osoba na otvorenom

Da bi se povećala bezbednost slepih i slabovidih osoba i zaštitila veća površina tela od povreda, uzrokovanih preprekama, predloženo je rešenje u vidu elektronskog sistema, koji bi ove osobe koristile tokom svog kretanja na otvorenom prostoru.

3. ELEKTRONSKI SISTEM MERENJA RAZDALJINE

Elektronski sistem zasnovan je na konstantnom merenju razdaljine od osobe do prepreke, u realnom vremenu i na generisanju audio upozorenja o neposrednom prisustvu prepreke na određenoj kritičnoj razdaljini. Sistem bi shodno potrebi za mobilnošću, bio integrisan u naočare odnosno u pojas na struku (Slika 3), a mogao bi biti postavljen i na svako drugo željeno mesto na telu, kao što su, na primer, abdomen ili grudni koš.



Slika 3. Pozicioniranje uređaja na telu, (1) naočare, (2) pojas na struku

Prevashodni cilj jeste zaštita gornjih delova tela, a zavisno od načina upotrebe belog štapa (širine zamaha, pozicije ruke), po potrebi i želji korisnika, pojas sa uređajem može se izmestiti i postaviti u visini grudnog koša. Uređaj je predviđen za upotrebu u kombinaciji sa belim štapom i/ili psom vodičem. U svakoj od ovih kombinacija, rezultat je povećanje zaštićene površine tela (od 60-80%), što doprinosi povećanju ukupne bezbednosti pri kretanju na otvorenom.

3.1. Arduino® platforma elektronskog sistema

Elektronski sistem baziran je na Arduino® platformi i razvijen korišćenjem razvojnog sistema Arduino® UNO Rev3, koji dolazi sa mikrokontrolerom ATMEL® ATmega328P, sa radnim taktom na 16 MHz [1]. Pored toga korišćen je

senzor ultrazvuka HC-SR04, kao i senzor temperature i vlažnosti vazduha DHT11C i pasivna piezo zujalica (Sl. 4).



Slika 4. Komponente elektronskog sistema

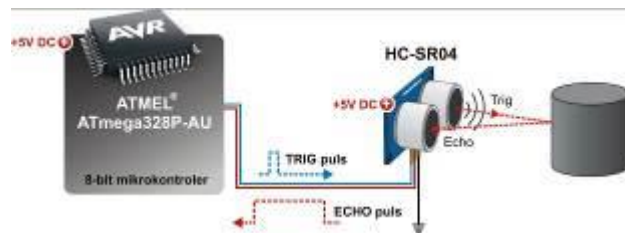
Sistem predviđa audio indikaciju radi pravovremenog upozorenja na prepreku, što se ostvaruje pomoću pasivne piezo zujalice (ili preko slušalica), sa tonovima na različitim frekvencijama u zavisnosti od blizine prepreke (900 Hz, 1200 Hz, 1500 Hz). Napajanje se ostvaruje upotrebom punjive litijum polimer baterije (Li-Po 3.7 V, 1000-2000 mAh). Kolo za napajanje sastoji se od dela za dopunjavanje i zaštitu baterije od prekomernog praznjenja (OD, overdischarge) i prepunjavanja (OC, overcharge), kao i dela za podizanje napona (step-up) na nivo od 5 V 100 mA, koji je potreban za funkcionisanje ostatka kola. Baterija se dopunjuje putem USB konekcije, a uključivanje i isključivanje uređaja vrši se pomoću kliznog prekidača. Za potrebe programiranja sistema, u isti je integrisan 6-pinski ICSP konektor (In Circuit Serial Programming). Blok dijagram elektronskog sistema uključuje sve navedene komponente (Slika 5).



Slika 5. Blok dijagram projektovanog elektronskog sistema

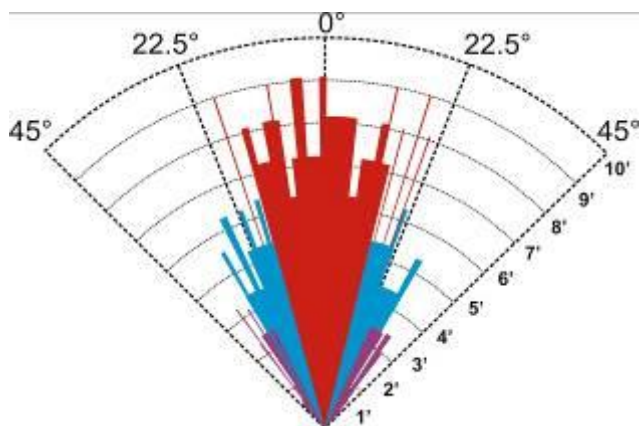
3.2. Generisanje ultrazvučnih talasa

Merenje razdaljine vrši se principom konstantnog emitovanja ultrazvučnih talasa sa predajnika senzora HC-SR04, njihovim prostiranjem u slobodan prostor kroz vazduh, odbijanjem od površine prepreke i vraćanjem nazad ka prijemu, koji se nalazi na istom senzoru (Slika 6).



Slika 6. Senzor HC-SR04, generisanje ultrazvučnih talasa

Senzor HC-SR04 ima rezoluciju od 0,3 cm i može da meri rastojanja od 2-450 cm, sa najvećom efikasnošću pri uglu od 15° u odnosu na svoju osu [2], odnosno ukupno do 30° (Slika 7).



Slika 7. Ultrazvučni senzor HC-SR04, intenzitet generisanog talasa u odnosu na ugao [2]

Ultrazvučni senzor HC-SR04, koji je korišćen u svrhu generisanja ultrazvučnih talasa, pobuđuje se dovođenjem okidačkog impulsa sa ATmega328P na pin *Trig* na senzoru, u trajanju od 10 μs. Senzor preko predajnika emituje povorku od osam impulsa ultrazvučnog talasa, frekvencije $f=40000$ Hz (*Ping*). Emitovani talas odbija se od prepreke i vraća ka prijemniku, čime se pobuđuje pin *Echo* i signal prosleđuje na ulaz mikrokontrolera. Dužina trajanja tog signala (*Ping Time*) proporcionalna je dvostrukoj pređenoj razdaljini emitovanog i odbijenog talasa.

3.3. Izračunavanje razdaljine prepreke

Brzina zvuka kroz vazduh v , direktno je srazmerna pređenoj razdaljini d (dato u metrima), a obrnuto srazmerna proteklom vremenu t (dato u sekundama) za koje zvuk pređe tu razdaljinu. Imajući u vidu da je u pitanju dvostruka razdaljina $2d$, jer talas istu razdaljinu pređe i u povratku nakon odbijanja od prepreke, izračunavamo je prema formuli (1).

$$d = \frac{v \cdot t}{2} \quad (1)$$

Poznavanjem obe promenljive (v , t) jednostavno je odrediti rastojanje d do prepreke na putanji kretanja. Za brzinu zvuka kroz vazduh, uopšteno se uzima vrednost $v=331,4$ m/s, dok proteklo vreme t (*Ping Time*) dobijamo na ulazu mikrokontrolera merenjem trajanja promene logičkog nivoa signala (*Echo*).

Imajući u vidu da su date promenljive u metričkim jedinicama većeg odnosno manjeg reda veličine (m/s, μs), neophodno je uraditi dimenzionu analitiku i date jedinice pretvoriti u jedinice cm/s odnosno s, a krajnju vrednost razdaljine prepreke, umesto u m, izraziti u cm, prema formuli (2).

$$d = \frac{v \cdot t}{2} \cdot \frac{10^2}{10^6} \quad (2)$$

Kompletni navedeni postupak vrši se direktno u programskom kodu u Arduino® IDE radnom okruženju.

4. UTICAJ TEMPERATURE I VLAŽNOSTI VAZDUHA NA PRORAČUN RASTOJANJA

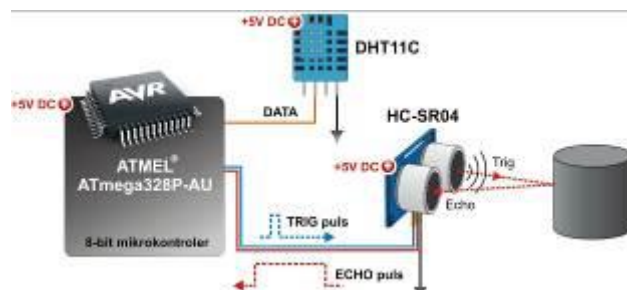
Brzina zvuka kroz vazduh menja se sa promenom temperature T (u °C) i relativne vlažnosti vazduha H (%). Ova dva parametra postaju kritična imajući u vidu da je projektovani sistem namenjen upotrebi na otvorenom prostoru u svim vremenskim uslovima. Zbog toga se koristi proširena formula (3) za izračunavanje brzine zvuka kroz vazduh v , koja u sebi sadrži i ova dva nova parametra.

$$v = 331,4 + (0,606 \cdot T) + (0,0124 \cdot H) \quad (3)$$

Da bi se postigla veća preciznost merenja, potrebno je sistem nadograditi senzorom temperature i relativne vlažnosti vazduha. Upotreba termistora delimično rešava problem merenja temperature i konverzije njegove otpornosti u temperaturnu vrednost upotrebom Steinhart-Hart jednačine (4), uz neophodne metričke konverzije.

$$\frac{1}{T} = A + B \cdot \ln(R) + C \cdot [\ln(R)]^3 \quad (4)$$

Međutim, relativna vlažnost vazduha kao drugi kritični parametar, bez adekvatnog senzora, ostaje na ovaj način izuzeta iz proračuna. U svrhu prevazilaženja ovog problema, umesto termistora, upotrebljen je senzor DHT11C, koji u sebi objedinjuje oba parametra i koji preko jednog izlaznog pina (*Data*) šalje digitalni signal na ulaz ATmega328P (Slika 8). Time je u programiranju izbegnuta upotreba Steinhart-Hart formule za konverziju otpornosti u temperaturu i dodatno je pojednostavljen celokupan sistem, manjim brojem komponenti u njemu.



Slika 8. Senzor DHT11C, merenje temperature i relativne vlažnosti vazduha

Za potrebe rada sa senzorom DHT11C u radnom okruženju Arduino® IDE, neophodno je prethodno učitavanje i aktiviranje posebne biblioteke DHTLib sa svim potrebnim funkcijama za rad (promenljive *DHT.temperature* i *DHT.humidity*), koja pojednostavljuje celokupan postupak programiranja ovog sistema u svega nekoliko linija programskog koda, o čemu se više reči može pronaći u [3].

```
vreme = pulseIn(echoPin,HIGH);
brzina = 331.4 + (0.606 * DHT.temperature) +
(0.0124 * DHT.humidity);
rastojanje = (vreme/2) * (brzina/10000);
```

Umesto upotrebljenog senzora DHT11C, moguća je upotreba senzora DHT22 sa boljim karakteristikama, koji takođe zadovoljava potrebe ovakvog sistema. Senzor DHT22 ima širi opseg merene temperature i relativne vlažnosti vazduha, uz bolju rezoluciju, ali i duži period

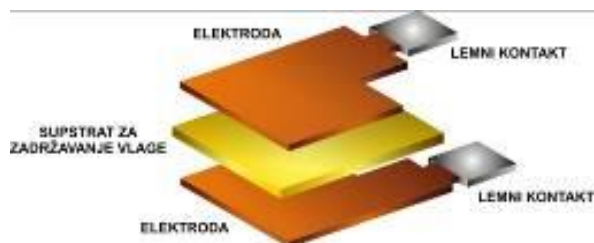
semplovanja, odnosno očitavanja atmosferskih parametara. Uporedni prikaz i karakteristike oba senzora DHT11C i DHT22 predstavljene su u Tabeli 1 [4].

Tabela 1. Karakteristike senzora DHT11C i DHT22 [4]

Senzor	DHT11C	DHT22
Temperatura	0 - 50 °C / ± 2 °C	-40 - 125 °C / $\pm 0,5$ °C
Relativna vlažnost	20 - 80 % / ± 5 %	0 - 100 % / $\pm 2-5$ %
Radni takt	1 Hz (1 očitavanje / 1 sek)	0,5 Hz (1 očitavanje / 2 sek)
Radni napon	3 - 5 V DC	3 - 5 V DC
Maksimalna struja merenja	2,5 mA	2,5 mA
Fizičke dimenzije	15,5 mm x 12 mm x 5,5 mm	15,1 mm x 25 mm x 7,7 mm

4.1. Struktura senzora DHT11C/DHT22

Oba senzora sačinjena su od NTC termistora (negativni temperaturni koeficijent), senzora relativne vlažnosti i IC za procesiranje analognih vrednosti. Senzor relativne vlažnosti vazduha sastoji se od dve elektrode sa pristupnim (lemlnim) kontaktima. Između dve elektrode nalazi se supstrat za zadržavanje vlage (Slika 9).

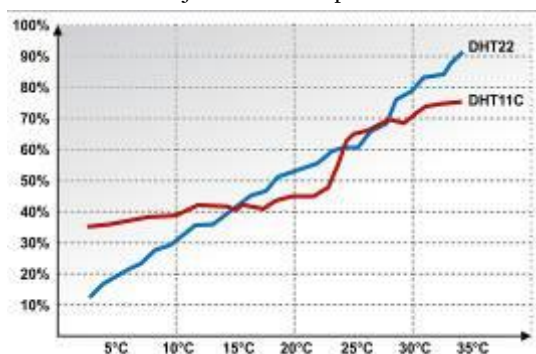


Slika 9. Struktura senzora relativne vlažnosti vazduha na DHT11C i DHT22 [4]

Povećanjem relativne vlažnosti vazduha, supstrat apsorbira srazmerno veću količinu vlage i samim time proporcionalno raste i elektroprovodljivost između elektroda direktno kroz supstrat, odnosno smanjuje se otpornost među elektrodama. Vrednost otpornosti među elektrodama konvertuje se manjim IC na senzoru i kao digitalan signal šalje ka ATmega328P, u realnom vremenu svakih 1 ili 2 sekunde, zavisno od tipa senzora, DHT11C ili DHT22, respektivno.

4.2. Performanse senzora DHT11C/DHT22

Oba senzora, zadovoljavaju potrebe projektovanog sistema, međutim u realnim uslovima korišćenja uređaja na otvorenom, bolje performanse ostvaruje DHT22, preciznijim očitavanjem vrednosti, što je ponajviše zasluga bolje rezolucije (Slika 10). Takođe ovaj senzor ima veći temperaturni opseg, koji se kreće i ispod 0 °C, što ga čini idealnim za korišćenje u zimskom periodu.



Slika 10. Očitavanje temperature i relativne vlažnosti vazduha sa DHT11C i DHT22

Međutim, u opsegu spoljašnjih dnevnih temperatura vrednosti 10-30 °C (od proleća do jeseni), izmerene vrednosti oba senzora mogu se smatrati prihvatljivim i dovoljno pouzdanim za upotrebu u ovakvom sistemu. Višestruko niža cena senzora DHT11C može dodatno umanjiti cenu samog uređaja, prućenu kompaktnijim fizičkim dimenzijama, što je svakako poželjno pri dizajnu uređaja, koji bi se koristio integrisan u naočare slepe i slabovide osobe.

5. ZAKLJUČAK

Projektovani elektronski sistem, prevashodno je osmišljen kao dodatak i pomoć slepim i slabovidim osobama u njihovom svakodnevnom kretanju na otvorenom. Uređaj ni u kojem slučaju ne predstavlja zamenu belom štapu, već dodatak za savladavanje problema sa preprekama koje se ne mogu na vreme detektovati štapom.

Uređaj je prilagodljiv svakoj slepoj i slabovidoj osobi pojedinačno, u pogledu potreba ili želja za personalizovanjem istog. Tako se, na primer, unapred može izabrati vrsta napajanja, broj kritičnih razdaljina za aktiviranje upozorenja kao i sami tonovi upozorenja, intenzitet i frekvencija, izlaz preko piezo zujalice ili slušalice i dr.

6. LITERATURA

- [1] B.van Dam, "Arduino UNO – 45 Projects For Beginners and Experts", 2016.
- [2] HC-SR04 Ultrasonic Ranging Module Datasheet
- [3] Circuit Basics, "How To Set Up DHT11 Humidity Sensor on an Arduino", 2015.
- [4] DHT11 Temperature and Humidity Module Datasheet

Kratka biografija:



Dejan Krstić, rođen je u Subotici 1985. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Mikrotalasna elektronika, odbranio je 2017. god. na temu "Realizacija antena različitog dizajna na papiru i fleksibilnim materijalima"

PROJEKTOVANJE INTEGRISANOG IZVORA REFERENTNOG NAPONA U 0,35 μm AMS CMOS TEHNOLOGIJI

DESIGN OF BANDGAP VOLTAGE REFERENCE IN 0.35 μm AMS CMOS TECHNOLOGY

Dragan Žilić, Fakultet tehničkih nauka, Novi Sad

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je prikazano projektovanje izvora referentnog napona vrednosti od 1,38 V. Kolo se sastoji iz dvostepenog operacionog pojačavača sa polarizacionim kolom (beta množačem) i potrebnih delova za generisanje napona sa pozitivnim i negativnim temperaturnim koeficijentom (eng. temperature coefficient - TC) [ppm/°C]. Izvor je projektovan u softverskom paketu Cadence korišćenjem 0,35 μm CMOS tehnologije. Napon napajanja je 3,3 V.

Abstract – This paper presents the design of bandgap voltage reference value of 1.38 V. The circuit is consisted of two stage operational amplifier with bias circuit (beta multiplier) and required blocks for generating voltage with positive and negative temperature coefficient (TC) [ppm/°C]. The circuit is designed in Cadence software using 0.35 μm CMOS technology. The power supply is 3.3 V.

Ključne reči: Projektovanje analognih integrisanih kola, bandgap naponska referenca, operacioni pojačavač, CMOS tehnologija.

1. UVOD

Izvori referentnog napona ili naponske reference imaju široku primenu u integrisanim kolima jer pružaju stabilan napon napajanja pri temperaturnim varijacijama. Kola naponske reference najveću primenu imaju u A/D (eng. analog to digital) i D/A (eng. digital to analog) konvertorima, naponskim regulatorima, i tako dalje. Na slici 1 prikazan je primer primene naponske reference u A/D konvertorima. Napon bita najmanje težine V_{LSB} u slučaju 3-bitnog A/D konvertora koji ima referentni napon (napon napajanja) V_{REF} od 3,3 V, se može definisati kao:

$$V_{LSB} = \frac{V_{REF}}{2^N} = \frac{3.3}{2^3} = 0,4125 \text{ V}, \quad (1)$$

gde je N broj bita.

Da bi kolo radilo ispravno maksimalna promena referentnog napona u zavisnosti od temperature mora da bude manja od 7,638 mV/°C [1]:

$$\frac{\Delta V_{REF}}{\Delta T} = \frac{V_{LSB}}{27^\circ \text{C}} = 7,638 \text{ mV/}^\circ\text{C}. \quad (2)$$

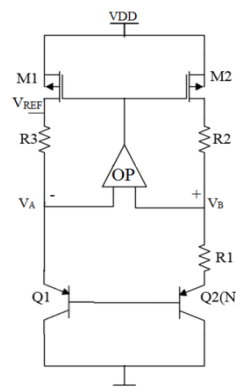
NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila dr Jelena Radić, docent

2. TEORIJSKA ANALIZA RADA IZVORA REFERENTNOG NAPONA

Na slici 2 prikazana je šema projektovane naponske reference. Kolo funkcioniše na principu strujnog ogledala. Mosfetovi M_1 i M_2 imaju ulogu izvora konstantne struje. Oni su kontrolisani izlaznim naponom operacionog pojačavača (OP).

Operacioni pojačavač se koristi kao pojačavač greške. Treba da obezbedi jednakost napona na ulazu. Time se postiže naponska ravnoteža u granama kola naponske reference.



Slika 2. Šema projektovane naponske reference [2]

Temperaturno nezavisni napon se generiše na izlazu V_{REF} . Otpornosti R_2 i R_3 moraju biti jednake vrednosti kako bi bio zadovoljen uslov $V_{SDM1} = V_{SDM2}$. Ako se pretpostavi da je napon ofseta OP jednak nuli u idealnom slučaju, referentni izlazni napon može da se definiše kao, [1], [2] i [3]:

$$V_{REF} = V_{EB1} + \frac{R_3}{R_1} \cdot \frac{kT}{q} \ln(N), \quad (3)$$

gde je k Boltzmanova konstanta, T apsolutna temperatura izražena u Kelvinima, q naelektrisanje elektrona i N odnos površina spoja baza-emiter tranzistora Q_2 i Q_1 .

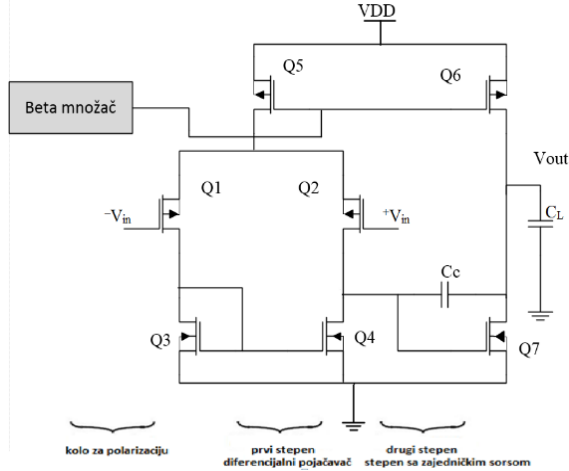
Napon V_{EB1} opada sa temperaturom (napon diodno vezanog PNP tranzistora), dok drugi deo jednačine (3) predstavlja pad napona na otporniku R_3 koji raste sa temperaturom. Temperaturna osetljivost referentnog napona može da se posmatra kao:

$$\frac{dV_{REF}}{dT} = \frac{dV_{EB1}}{dT} + \frac{R_3}{R_1} \cdot \frac{k}{q} \ln(N). \quad (4)$$

Iz (4) sledi da je referentni napon nezavisan od promene temperature ako naponi sa pozitivnim (V_{EB1}) i negativnim (napon na otporniku R_3) temperaturnim koeficijentom imaju iste temperaturne koeficijente suprotnih predznaka. Ova topologija ograničava CMOS dizajn ukoliko je potreban napon na izlazu manji od 1,2 V, jer se na izlazu dobija napon približno dva puta veći od napona praga provođenja (bipolarnog) tranzistora.

2.1. Dvostepeni operacioni pojačavač

Za postizanje dobrih performansi i ispravan rad *bandgap* naponske reference izabran je dvostepeni operacioni pojačavač (slika 3). Prednosti topologije su širok opseg napona na izlazu i veliko pojačanje čime se obezbeđuje veća mogućnost za izjednačavanje napona u tačkama A i B (slika 2). Takođe dvostepeni OP ima mali šum koji manje utiče na izlazni napon. U obzir može da se uzme i snaga disipacije koja nije velika.



Slika 3. Šematski prikaz dvostepenog OP [3]

Ukupno pojačanje OP iznosi [3]:

$$A_v = g_{m1}g_{m7}(r_{ds6}||r_{ds7})(r_{ds2}||r_{ds4}). \quad (5)$$

Jednačine za dominantni i nedominantni pol su [3]:

$$\omega_{p1} \cong \frac{1}{g_{m7}(r_{ds4}||r_{ds2})(r_{ds6}||r_{ds7})C_c}, \quad (6)$$

$$\omega_{p2} \cong \frac{g_{m7}}{C_{db2}+C_{db4}+C_{gs7}+C_{db7}+C_{db6}+C_L}. \quad (7)$$

Pored ovih polova dobija se i jedna nula na frekvenciji:

$$\omega_z = \frac{-g_{m7}}{C_c}. \quad (8)$$

Nula unosi fazni pomak i utiče na stabilnost kola, a predstavlja problem jer se nalazi u desnoj poluravni. Povećavanje kapacitivnosti C_c smanjuje frekvenciju prvog, dominantnog pola bez promene položaja drugog pola i time povećava stabilnost kola. Međutim, pošto se ovim načinom smanjuje i položaj nule ukupan rezultat ne doprinosi poboljšanju stabilnosti kola. Zbog toga se uvodi kompenzacioni otpornik R_c (na red sa Milerovim kondenzatorom C_c) koji obezbeđuje dodatni stepen slobode u podešavanju položaja nule.

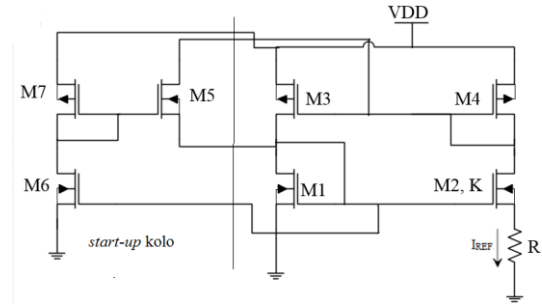
Frekvencija nule u slučaju OP koji ima kompenzacioni otpornik R_c na red sa kondenzatorom C_c se dobija kao [3]:

$$\omega_z = \frac{1}{C_c \left(\frac{1}{g_{m7}} - R_c \right)}. \quad (9)$$

Odabirom odgovarajuće vrednosti otpornika nula se može pomeriti u levu poluravan na vrednost nešto veću od frekvencije gde strujno pojačanje pojačavača ima vrednost jedan (što je približno jednako proizvodu pojačanja i propusnog opsega), eliminisati ili poklopiti sa nedominantnim polom, čime se poništava njegovo dejstvo.

2.2. Beta množač sa start-up kolom

Beta množač predstavlja kolo za polarizaciju dvostepenog OP (slika 4). Da bi se dobila željena vrednost referentne struje I_{REF} , potrebno je odrediti vrednost otpornosti R iz formule (10):



Slika 4. Šema beta množača sa start-up kolom [4]

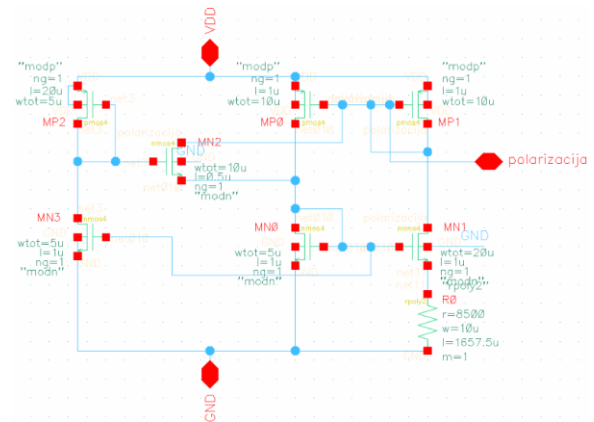
$$I_{REF} = \frac{2}{R^2 K P_n \frac{W_1}{L_1}} \left(1 - \frac{1}{\sqrt{K}} \right)^2, \quad (10)$$

gde K predstavlja odnos širina kanala tranzistora M_2 i M_1 . Obično se uzima da je $K=4$, jer je u tom slučaju vrednost otpornika $R=1/g_m$ (recipročna vrednost transkonduktanse mosfeta M_1) [4]. Start-up kolo se dodaje da bi se sprečilo stanje kada struja u beta množaču ima vrednost nula.

3. PROJEKTOVANJE NAPONSKE REFERENCE I REZULTATI SIMULACIJE

3.1. Projektovanje beta množača sa start-up kolom

Projektovano kolo prikazano je na slici 5. Na osnovu jednačine (10) i podešavanjem vrednosti vrednosti otpornika tokom simulacije, dobijene su se sledeće vrednosti parametara: $K=4$, $R=8,5 \text{ k}\Omega$, $I_{REF}=10,2 \mu\text{A}$ i $I_{MN0}=10,8 \mu\text{A}$. U tabeli 1 date su projektovane (ili optimalne) dimenzije mosfetova.



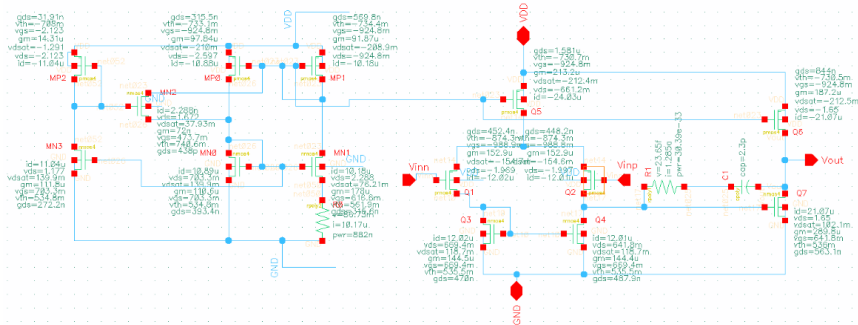
Slika 5. Šema projektovanog beta množača

Tabela 1. Optimalne dimenzije mosfetova beta množača

Mosfet	W [μm]	L [μm]
MN ₀	5	1
MN ₁	20	1
MP ₀ , MP ₁	10	1
MN ₂	5	0,5
MN ₃	5	1
MP ₂	5	20

3.2. Projektovanje dvostepenog OP

Šema projektovanog OP prikazna je na slici 6. Proračun vrednosti parametara komponenti dvostepenog pojačavača izvršen je na način objašnjen u [1] i [3], a zahtevane vrednosti karakteristika kola su date u tabeli 2. Dužina kanala od $L=1 \mu\text{m}$ je korišćena za sve mosfetove u kolu operacionog pojačavača.



Slika 6. Šema projektovanog dvostepenog operacionog pojačavača

Tabela 2. Željene vrednosti parametara dvostepenog pojačavača

Parametar	Vrednost
Naponsko pojačanje [dB]	>70
GBW* [MHz]	>5
Potrošnja [mW]	<1
Opseg ulaznog napona [V]	0,2 – 2,0
Opseg izlaznog napona [V]	0,2 – 3,0
Maksimalna brzina promene napona na izlazu [V/μs]	≥10
C_L [pF]	10
Fazna margina (PM) [°]	≥60
V_{OSET} [V]	≈ 0

*GBW - (eng. gain bandwidth) proizvod pojačanja i propusnog opsega

Za kompenzaciju operacionog pojačavača korišćena je tehnika ubacivanja Milerove kapacitivnosti i otpornika gde je vrednost otpornika podešena tako da se položaji nule i nedominantnog pola poklapaju. Jednačina kojom se određuje vrednost otpornika u primenjenoj tehnici ima oblik [3]:

$$R_c = \frac{1}{g_{m7}} \left(1 + \frac{C_L}{C_c} \right) \approx 18,4 \text{ k}\Omega. \quad (11)$$

Rezultati simulacija su pokazali da se povećanjem širine $W_{1,2}$ povećava pojačanje i smanjuje PM operacionog pojačavača. Smanjenjem vrednosti kondenzatora C_c povećava se parametar SR i smanjuje PM . Povećanje širine W_5 uzrokuje povećanje polarizacione struja, što za rezultat ima povećanje parametara SR i PM , ali i smanjenje pojačanje usled smanjenja izlaznih otpornosti mosfet tranzistora.

Tabela 3. Optimalne vrednosti parametara OP

Parametar	Vrednost
$W_{1,2}$	25,0 μm
$W_{3,4}$	12,0 μm
W_5	23,0 μm
$W_{6,7}$	19,3 μm
L	1,0 μm
C_C	2,3 pF
R_C	18,4 kΩ

Povećanjem odnosa dimenzija tranzistora Q_6 i Q_7 povećava se pojačanje i smanjuje PM . Optimalne vrednosti parametara komponenti kola su date u tabeli 3.

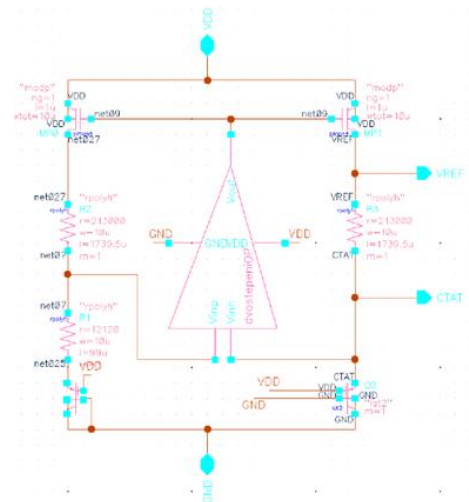
3.3. Projektovanje šematika i lejauta naponske reference

Optimalne vrednosti parametara u kolu naponske reference, prikazane na slici 7, su: $W_{P0}=W_{P1}=10 \mu\text{m}$, $R_4=12,12 \text{ k}\Omega$, $R_0=R_5=213 \text{ k}\Omega$. Broj lateralnih PNP tranzistora je $N=3$. Pokazalo se da ovaj broj daje najbolje performanse za postizanje nezavisnosti referentnog napona od promene temperature.

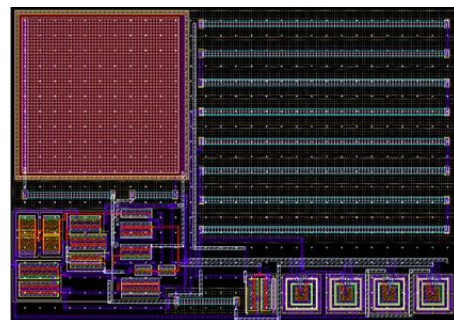
Na slici 8 prikazan je konačan izgled nacrtu fizičke realizacije (eng. layout) bandgap naponske reference. Prilikom povezivanja komponenti kola korišćena su samo dva metalna sloja. Površina koju lejaut zauzima iznosi $133 \mu\text{m} \times 92 \mu\text{m} = 12236 \mu\text{m}^2$.

Rezultati simulacije kola sa uključenim parazitnim kapacitivnostima i otpornostima prikazani su na slici 9, a dobijene vrednosti parametara kola, uključujući i podatke za operacioni pojačavač, su date u tabeli 4.

Za faznu marginu i pojačanje operacionog pojačavača dobijene su veoma dobre vrednosti, a prilikom projektovanja naponske reference parametri kola su podešeni tako da se karakteristike napona sa pozitivnim i negativnim temperaturnim koeficijentom poklope na temperaturu oko 27.5°C (27°C za rezultat simulacija kola na novu šematika).

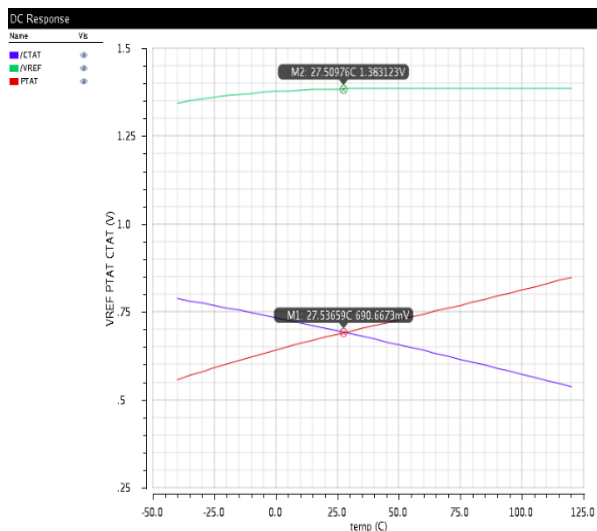


Slika 7. Kolo projektovane naponske reference



Slika 8. Lejaut naponske reference

U tabeli 5 upoređene su vrednosti parametara dobijenih u ovom radu nakon post-lejaut simulacije sa vrednostima parametara radova iz pojedinih naučnih časopisa prethodnih godina, [5]-[9].



Slika 9. Zavisnost napona na izlazu naponske reference od promene temperature

Tabela 5. Poređenje dobijenih vrednosti parametara naponske reference sa drugim radovima

Referenca	V _{DD} [V]	V _{REF} [V]	S [ppm/°C]	P _{dis} [mW]	Površina [mm ²]	T [°C]	tehnologija [μm]
Ovaj rad	3,3	1,38	9,8	0,270	12,20	-40 - 120	0,35
[5]	1,2	1,09	147,0	0,0001	0,0294	-40 - 120	0,18
[6]	3,3	1,22	40,0	/	140,0	-50 - 150	0,35
[7]	1,4	0,80	41,5	0,0560	0,0094	-40 - 85	0,04
[8]	1,2	0,63	29,0	0,060	0,0380	-10 - 100	0,13
[9]	3,3	1,21	32,2	/	31,80	-40 - 80	0,35

Nedostatak, prvenstveno, jeste velika površina koju kolo zauzima na čipu. Razlog tome jeste što su u kolu korišćene velike dimenzije otpornika.

5. ZAKLJUČAK

U ovom radu je projektovano kolo naponske reference koje obezbeđuje stabilan referentni napon od 1,38 V u širokom temperaturnom opsegu (-40 °C – 120 °C).

Vrednost temperaturnog koeficijenta u pomenutom temperaturnom opsegu, nakon post-lejaut simulacije, iznosi $TC=9,8 \text{ ppm/}^{\circ}\text{C}$ što predstavlja izuzetno dobar rezultat u poređenju sa drugim radovima [5]-[9].

Ukupna potrošnja kola je 0,27 mW. Zbog velike površine koju kolo zauzima usled velikih vrednosti korišćenih otpornika, predlog poboljšanja jeste topologija bez otpornika koja je prikazana u radu [10].

6. LITERATURA

- [1] Phillip E. Allen, Douglas R. Holberg: „CMOS Analog Circuit Design”, Oxford University Press, 2002.
- [2] Wenguan Li, Ruohu Yao and Lifang Guo: „A Low Power CMOS Bandgap Voltage Reference with Enhanced Power Supply Rejection”, IEEE 8th International Conference on ASIC, pp. 300-304, 2009.
- [3] Tony Carusone, David Johns, Ken Martin: „Analog integrated circuit design”, 2nd edition, John Wiley&Sons, 2012.
- [4] R. Jacob Baker: „Circuit design, layout, and simulation”, 3rd edition, IEEE press, John Wiley&Sons, 2010.
- [5] Yuji Osaki, Tetsuya Hirose, Nobutaka Kuroki, and Masahiro Numa: „1.2-V Supply, 100-nW, 1.09-V Bandgap and 0.7-V, Supply, 52.5-nW, 0.55-V Subbandgap Reference Circuits for

Tabela 4. Rezultati post-lejaut simulacija kola naponske reference

	Parametar	Vrednost
Naponska referenc	V _{REF(27 °C)} [V]	1,38
	S [ppm/°C]	9,8
	T [°C]	-40 – 120
	P _{dis} [mW]	0,27
Operacioni pojačavač	A _v [dB]	79,59
	P _{dis} [mW]	0,245
	PM [°]	80,24
	GBW [MHz]	9,26
	Opseg izlaznog napona [V]	0,15-3
	Opseg ulaznog napona [V]	0,2-2,75
	V _{OFFSET} [μV]	11,2
	SR ⁺ [V/μs]	+1,59 / 11,36

Prednost projektovane naponske reference je u tome što je postignuta jako mala osetljivost izlaznog napona 9,8 [ppm/°C] na promenu temperature u širokom temperaturnom opsegu (od -40 °C do 120 °C) nakon post-lejaut simulacije.

Nanowatt CMOS LSIs”, IEEE Journal of Solid-State Circuits, vol. 48, no. 6, June 2013.

- [6] Juan Pablo Martinez Brito, Hamilton Kllimach, and Sergio Bampi: „A Bandgap reference circuit in 0.35 μm CMOS technology with trimming array”, dostupno na adresi: <http://sbmicro.org.br/sforum-eventos/sforum2004/5005.pdf> (pristupljeno: 03/05/2018).
- [7] Chengyue Yu, and Lier Siek: „An Area-Efficient Current-Mode Bandgap Reference With Intrinsic Robust Start-Up Behavior”, IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 62, pp. 937-941, July 2015.
- [8] Dong-Ok Han, Jeong-Hoon Kim, and Nam-Heung Kim: „Design of Bandgap Reference and Current Reference Generator with Low Supply Voltage”, IEEE International Conference on Solid-State and Integrated-Circuit Technology, pp. 1733-1736, 2008.
- [9] Chao Feng, Jinhui Wang, Wei Wu, Ligang Hou, and Jianbo Kang: „CMOS 1.2V Bandgap Voltage Reference Design”, IEEE 10th International Conference on ASIC, pp. 1-4, 2013.
- [10] Charalambos M. Andreou, Savvas Koudounas and Julius Georgiou: „A Novel Wide-Temperature-Range, 3.9 ppm/°C CMOS Bandgap Reference Circuit”, IEEE Journal Of Solid-State Circuits, no. 2, vol. 47, February 2012.

Kratka biografija:



Dragan Žilić rođen je u Novoj Gradiški (Republika Hrvatska) 1986. godine. Diplomski rad je odbranio 2016. godine na Fakultetu tehničkih nauka u Novom Sadu iz oblasti Elektrotehnike i računarstva - Računarsko projektovanje digitalnih integrisanih kola.

AKTIVNO OPTEREĆENJE ZA ISPITIVANJE DC NAPAJANJA**ACTIVE LOAD FOR TESTING DC POWER SUPPLIES**Robert Fruža, Vlado Porobić, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U ovom radu predstavljen je razvoj aktivnog opterećenja za ispitivanje DC napajanja sa datim teorijskim osnovama. Realizovan je uređaj koji svojim mogućnostima može da odgovori na nekoliko zahteva između kojih su zadavanje opterećenja u bilo kojoj tački napajanja u ručnom i u automatskom režimu rada, iščitavanje vrednosti putem serijske veze, kao i rad kako u statičkom tako i dinamičkom režimu.

Abstract – This paper describes a research of active load for testing DC power supplies with theoretical foundations. Finished device can respond to several requests among which are setting point of load in manual as well as in automatic mode, reading measured value using serial data interface, with static and dynamic mode of operation.

Ključne reči: Aktivno opterećenje u lokalnoj tački napajanja, MOSFET tranzistor, linearni režim, projektovanje, HALL sonda, povratna sprega

1. UVOD

U idealnom slučaju, svako napajanje je invarijantno po amplitudi bez obzira na struju kojom se opterećuje, sve dok se ne prekorači maksimalna struja, kada napajanje ulazi u ograničenje zbog strujne zaštite. Međutim ovo nije slučaj u realnosti, jer napajanja pri različitim opterećenjima, pogotovo dinamičkim menjaju svoj izlazni napon, u zavisnosti od dinamike opterećenja koje se napaja. Ove varijacije prenapona ili podnapona u prelaznim režimima mogu biti problematične ako prelaze maksimalnu dozvoljenu granicu napona uređaja za koji je projektovano napajanje. Aktivno opterećenje se koristi, kada je potrebno izvršiti karakterizaciju napajanja koje se ispituje najčešće glavnog izvora koje napaja ceo uređaj, a to može biti regulisano linearno, prekidačko napajanje ili neregulisano sa diodnim ispravljačem. Za ispitivanje napajanja na samoj štampanoj pločici koja može imati dva, tri pa čak i više od deset naponskih nivoa koristi se posebno aktivno opterećenje kojim se vrši lokalno opterećenje na samoj tački na pločici, i to samo za određen naponski nivo. Ovi naponski nivoi su realizovani sa izvora različitih lokalnih DC/DC pretvarača, pa je zbog toga potrebno koristiti kako statičko tako i dinamičko opterećenje na samom mestu potrošnje u tački na štampanoj pločici [1].

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vlado Porobić, van.prof.

Ovi efekti posebno dolaze do izražaja pri malim naponima (0,8 V-3,3 V), koji se tipično sreću u napajanjima procesorskih jezgara i memorijskih modula. Savremene digitalne strukture po prirodi imaju dinamički složene karakteristike i potrošnju, a greške u projektovanim i izvedenim lokalnim napajanjima mogu imati za posledicu ograničenu funkcionalnost. Zbog toga se dešava pogrešno očitavanje podataka, ponovno pokretanje sistema i u najgorem slučaju uništenje komponente ili celog uređaja. Uzrok loše projektovanog napajanja je pojava oscilacija koje dovode do velikog termičkog naprezanja komponenta.

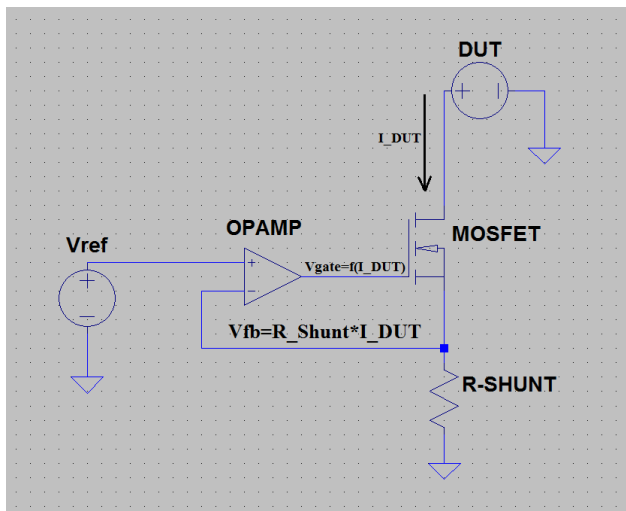
Aktivno opterećenje u odnosu na prekidačko opterećenje, ogleda se u tome što opterećenje koje je priključeno na napajanje sve vreme kontinualno, odnosno opterećenje se nalazi u linearnom režimu. Prekidačko opterećenje ima samo dva diskretna stanja uključeno i isključeno, gde se jačina struje menja faktorom ispunjenosti opterećenja. Kod aktivnog opterećenja struja se zadaje na dva načina, jednosmernom referencom koja je proporcionalna struji opterećenja za rad u statičkom režimu, a promenljivom referencom za rad u dinamičkom režimu. Realizovano opterećenje može da zada konstantna, impulsna i promenljiva strujna opterećenja, kao i da vrši merenja napona i struje. Kao laboratorijski merni uređaj može se koristiti za osnovnu karakterizaciju napajanih sklopova u realizovanim složenim digitalnim sistemima. Tako se na osnovu dobijenih rezultata merenja dobija karakteristika lokalnog napajanja i napojnih vodova na štampanoj pločici do potrošača. Na ovaj način ima se uvid u kvalitet i realne mogućnosti ispitivanog napajanja. Time se dobija detaljna slika o strujnim, termičkim naprezanjima, padu i prebačanju napona u samoj tački na kojoj se nalazi potrošač, kao i proveru celog sistema od izvora pretvarača na pločici do tačke opterećenja gde se nalazi lokalni potrošač [1-2].

2. ANALIZA PROBLEMA

Problem je razložen na dve glavne celine, energetski deo gde se vrši opterećenje napajanja konstantnom strujom, i druga celina koja obuhvata merenje napona i struje napajanja koje se ispituje.

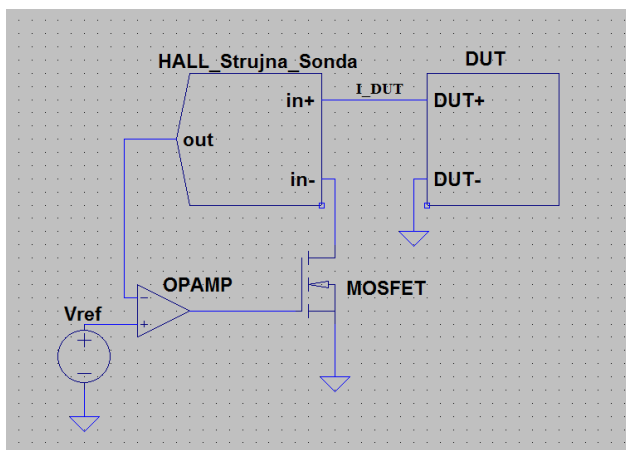
Energetski deo, se može uprošćeno prikazati na slici 1, gde se naponskom referencom na ne-invertujući ulaz, zadaje željena vrednost struje kroz napajanje pod opterećenjem. Na šant otporniku se razvija napon, koji je proizvod njegove poznate otpornosti i struje koja otiče kroz sors elektrodu tranzistora, odnosno struja kroz napajanje koje se ispituje DUT (eng. Device Under Test). Napon sa šant otpornika se dovodi na invertujući ulaz operacionog pojačavača, a kako bi se ispunio uslov da

ulazi operacionog pojačavača budu na istom potencijalu. Izlaz operacionog pojačavača menja napon direktno na gejtju koji održava MOSFET u aktivnom režimu i time čini zatvorenu negativnu povratnu spregu [3].



Slika 1. Pojednostavljena šema energetskeg dela

U ovoj izvedbi struja opterećenja je merena pomoću strujne sonde tipa Hall, kako bi se izbegli dodatni operacioni pojačavači sa šant otpornika i time realizovala direktna povratna sprega. Na slici 2. je data uprošćena šema energetskeg dela sa strujnom sondom.



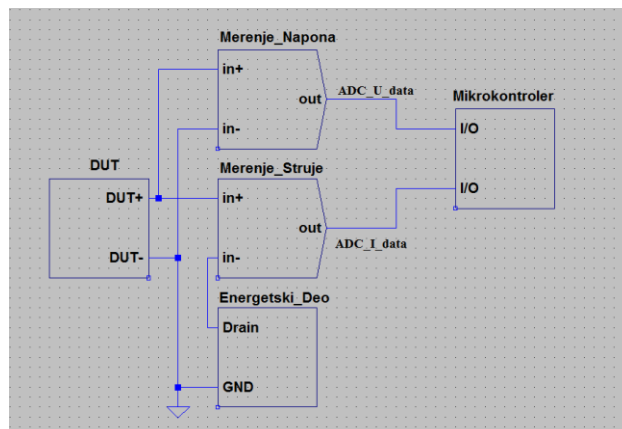
Slika 2. Energetski deo i merenje struje HALL sondom

Deo za merenje napona i struje zajedno kao pojednostavljena celina prikazan je na slici 3. Vršiti se merenje napona i struje napajanja koje se ispituje. Merenje struje vrši se spoljnim AD konvertorima, pomoću strujne sonde tipa Hall koja se koristi i za povratnu spregu po struji, dok se merenje napona vrši preko otporničkih razdelnika. Izlazi mernih blokova su spojeni na mikrokontroler digitalnim linijama. Kako bi se zadržala maksimalna rezolucija pri različitim naponima, koriste se signalni releji za odabir odgovarajućeg naponskog razdelnika, tako da odabrani opseg bude što bliži naponu napajanja koje se ispituje.

Zbog potrebe da se struja i napon simultano očitavaju, sa brzinom od 2 MSps i 12-bitnom rezolucijom, neophodno je koristiti spoljne A/D konvertore, a ne interne A/D konvertore mikrokontrolera zbog šuma koji bi uticao na analogni deo konvertora, a kako bi se ispunio ovaj zahtev

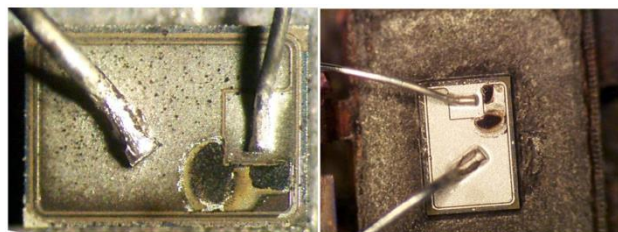
za brzinom očitavanja 2 MSps i 24 bita istovremeno (2x12bita), potrebno je koristiti A/D konvertore sa paralelnim izlazom.

D/A konvertor mora biti visoke rezolucije, zbog toga što MOSFET radi u linearnoj oblasti samo od 2 V do 5 V U_{GS} . Da bi u tom uskom opsegu ostala visoka rezolucija koristi se 16-bitni D/A konvertor, a pošto je maksimalna očekivana brzina do 1 KHz koristi se D/A konvertor sa SPI interfejsom. Rereferenca za konvertore je spoljna od 4,096 V pa se za 12-bitni konvertor dobija najmanja rezolucija od 1 mV.



Slika 3. Merenje napona i struje napajanja

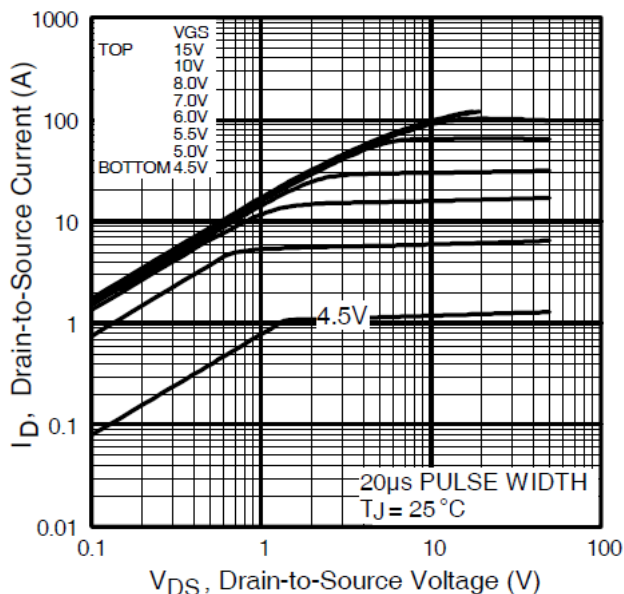
Zbog rada u aktivnom režimu, MOSFET-i se ponašaju kao bipolarni tranzistori. Potrebno je staviti otpornik male vrednosti (ispod 1 Ω) u kolo sors elektrode, kako bi se oborila strmina karakteristike aktivnog dela odnosno kako bi promena struje bila što linearnija sa promenom napona gejtja. Takođe neophodno je postaviti više MOSFET-a paralelno, jer se u linearnom režimu dozvoljava do 50 W disipacije po tranzistoru, zbog termičkog bežanja i negativnog temperaturnog koeficijenta u linearnom režimu kao i stvaranja vrućih tačaka koje mogu oštetiti ili potpuno uništiti tranzistor kao na slici 4. Ovo pravilo važi bez obzira na snagu tranzistora, tip kućišta i SOA (eng. Safe Operating Area) karakteristiku, koja odražava isključivo prekidački režim rada MOSFET-a. Pozitivni temperaturni koeficijent kod MOSFET tranzistora važi samo za rad u prekidačkom režimu, gde čak ni pri paralelovanju više tranzistora nije potrebno koristiti otpornike u kolu sorsa za izjednačavanje struje sorsa tj zatvaranje lokalne povratne sprege po tranzistoru. Za predviđenu disipaciju uređaja od 150W, koristi se 4 tranzistora, pa je maksimalna disipacija 37,5 W po tranzistoru [4].



Slika 4. Uništen MOSFET stvaranjem lokalnih vrućih tačaka

Proširena karakteristika napona gejtja V_{GS} za MOSFET IRFP-250N, daje linearnu transkonduktansu za napone

gejta 2,5 V do 5 V slika 5, što odgovara maksimalnoj struji od 12 A po tranzistoru. Za četiri paralelna tranzistora, maksimalnu struju opterećenja 20 A, struja po tranzistoru iznosi 5 A, što je u granicama linearnog dela transkonduktanse, pa je tako omogućeno jednostavno zadavanje reference struje opterećenja.



Slika 5. Izlazna karakteristika MOSFETa IRFP-250N

Takođe bitna karakteristika pored transkonduktanse je linearnost tranzistora, kako bi se za različite napone V_{DS} odnosno za različite napone ispitnog napajanja, imala uvek ista struja koja teče kroz drejn-sors I_{DS} . Tako se za poznat napon gejta V_{GS} , dobija uvek poznata struja I_{DS} tj strujna opterećenja napajanja. Za napone do 1 V_{DS} , ovo pravilo ne važi pošto se MOSFET tada nalazi u omskom režimu, pa struja zavisi od otpornosti R_{DS} koja postaje dominantna pri malom naponu i nije moguće ostvariti maskimalnu struju za koje je dizajnirano opterećenje kao za napone preko 1 V.

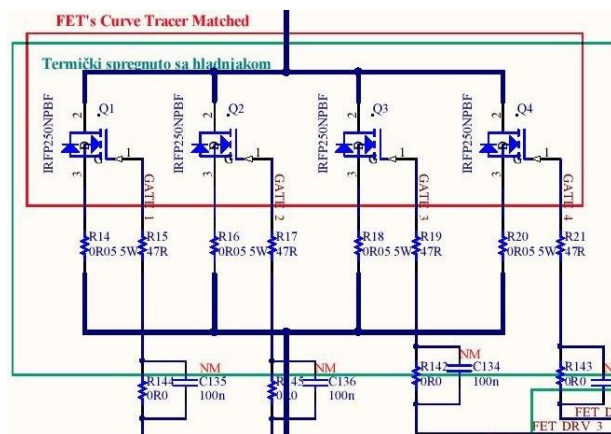
Kako se koristi više tranzistora u paralelnom radu i u aktivnom režimu, neophodno je izabrati tranzistore tako da budu upareni. Uparivanje tranzistora se vrši na traseru krivih Tektronix Tip-575, tako da se karakteristike ne razilaze više od 5% pri malim i pri velikim radnim strujama [3-4].

3. PROJEKTOVANJE ŠTAMPANE PLOČICE

Blok za napajanje deli se na dve celine AC/DC i DC/DC. Napon se spušta sa mrežnog naizmeničnog preko torusnog transformatora i potom se ispravlja diodnim ispravljačima u kombinaciji sa elektrolitskim kondenzatorima, gde se dobijaju naponi ± 15 V. Jednosmerni napon se dalje spušta preko linearnih regulatora na ± 12 V i ± 5 V. Linearni regulatori se koriste kako bi se obezbedio što čistiji napon za analogni deo tj bez talasnosti i šuma, a ujedno maksimalna očekivana struja u analognom delu je oko 100 mA po napajanju što je disipacija od 0,3 W za pad napona od 3 V za 12 V deo, odnosno 0,7 W disipacije za 5 V deo. Napajanje 3,3 V napaja mikrokontroler i digitalni deo A/D konvertora, urađeno je sa prekidačkim napajanjem spuštačem napona, zbog očekivane struje do maksimum 500 mA, jer bi pri

korišćenju linearnog napajanja disipacija regulatora iznosila 17,4 W.

U analognom bloku šeme, nalazi se energetski deo aktivnog opterećenja, drajverski deo, kao i merenje struje sa prilagodnim kolima povratne sprege. Izborom rada ručno ili preko mikrokontrolera, bira se izvor reference, koju analogni multiplexer prosleđuje drajverskom kolu. Drajversko kolo upravlja energetskim MOSFET tranzistorima Slika 6 koji se nalaze u aktivnom režimu i to pojedinačno jedan operacioni pojačavač za jedan MOSFET. Povratna sprega je kompenzovana i jedinična za sva četiri pojačavača, a dobija se sa strujne sonde, stim da je ostavljena mogućnost odabira povratne sprege direktno sa sonde, ofsetovane ili baferovane sprege. Diferencijalni pojačavač sa strujnog šanta, služi samo za proveru i kontrolu osciloskopom pri ispitivanju uređaja, a ne kao povratna sprega, jer bi neophodno bilo dodati još jedan pojačavač, pa se povećava mogućnost nestabilnosti povratne sprege tj pojave oscilacija i nastajanja pozitivne povratne sprege [5].



Slika 6. Analogni deo sa paralelovanim MOSFET tranzistorima

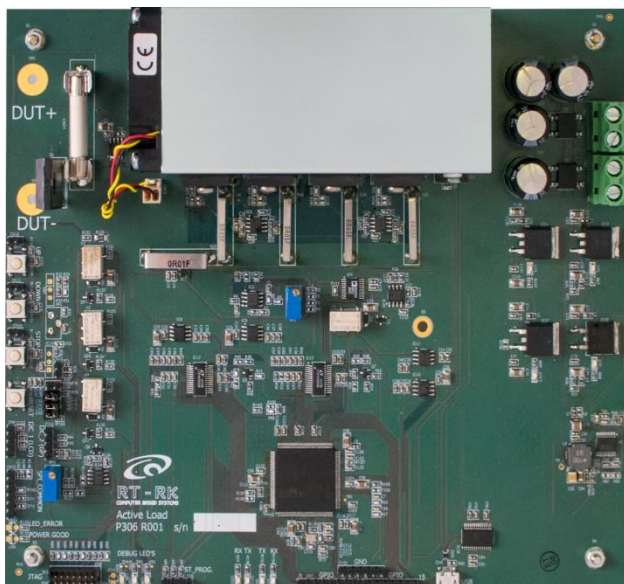
Iako su MOSFETi naponsko upravljane komponente, neophodno je dodati zasebna drajverska kola, kako bi se savladala kapacitivnost gejta C_{GS} , tj struja punjenja kapacitivnosti gejta, koja u prelaznim režimima predstavlja kratak spoj za D/A konvertor koji bi napajao MOSFET, samo dodavanjem otpornosti u kolo gejta ograničava se struja punjenja kapacitivnosti ali i brzina MOSFETa koju treba uzeti u obzir za slučaj dinamičkog rada opterećenja. Zbog toga je za drajversko kolo odabran linearni dupli operacioni pojačavač, za svaki MOSFET po jedan pojačavač. Na ovaj način poboljšana je dinamika cele povratne sprege uz zanemarljivo kašnjenje, sada po operacionom pojačavaču postoji samo jedna kapacitivnost gejta C_{GS} [4], [5].

Mikrokontrolerski blok sastoji se od mikrokontrolera i dela za USB komunikaciju sa računarom. Glavne funkcije mikrokontrolera su zadavanje reference struje za D/A konvertor, učitavanje vrednosti sa A/D konvertora, serijska komunikacija, ispisivanje na LCD i unos komandi sa tastera korisničkog interfejsa. USB komunikacija je ostvarena preko FT-232 integralnog kola i UART komunikacije sa mikrokontrolerom. Na mikrokontroleru su predviđene opcije za JTAG

programator, ST mikro programator kao i LED diode za proveru stanja mikrokontrolera.

Blok korisničkog interfejsa sastoji se od seta tastera, koji se direktno vezuju na portove mikrokontrolera, LED dioda za signalizaciju stanja sistema, i univerzalnog SPI-I²C modula sa LCD displejom i tastaturom sa 16 tastera.

Štampana pločica Slika 7 izrutirana je u četiri sloja, tako da su energetske vodove najveće širine zbog maksimalne struje od 20 A, a ujedno isti vodovi moraju biti što kraći, kako bi otpor i zagrevanje voda bili što manji. Signalni vodovi i vodovi za napajanje su izrutirani u gornjem i donjem sloju, dok su unutrašnji slojevi mase kojima koje se referenciraju gornji i donji sloj. U prvom unutrašnjem sloju je izrutirano i 3,3 V napajanje za mikrokontroler. Debljina svih vodova je 75 μm , zbog velikih struja do 20 A u energetskom delu. Pored debljine i širine vodova za energetske deo koje diktira maksimalna struja koja kroz njih protiče, potrebno je i uskladiti impedansu vodova digitalnog dela na 50 Ω [6].



Slika 7. Štampana pločica sa zalemljenim komponentama

4. REZULTATI MERENJA

Merenja u statičkom režimu su vršena za struje 10mA, 1A, 10A i 20A, a u dinamičkom režimu za frekvenciju od 100Hz i 1KHz, pri struji 1A, faktorom ispune 50% i stepom 0A i 1A.



Slika 8. Statički režim, opterećenje 1A, napon 5V, 1A/pod.

Zadatkom referencom struje od 5 mA, uređaj postiže traženu struju, međutim strujni šum je suviše velik 10 mA, pa je zato srednja vrednost struje oko 5 mA, tj strujni šum je istog reda veličine kao i referenca pa je uređaj na granici stabilnosti. Tek za referncu struje od 10 mA pa sve do 20 A opterećenje radi stabilno sa znatno manjim strujnim šumom u odnosu na zadatu vrednost struje slika

8. Za minimalni napon od 0,75 V maksimalna moguća struja je oko 10A, gde postaje dominantna otpornost samog MOSFET tranzistora R_{DSon} , kao i uticaj otporničkih šantova.



Slika 9. Dinamički režim, struja 1 A, frekvencija 1 KHz, faktor ispune 50%, 500 mA/pod

U dinamičkom režimu, na slici 9 za frekvenciju od 100 Hz opterećenje prati referencu bez prebačaja, dok granična frekvencija sa prebačajem od 4,7% iznosi 1 KHz. Za veće frekvencije, prebačaj je znatno veći, a odziv postaje oscilatorno granično stabilan, sve do trenutka kada razlika izlaza operacionog pojačavača i povratne sprege ne postane 180°, kada sistem postaje nestabilan i nastaje pozitivna povratna sprega.

5. ZAKLJUČAK

U okviru ovog rada prikazano je rešenje i realizacija aktivnog opterećenja, sa projektovanjem električne šeme, štampane pločice, merenjima i ispitivanjima. Prednost ovog uređaja je što omogućava da se realizuje aktivno opterećenje na bilo kom mestu štampane pločice i ispita lokalno napajanje kao i svi vodovi, njihovo strujno i termičko naprezanje kao i padovi napona, od struja reda 10 mA do 20 A. Takođe u programskoj realizaciji pored zadavanja konstantne struje u statičkom i dinamičkom režimu, moguće je zadati konstantan otpor i konstantnu snagu. Time se znatno proširuje opseg primene uređaja kako sa lokalnih naponskih nivoa izvora na štampanoj pločici, tako i ispitivanje namenskih napajanja, solarnih ćelija, baterija, kondenzatora i gorivih ćelija.

6. LITERATURA

- [1] J. Williams, "Design of a 100A active load to test power supplies", EDN 09-22, 2011
- [2] J. Williams, "A closed loop, wideband 100A active load", Linear Technology, AN-133, 2011
- [3] R. Pease, "Troubleshooting analog circuits", National Semiconductor, EDN, 1993
- [4] A. Ball, "Thermal stability of MOSFETs", ON Semiconductor, AND-8199/D, 2005
- [5] J. Graeme, "OP Amp Performance Analysis", Burr-Brown, 2000
- [6] L. Ritchley, "PCB Routers and routing methods", PC design Magazine, 1999

Kratka biografija:



Robert Fruža rođen je u Novom Sadu 1985. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranio je 2018.god.

MOGUĆNOSTI SAVREMENIH MOTORNIH ZAŠTITNIH PREKIDAČA

FUNCTIONALITIES OF MODERN MOTOR PROTECTION CIRCUIT BREAKERS

Miloš Stevanović, Dragan Milićević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ENERGETSKA ELEKTRONIKA I ELEKTRIČNE MAŠINE

Kratak sadržaj – U ovom radu prikazana je analiza mogućnosti upotrebe savremenih zaštitnih prekidača. Akcenat je stavljen na određena savremena rješenja koja pružaju novije generacije motornih zaštitnih prekidača u električnim kolima elektromotornih pogona. Kroz nekoliko sprovedenih eksperimenata u laboratorijskim uslovima prikazane su i analizirane neke od naprednih zaštitnih funkcija prekidača jednog od vodećih proizvođača u ovoj oblasti. Rezultati eksperimenata su detaljno obrazloženi i priloženi u vidu grafičkih reprezentacija relevantnih veličina.

Ključne reči: *Savremeni prekidač, Zaštitne funkcije, Električne mašine, Elektromotorni pogon.*

Abstract – This paper presents the possibilities of using modern protective circuit breakers. Emphasis is placed on certain contemporary solutions that are provided by newer generations of motor protection circuit breakers. Some of the advanced protective functions of circuit breakers, produced by one of the leading manufacturers in this field, have been shown and analyzed through several experiments carried out in laboratory conditions. The results of the experiments are clarified in detail and they are shown in the form of graphic presentations of relevant responses

Keywords: *Modern circuit breaker, Protective functions, Electrical machines, Electromotor drive.*

1. UVOD

U elektromotornom pogonu bez adekvatne zaštite, struje kratkog spoja i struje preopterećenja mogu ugroziti ljudske živote i opremu samog pogona. Stoga je jedan od osnovnih ciljeva prepoznavanje ovakvih kritičnih situacija i pravovremena reakcija na iste.

Danas se kombinacijom sklopnih elemenata sa odgovarajućim vrstama releja, senzora i savremenih elektronskih uređaja može dobiti izuzetno efikasan i pouzdan sistem zaštite elektromotornog pogona koji doprinosi prije svega povećanju bezbjednosti ljudi, a pored toga povećava raspoloživost i pouzdanost samog pogona.

Jedan od najčešće prisutnih sklopnih i zaštitnih elemenata jeste niskonaponski (NN) prekidač.

2. SAVREMENI NISKONAPONSKI PREKIDAČI

Niskonaponski prekidač je mehanički sklopni aparat koji je osposobljen da uklapa, provodi i prekine struju u normalnim uslovima rada, dok u uslovima preopterećenja i kratkog spoja ima sposobnost da uklopi, određeno vrijeme provodi i prekine struju kvara. Ima dvije osnovne radne funkcije: uklopnu/isklopnu i zaštitnu.

Niskonaponski prekidači mogu se razlikovati prema različitim kriterijumima. Prema načinu mehaničke izvedbe mogu se podijeliti na vazdušne, kompaktne i automatske instalacione, prema načinu ugradnje u razvodne ormare mogu biti izvlačivi, utični i fiksni, a prema broju polova jednopolni, dvopolni, trolpolni i četvoropolni. Na početku svog razvoja mogućnosti niskonaponskih prekidača bile su sasvim skromne s obzirom da su imali sposobnost realizovanja svega dvije do tri osnovne zaštitne funkcije (zaštita od struje kratkog spoja, preopterećenja i preniske vrijednosti napona). Moć prekidanja struje kratkog spoja bila je mala tako da su u električnom kolu zahtijevali i neke dodatne sklopne i zaštitne uređaje. Daljim usavršavanjem prevaziđena su početna ograničenja i danas se može konstatovati da savremeni prekidači svojim osobinama nadmašuju zaštitne komponente sa sličnim ulogama u električnim kolima i istovremeno pružaju znatno veći broj zaštitnih funkcija i dodatnih operacija koje značajno povećavaju mogućnosti prekidača, pouzdanost napajanja i kvalitet zaštite šticeenog uređaja. Široka paleta dodatne opreme omogućava jednostavno prilagođavanje ovih prekidača svim uslovima koji se javljaju u pogonu. S obzirom da se vazdušni i kompaktni niskonaponski prekidači često koriste u električnim kolima elektromotornih pogona i tehnološki su značajno napredovali u odnosu na instalacione automatske, u nastavku će biti dato nešto više informacija o ove dvije vrste niskonaponskih prekidača.

Vazdušni prekidači ili ACB (eng. *Air Circuit Breakers*) su NN prekidači koji se proizvode za velike nazivne struje, do 6300 A [1]. Kako bi izdržali velike mehaničke sile i veliko zagrijavanje koje se javlja pri prekidanju velikih struja kvara izvedeni su mehanički vrlo robusno. Za razliku od kompaktnih i automatskih instalacionih prekidača pogodni su za održavanje budući da se sklapaju od više dijelova koje je tokom održavanja usljed neispravnosti moguće jednostavno zamijeniti.

Ovi prekidači koriste se za uključivanje i zaštitu motora, generatora, transformatora, sabirnica, kablova, itd. Vazdušni prekidači su posebno fleksibilni i sposobni za komunikaciju sa drugim uređajima.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Milićević, prof.

Kompaktni prekidači ili MCCB (eng. *Moulded Case Circuit Breakers*) su NN prekidači smješteni u kompaktna kućišta od izolacionog materijala. Kućište onemogućuje neposredan dodir dijelova pod naponom. Nisu predviđeni za održavanje i popravku, te je u slučaju neispravnosti bilo kojeg dijela potrebno zamijeniti cijeli prekidač. Proizvode se za nazivne struje do 3200 A [1].

S obzirom na činjenice da pružaju veliki broj zaštitnih funkcija, imaju velike mogućnosti podešavanja zaštita, upravljanja i prenosa podataka, u današnje vrijeme sve češće se upotrebljavaju kompaktni NN zaštitni prekidači u kombinaciji sa elektronskim okidačima, odnosno mikroprocesorski upravljanim zaštitnim jedinicama.

2.1. Okidači niskonaponskih prekidača

U sklopu niskonaponskih prekidača koriste se posebni elementi za mehaničko okidanje prekidača u slučajevima kada se na mjestu ugradnje prekidača pojave neželjene vrijednosti struje ili napona. Ovi elementi se nazivaju okidači i u zavisnosti od primjene prekidač može biti opremljen sa više različitih vrsta okidača, a to su [2]:

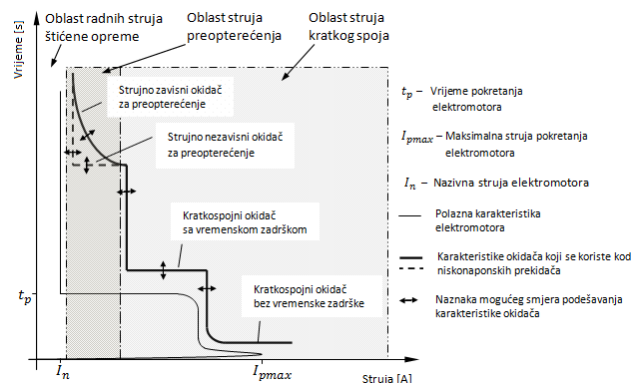
1. Prekostrujni okidač za zaštitu od preopterećenja:
 - strujno zavisni okidač za preopterećenje,
 - strujno nezavisni okidač za preopterećenje.
2. Prekostrujni okidač za zaštitu od kratkog spoja:
 - kratkospojni okidač s vremenskom zadržkom,
 - kratkospojni okidač bez vremenske zadržke.
3. Podnaponski okidač za zaštitu od pojave preniskih vrijednosti napona.

Kao ilustrativni primjer zahtjeva koji prekidači moraju ispuniti, na slici 1 dat je principski prikaz prilagođavanja karakteristika reagovanja pojedinačnih zaštitnih okidača polaznoj karakteristici asinhronog motora.

Na slici 1 može se vidjeti da postoje tri različite oblasti u zavisnosti od vrijednosti struja koje dostiže štice uređaj (u ovom slučaju asinhroni motor). Prva oblast odnosi se na vrijednosti radnih struja štice opreme, tako da u ovoj oblasti ne smije doći do isključenja prekidača. U oblasti struja preopterećenja podešava se karakteristika reagovanja okidača za preopterećenje, dok se u oblasti struja kratkog spoja podešava karakteristika reagovanja kratkospojnog okidača. Kod ovih okidača u određenom opagu može se podešavati proradna struja okidača kao i vrijeme trajanja struje preopterećenja i struje kratkog spoja.

Prema tome, postoji podjela niskonaponskih prekidača i prema ugrađenim okidačkim jedinicama, odnosno vrstama okidača koje posjeduju, na prekidače sa hidro-magnetskim, termo-magnetskim, magnetskim (bez termičke zaštite) i mikroprocesorskim okidačima. Standardni niskonaponski prekidači se najčešće proizvode sa analognim okidačima, tako da se okidač za zaštitu od struje kratkog spoja realizuje kao elektromagnetni element, dok se okidač za zaštitu od preopterećenja realizuje kao bimetalni element.

U posljednje vrijeme u elektromotornim pogonima sve veću ulogu imaju savremeni niskonaponski prekidači sa mikroprocesorskim okidačima. U okviru jednog savremenog niskonaponskog prekidača mogu biti implementirane zaštitne funkcije koje pokrivaju najširi mogući opseg nepovoljnih strujnih i naponskih uslova koji se mogu pojaviti na mjestu ugradnje zaštitnog prekidača.



Slika 1. Principski prikaz prilagođavanja karakteristika reagovanja zaštitnih okidača [3]

Što se tiče zaštitnih funkcija za zaštitu električnih mašina u elektromotornim pogonima, u okviru jedne savremene elektronske okidačke jedinice pored standardnih zaštitnih funkcija (zaštita od preopterećenja, kratkog spoja, gubitka faze, debalansa struja) mogu biti implementirane i neke dodatne napredne zaštitne funkcije (zaštita od zemljospoja, oscilovanja tereta, ukočenog rotora, preniskih vrijednosti struja, PTC zaštita).

3. LABORATORIJSKO ISPITIVANJE NAPREDNIH ZAŠTITNIH FUNKCIJA PREKIDAČA ABB SACE TMAX XT4N, M-LRIU

U nastavku će biti prikazano laboratorijsko testiranje dve od ukupno devet zaštitnih funkcija koje posjeduje elektronska okidačka jedinica za zaštitu elektromotora, koja nosi oznaku Ekip M-LRIU, u okviru savremenog zaštitnog prekidača SACE Tmax XT4N 250. Rezultati laboratorijskih ogleda biće prikazani u obliku odgovarajućih snimljenih signala struje. Takođe, izvršena su određena softverska testiranja ovih zaštitnih funkcija, a rezultati ovih testiranja će zbog ograničenog broja stranica biti izostavljeni. Softversko testiranje poslužilo je za bolje razumijevanje načina funkcionisanja algoritama određenih zaštitnih funkcija, a rezultati dobijeni ovim testiranjem poslužili su kao referentne vrijednosti za poređenje sa rezultatima dobijenim u laboratorijskim ogledima.

3.1 Ispitivanje funkcije zemljospojne zaštite

Kako bi se razumjela logika detekcije i rada zaštitne komponente po pitanju zemljospojne zaštite, izvršen je niz softverskih testova. U prvim testovima su zadavane različite efektivne vrijednosti linijskih struja, dok su fazni stavovi ostali nepromjenjeni, a u kasnijim testovima promjenjeni su fazni stavovi za iste efektivne vrijednosti struja.

Posmatralo se stanje prekidača i stvarno vrijeme reagovanja zaštitne jedinice kao i efektivna vrijednost kompleksne rezultantne struje nastale kao zbir kompleksnih linijskih struja. Primećeno je da ukoliko efektivna vrijednost rezultantne struje premaši zadatu struju reagovanja (u analiziranom slučaju 8 A), funkcija za zaštitu od zemljospoja će dati nalog zaštitnoj jedinici prekidača da prebaci prekidač u okinuto stanje. Takođe, došlo se do zaključka da na efektivnu vrijednost rezultantne struje utiče promjena i efektivne vrijednosti i faznog stava struja kroz sva tri pola zaštitnog prekidača.

Dakle, za razliku od algoritama u okviru drugih zaštitnih funkcija prekidača koji djeluju na osnovu pojedinačno izmjerenih vrijednosti linijskih struja kroz polove prekidača, algoritam ove zaštitne funkcije vrši vektorsko sabiranje izmjerenih struja i djeluje na osnovu efektivne vrijednosti dobijene rezultatne struje.

Na slici 2 je pored karakteristika okidača za preopterećenje (kriva crvene boje) i kratkog spoja (kriva žute boje) prikazana i karakteristika okidanja zemljospojnog okidača (kriva plave boje), pri podešenoj struji reagovanja od $0,2 \cdot I_n$ i vremenu reagovanja 0,4 s.

Prema ovoj karakteristici, zemljospojni okidač bi pri rezultatnoj struji od $0,1 \cdot I_n$ trebao da reaguje za dvije sekunde, sa druge strane za rezultatne struje veće od $0,3 \cdot I_n$ ovaj okidač bi trebao da reaguje za četiri sekunde, naravno treba uvažiti i propisano vrijeme tolerancije koje u ovom slučaju iznosi $\pm 15\%$ od očitnog vremena [4].

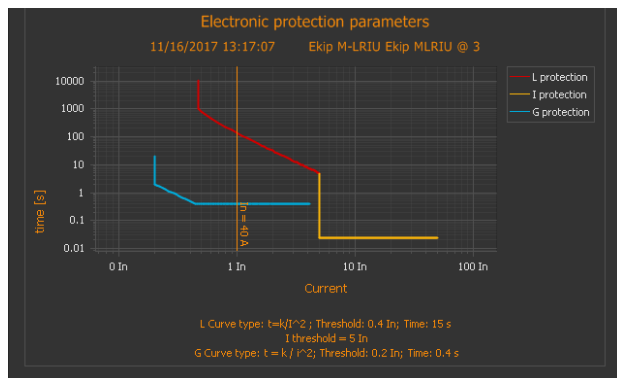
U laboratorijskom ogledu pored pomenutog zaštitnog prekidača i kontaktora za manipulaciju rada asinhronne mašine upotrebljena je i sledeća laboratorijska oprema: asinhrona mašina AM1 (slika 3.a.), monofazni autotransformator (slika 3.b.) i trofazni suvi transformator (slika 3.c.).

U ogledu je formirano glavno i pomoćno strujno kolo. Glavno trofazno strujno kolo preko koga se napaja asinhrona mašina formirano je od zaštitnog prekidača i kontaktora za manipulaciju rada pogonske mašine. Pomoćno strujno kolo je formirano od monofaznog autotransformatora sa koga je napojena jedna faza trofaznog suvog transformatora, tako da je dobijeno pomoćno monofazno strujno kolo.

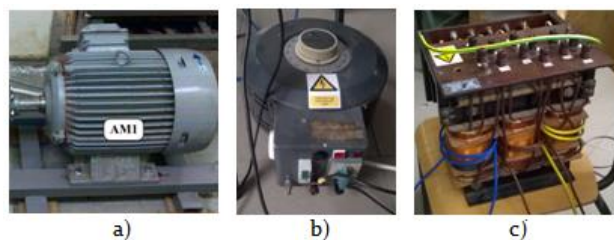
Struja reagovanja zaštitne funkcije podešena je na $0,2 \cdot I_n$ odnosno 8 A, a vrijeme reagovanja na 0,2 s. Na početku ogleda asinhrona mašina je preko glavnog strujnog kola startovana direktnim priključenjem na mrežu, a potom puštena da radi u motorskom praznom hodu.

U ovom periodu kroz sva tri pola prekidača protiče struja čija je amplituda približno 15 A, odnosno $10,6 \cdot I_n$ efektivne vrijednosti, što se može vidjeti na slici 4 u prvih 1,2 s, gdje je prikazana snimljena struja napojne faze asinhronog motora u kojoj je oponašan zemljospoj.

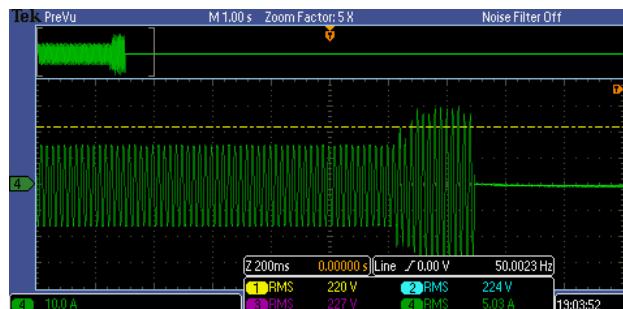
Ove vrijednosti struja kroz polove prekidača izazvale su djelovanje zaštite od zemljospoja nakon 0,22 s, odnosno prekidački mehanizam je isključio asinhronu mašinu sa napajanja.



Slika 2. Karakteristike okidanja zemljospojnog okidača, okidača za preopterećenje i kratkospojnog okidača



Slika 3. Izgled upotrebljene opreme u laboratorijskom ogledu za ispitivanje funkcije za zaštitu od zemljospoja



Slika 4. Uvećan signal struje koja protiče kroz provodnik faze u kojoj je oponašan zemljospoj

napajanja. Vrijednosti struja kroz sva tri provodnika opale su na nulu, što se može vidjeti posmatranjem samo jedne od tri struje na slici 4. S obzirom da je zemljospojna zaštita reagovala za 0,22 s, stvarno vrijeme reagovanja poklapa se sa definisanim.

3.2 Ispitivanje funkcije za zaštitu asinhronne mašine od oscilovanja tereta

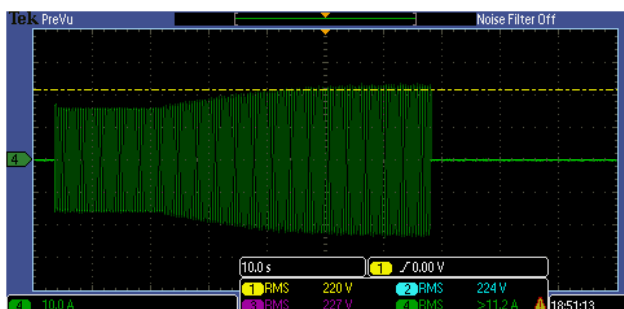
Sličnom procedurom kao i u slučaju prvog eksperimenta, nizom softverskih testiranja se pokušala shvatiti priroda rada ove zaštite. Iz sprovedenih softverskih testova došlo se do zaključka da ukoliko su struje u sve tri faze jednake sa podešenom strujom reagovanja ova zaštitna funkcija neće reagovati, što važi i za vrijednosti struja koje su manje od podešene struje reagovanja. S druge strane, ukoliko je efektivna vrijednost bilo koje od tri struje veća od podešene struje reagovanja, zaštitna funkcija će reagovati u definisanom vremenskom intervalu.

U cilju praktične provjere ove zaštitne funkcije sproveden je odgovarajući laboratorijski ogled. Pored zaštitnog prekidača i kontaktora za manipulaciju rada pogonske mašine upotrebljena je i dodatna laboratorijska oprema: pogonska i opteretna asinhrona mašina čija su vratila mehanički spregnuta, kao i regenerativni frekventni pretvarač neophodan za dovođenje testirane grupe u odgovarajuće pogonsko stanje. Više informacija o pomenutom pretvaraču može se naći u literaturi [5].

Pogonska mašina je napajana preko glavnog strujnog kola, a opteretna je napajana i upravljana preko regenerativnog frekventnog pretvarača. Frekventni pretvarač je podešen da radi u brzinskom režimu rada, odnosno da reguliše brzinu opteretne mašine (samim tim i brzinu obtanja pogonske mašine, s obzirom da su mašine mehanički spregnute) na osnovu zadate reference brzine koju unosi korisnik. U prvom ogledu struja reagovanja podešena je na 16 A, a vrijeme reagovanja zaštite na 10 s. Na početku ogleda frekventnom pretvaraču je zadata referentna vrednost brzine opteretne mašine na 750 o/min

(sinhrona brzina pogonske mašine) i data je komanda za njeno pokretanje. Nakon što je rotor opteretne mašine zajedno sa rotorom pogonske mašine postigao zadatu brzinu, startovana je pogonska mašina. Na ovaj način izvršen je takozvani sinhroni ili meki start pogonske mašine u cilju smanjenja polazne struje. Na slici 5 može se vidjeti snimljena struja kroz jedan pol prekidača od trenutka uključenja do isključenja napajanja pogonske mašine.

Dakle, nakon uključenja napajanja, pogonska mašina nastavlja da se obrće sinhronom brzinom narednih 17 sekundi, a momenti koje razvijaju mašine djeluju u istom smjeru. Kroz polove prekidača protiče struja amplitude oko 16 A što se može vidjeti na slici 5, odnosno oko 11,3 A efektivne vrijednosti, što je manje od podešene struje reagovanja zaštitne funkcije. Nakon toga, referenca brzine smanjena je na 730 o/min. Zadaivanjem nove referentne vrijednosti brzine frekventni pretvarač preko opteretne mašine nastoji da smanji brzinu sa 750 o/min na 730 o/min postepenim povećavanjem momenta opteretne mašine u suprotnom smjeru od momenta pogonske mašine. Ovo povećanje opterećenja na vratilu pogonske mašine nastoji da kompenzuje postepenim povećanjem momenta, odnosno struje koju preuzima iz mreže kroz zaštitni prekidač. U određenom trenutku postiglo se ustaljeno stanje, brzina obrtanja je smanjena na 730 o/min, a struja kroz pol prekidača se ustalila na približno 15,5 A efektivne vrijednosti. Pošto su struje kroz polove prekidača i dalje manje od podešene granične vrijednosti, referentna brzina je sada smanjena na 720 o/min. Rezultat toga je dodatno povećanje momenta opteretne mašine u suprotnom smjeru u cilju postizanja nove zadate brzine. Brzina obrtanja rotora mašina nastavlja postepeno da opada, a usljed dodatnog povećanja opterećenja na vratilu, pogonska mašina iz mreže preuzima još veću struju da bi obezbijedila jednakost momenata. U određenom trenutku efektivne vrijednosti struja kroz polove prekidača prelaze vrijednost zadate struje reagovanja od 16 A. Na slici 5 je to vremenski interval u kome amplituda struje iznosi približno 23 A, što je oko 16,2 A efektivne vrijednosti i nastavlja da raste jer nije postignuto novo stacionarno stanje. U trenutku kada je efektivna vrijednost struje kroz polove prekidača prešla zadatu struju reagovanja, zaštitna jedinica detektuje stanje potrebe za reakcijom zaštite, a nakon podešenog vremena od 10 sekundi dolazi i do reakcije isključenja prekidača, što se na slici 5 vidi naglim opadanjem struje na nulu.



Slika 5. Snimljena struja u toku oglada kroz jedan pol zaštitnog prekidača u realnom vremenu

4. ZAKLJUČAK

Savremeni niskonaponski prekidači nude novu tehnologiju, jednostavan rad i uštedu prostora. Posjeduju mogućnost komunikacije sa drugim elektronskim uređajima, što otvara niz novih mogućnosti kao što su brza parametrizacija, dijagnostika, upravljanje, održavanje. Mogućnost komunikacije na više načina smanjuje troškove i povećava produktivnost štice postrojenja, pogona ili objekta. Posjeduju jako jednostavno i fleksibilno kako manuelno tako i softversko podešavanje zaštitnih parametara uz mogućnost uključenja/isključenja pojedinačnih zaštitnih funkcija. Mogućnost daljinske kontrole i upravljanja prekidačem i nadzora određenih električnih veličina u kolu i mogućnost elektronskog testiranja zaštitnih funkcija korisniku značajno olakšavaju upotrebu ovog prekidača i povećavaju pouzdanost i sigurnost elektromotornog pogona.

5. LITERATURA

- [1] Javor Škare, Miroslav Mesić, "Zaštitni uređaji i selektivnost u podsustavima istosmjernog razvoda i napajanja", Zagreb, 2009 god. dostupno na: <https://www.srce.unizg.hr/>
- [2] Strahil J. Gušavac, "Osnovni principi projektovanja u mrežama srednjeg i niskog napona", Fakultet tehničkih nauka u Novom Sadu, 2014. god.
- [3] Miloš Stevanović, "Metode za puštanje u rad trofaznih kaveznih asinhronih motora (analiza teorijskih i praktičnih aspekata)", diplomski rad, Fakultet Tehničkih Nauka u Novom Sadu, 2017. god.
- [4] ABB, "SACE Tmax XT New low voltage moulded-case circuit-breakers up to 250A -technical catalog", Autorsko pravo ABB 2011. god., dostupno na: <https://library.e.abb.com>
- [5] Goran Krstić, "Primena regenerativnog frekventnog pretvarača u elektromotornom pogonu sa asinhronim motorom", diplomski – master rad, Fakultet Tehničkih Nauka u Novom Sadu, 2010. god.

Kratka biografija:



Miloš Stevanović rođen je 1993. god. u Trebinju. 2012 godine upisuje osnovne akademske studije na FTN-u Novi Sad, smer Energetika, elektronika i telekomunikacije. 2017. godine upisuje master akademske studije na istom fakultetu, smer Energetska elektronika i električne mašine.



Dragan Milićević rođen je 1977. god. u Tuzli. U zvanje docenta Univerziteta u Novom Sadu je izabran 2014. god. Oblast interesovanja su višefazne mašine, elektromotorni pogoni i obnovljivi izvori električne energije.

АПЛИКАТИВНО РЕШЕЊЕ КАО ИНТЕЛИГЕНТНИ САВЕТНИК ЗА ТРГОВИНУ АКЦИЈАМА НА БЕРЗИ**APPLICATIVE SOLUTION AS AN INTELLIGENT ADVISOR FOR STOCK TRADING**

Борко Цвејић, Факултет техничких наука, Нови Сад

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – У овом раду је представљено апликативно решење као помоћ приликом трговања акцијама на Београдској берзи. Коришћена је неуронска мрежа са простирањем сигнала унапред, а апликација је развијана у програмском окружењу MATLAB.

Кључне речи: Берза, Неуронске мреже, Предикција цене

Abstract – In this paper, an application that assists in stock trading on Belgrade stock market is presented. A feed-forward neural network was used, and the application was developed in the MATLAB programming environment.

Keywords: Stock market, Neural network, Price prediction

1. УВОД

Предвиђање вредности акција на берзи је важан финансијски проблем који добија све више пажње. Берза је постала место инвестирања не само за стратешке инвеститоре већ и за обичне људе. Главна карактеристика свих берзи је њена неизвесност [1].

Предвиђање подразумева разумевање на основу којих променљивих можемо предвидети вредност осталих променљивих. Потребно је схватити статистичко значење различитих променљивих као и одредити најзначајније из угла предвиђања изгледа тржишта.

У овом раду је представљена апликација која служи као помоћ при трговини хартијама од вредности на Београдској берзи. Истраживање покрива пет хартија од вредности које се у тренутку истраживања налазе на prime listing-у Београдске берзе: AERO, NIIS, KMBN, MTLC и JESV.

За предвиђање кретања цена акција коришћена је вештачка неуронска мрежа (ВНМ) са простирањем сигнала унапред. Мрежа је обучавана Левенберг-Маркард алгоритмом са историјским подацима до 2017. године, а тестирана над подацима из 2017. године.

У студији случаја тргујући на основу препорука мреже, остварена је зарада од 23,7% на уложена средства у динарима на годишњем нивоу.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је др Владимир Бугарски, доцент.

2. ТЕХНИЧКИ ИНДИКАТОРИ

Техничка анализа је облик анализе тржишта хартија од вредности заснован на проучавању цена и обима трговања. То је процес којим се „на основу кретања цена акција у прошлости, формулише став о будућим токовима“ [2]. Аналитичари верују да инвеститори понављају понашање инвеститора који су им претходили. Пошто се понашање инвеститора толико често понавља, аналитичари верују да је могуће препознати и предвидети обрасце кретања цена на графиконима.

Статистичке и математичке методе на које се техничка анализа ослања називају се и методе филтрације јер покушавају да раздвоје шаблоне у кретању цена од привременог колебања [3]. Цену акције прати тренд који покушава да се одржи док се не деси догађај која мења баланс продаје и потражње. Емпиријски је показано да је праћење реакције тржишта и анализа техничких индикатора прецизнија и од најбољих анализа добро информисаних трговаца.

Појам техничке анализе обухвата све методе које се користе приликом праћења цена, а у циљу што јаснијег сагледавања тог догађаја. У овом раду, по узору на рад [4] у којем је тестирана оптимална комбинација индикатора, коришћени су следећи технички индикатори:

1. Прости помични просек (енг. Simple moving average - SMA)
2. Тежински помични просек (енг. Weighted moving average - WMA)
3. Експоненцијални помични просек (енг. Exponential moving average - EMA)
4. Индекс релативне снаге (енг. Relative strength index - RSI)
5. Моментум осцилатор (енг. Momentum oscillator - M).

SMA, EMA и RSI су рађени за 5 дана, M за 2 дана а код WMA су одабране тежине 0.9, 0.85, 0.8, 0.75, 0.7 да би дани у ближој прошлости имали већи утицај.

3. ВЕШТАЧКЕ НЕУРОНСКЕ МРЕЖЕ

Вештачке неуронске мреже (ВНМ) су софтверски или хардверски модели пројектовани на основу структуре и понашања биолошких неурона и нервног система. Чак и најнапредније неуронске мреже су лошије у односу на најпримитивнији облик живота, али и даље представљају одличан начин представе биолошких неуронских мрежа, конкретно мозга.

Основни елемент ВНМ је неурон. Снага везе између два неурона се назива тежина и њена вредност је важан параметар који помаже у обучавању ВНМ. Тежина одређује позицију и облик основних функција мреже [5,6]. Предвиђање континуалних променљивих се назива регресија, а предвиђање категоријских променљивих класификација.

Основни кораци ка оспособљавању неуронске мреже су: прикупљање и припрема података, дизајн мреже, обука мреже, валидација обуке и тестирање мреже.

4. СТУДИЈА СЛУЧАЈА

За предвиђање кретања цена на Београдској берзи изабрана је мрежа са простирањем сигнала унапред која има један улазни слој са 5 улазних података, један излазни слој са 2 неурона и један скривени слој у којем се налази 20 неурона. Коришћен је Левенберг-Маркард алгоритам за тренирање мреже, код које је максималан број итерација 1000, минимална вредност градијента 10^{-7} , грешка 0, а број промашаја валидације који представља број итерација где грешка над валидационим подскупом расте 6. За сваку акцију са берзе је обучавања посебна ВНМ. Свака мрежа је тренирана 10 пута, а у наставку се користи она које је дала најбољу тачност генерализације. За тренирање ВНМ коришћени су историјски подаци доступни на сајту Београдске берзе. У случају АЕРО и ЈЕСВ то је опсег од 2011. до 2016. год, НИИС од 2010. до 2016. год, КМБН од 2006. до 2016. год, а за МТЛС од 2005. до 2016. год.

Од преузетих историјских података, издвојени су и коришћени: датум трговања, цена на затварању, минимална дневна цена и максимална дневна цена. Мрежа предвиђа колика ће бити минимална и максимална цена деоница у наредних 5 дана за сваку од 5 акција. Средње вредности минималних и максималних цена се даље пореде са тренутним ценама деоница, а затим се рачуна трговином које акције ће се остварити највећа зарада и обавештава се корисник који онда одлучује да ли ће послушати савет неуронске мреже. На слици 1 је приказано поређење стварних максималних цена тестног подскупа и максималних цена које је неуронска мрежа предвидела за КМБН акцију.

Одлучивање која деоница је најисплативија за трговање се врши на основу поређења одступања средње вредности минималне и максималне цене од тренутне цене сваке деонице:

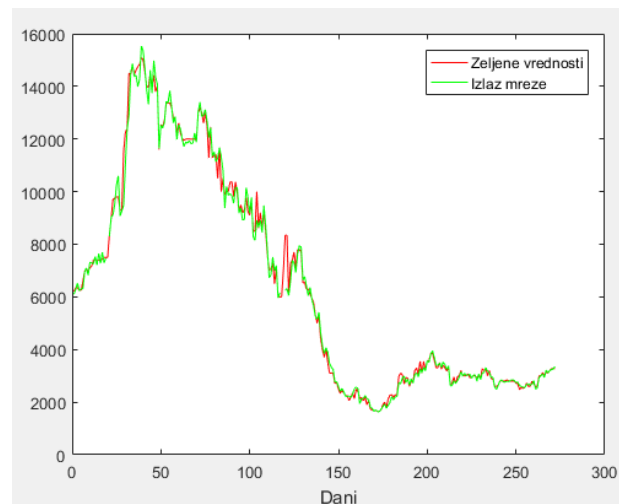
$$\text{Средња вредност} = (\text{Минимална цена} + \text{Максимална цена}) / 2 \quad (4.1)$$

$$\text{Одступање} = \text{Тренутна цена} - \text{Средња вредност} \quad (4.2)$$

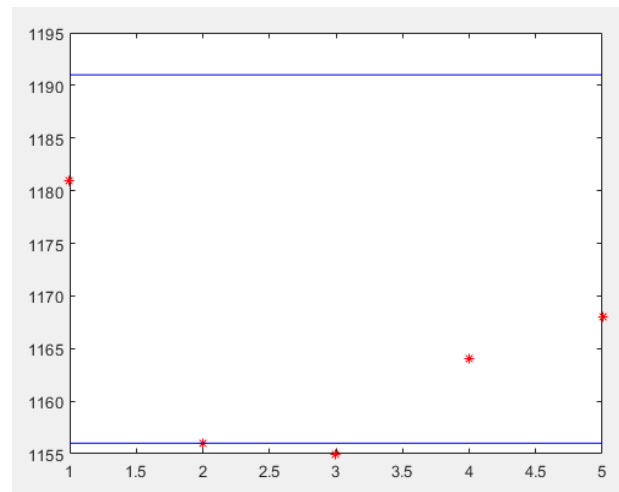
Уколико је највеће одступање негативно, то је знак да одређену акцију треба купити јер ће јој се цена повећати у будућности, у супротном апликација саветује да акцију продамо јер ће јој цена опасти.

У следећем примеру, на слици 2, види се да је неуронска мрежа погрешила предвиђање минималне цене АЕРО акције за само 1 динар, а максималну за 10 динара (црвене тачкице су стварне вредности а

плаве линије предвиђене границе). Ово је ипак довољно добро предвиђање јер је цена АЕРО акције на дан предвиђања износила 1174 динара, па је процењено да је ово добар тренутак за продају, јер је максимална предвиђена цена 1191 динар, а минимална предвиђена цена 1156 што даје средњу вредност 1173. Одступање из једначине (4.2) је позитивно и износи 1 динар, што предлаже продају деоница јер ће цена падати у наредним данима. Стварне минималне и максималне цене су износиле 1155 динара и 1181 динар што потврђује исправну одлуку да се деонице продају.



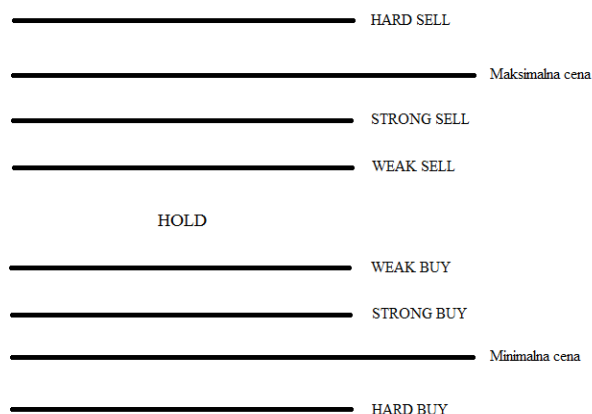
Слика 1. Поређење стварних цена са излазом мреже за КМБН акцију



Слика 2. Пример предвиђања неуронске мреже за АЕРО акцију

Студија случаја је рађена над периодом од годину дана, од 14. децембра 2016. до 15. децембра 2017. године. Врши се тестирање за различите параметре, а почетно улагање је једнако за све тестове и износи 1.000.000 (милион) динара. Виртуелно се тргује одређеним бројем акција за које је ВНМ израчунала да су најисплативије за тај дан. Трговање акцијама се извршава према дефинисаним границама: **HARD BUY**, **STRONG BUY**, **WEAK BUY**, **HARD SELL**, **STRONG SELL** и **WEAK SELL**. За сваку границу повезан је проценат новца који се одваја за куповину деоница одређене акције, док је код продаје, **HARD SELL**

резервисан за продају свих деоница одређене акције, **STRONG SELL** за продају половине, а **WEAK SELL** за продају четвртине деоница. На слици 3 је приказан пример граница које активирају продају или куповину деоница.



Слика 3. Границе активације трговања са **HARD** границама ван предвиђеног минимума и максимума

На пример, уколико се тренутна цена неке акције налази између **STRONG SELL** и **WEAK SELL**, то значи да ће апликација предложити извршавање **WEAK SELL** продаје деоница. Слично, уколико се тренутна цена налази између **STRONG BUY** и минималне цене, апликација ће предложити извршавање **STRONG BUY** куповине деоница.

Границе које активирају продају или куповину деоница се рачунају следећим једначинама:

$$\text{Разлика} = \text{Максимална цена} - \text{Минимална цена} \quad (4.3)$$

$$\text{WEAK BUY} = \text{Минимална цена} + \text{WB\%} * \text{Разлика} \quad (4.4)$$

$$\text{STRONG BUY} = \text{Минимална цена} + \text{SB\%} * \text{Разлика} \quad (4.5)$$

$$\text{HARD BUY} = \text{Минимална цена} - \text{HB\%} * \text{Разлика} \quad (4.6)$$

$$\text{WEAK SELL} = \text{Максимална цена} - \text{WS\%} * \text{Разлика} \quad (4.7)$$

$$\text{STRONG SELL} = \text{Максимална цена} - \text{SS\%} * \text{Разлика} \quad (4.8)$$

$$\text{HARD SELL} = \text{Максимална цена} + \text{HS\%} * \text{Разлика} \quad (4.9)$$

WB%, SB%, HB%, WS%, SS% и HS% су параметри којима се одређују вредности граница за трговање, а минимална и максимална цена су границе које предвиђа неуронска мрежа.

Вршени су тестови за различит број акција којима се тргује у једном дану. Такође, различита средства се издвајају за куповину деоница у зависности да ли се ради о најисплативијој акцији или нпр. трећој по реду. Уколико на крају дана остане мање од 100.000 (сто хиљада) динара, следећом куповином се троши сав преостали новац како се укупан број трансакција не би превише повећао на малим дневним трансакцијама. У свакој трансакцији су урачунате провизије укупне вредности 1.5% од износа трансакције у које спадају провизија брокерске куће, провизија банке и провизија Централног регистра,

депоа и клиринга хартија од вредности, у циљу симулације стварне продаје и куповине.

4.1. Резултати

Вршени су тестови за случајеве када се тргује са 2, 3, 4 и 5 најисплативијих дневних акција. Прво су вршени тестови за случајеве где се тргује са 4 најисплативије дневне акције. Пронађени су параметри за које је остварена зарада од 23,4% на уложена средства у динарима на годишњем нивоу. Затим су исти параметри коришћени за тестове са 2, 3 и 5 акција и добијена је нова највећа зарада од 23,7% при трговању са 3 најисплативије дневне акције.

Највећа зарада је остварена уколико су само **WEAK SELL** и **WEAK BUY** границе унутар предвиђених минимума и максимума, а резултати су приказани на графику 1. У тестовима који су рађени за 3 најисплативије дневне акције зарада често прелази 22%, док код тестова рађених са 2 најисплативије акције зарада најчешће не достиже ту вредност. Тестови рађени са 4 и 5 најисплативије акције такође могу да пређу зараду од 22% али резултати нису константни као код тестова са 3 акције.

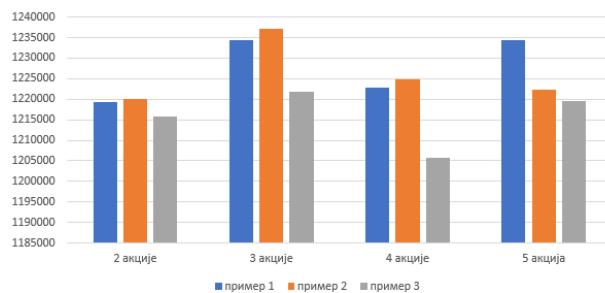


График 1. Стања на рачуну у зависности од броја акција којима је трговано

На графику 2 се може видети како висина улагања по акцијама утиче на зараду код тестова где се тргује са 3 акције. Ниски улози представљају улоге од 30% - 40% за најисплативију акцију који опадају за 5% или 10% за ниже границе, а високи од 60% - 80% који затим опадају за 5% или 10% за ниже границе. Улози од 90% дају слабије резултате од улога од 80%.

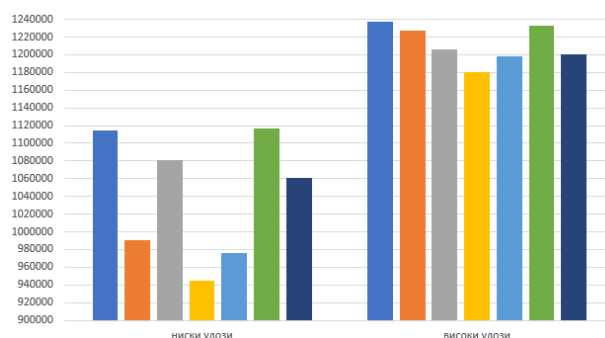


График 2. Приказ разлике између тестова са ниским и високим улозима

Вредности граница активације такође значајно утичу на зараду што се може видети на графику 3. Широк опсег значи (лево на графику) да су WS% и WB% параметри из једначина (4.4) и (4.7) постављени на 30% - 40% па обухватају шири опсег података, а ужи

опсег (десно на графику) значи да су ти параметри на 10% - 20%. Што се тиче *HS%* и *HB%* параметара из једначина (4.6) и (4.9), нема велике разлике да ли ће бити постављени на 5% или 15% изнад нпр. максималне предвиђене границе, јер број трансакција које се обаве под *HARD SELL* и *HARD BUY* границама није велики.

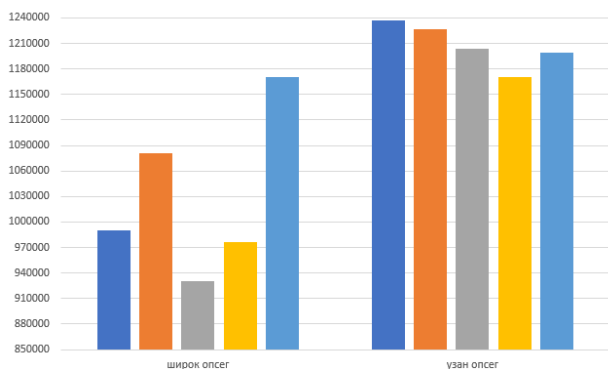
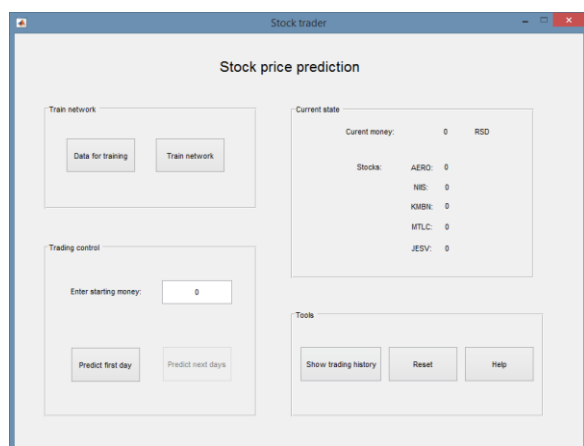


График 3. Разлике код тестова са широким и узким опсегом параметара граница активације

5. ГРАФИЧКИ КОРИСНИЧКИ ИНТЕРФЕЈС

За креирање графичког корисничког интерфејса коришћен је Matlab GUIDE. Приликом првог покретања програма потребно је унети са колико новца се жели трговати. Програм се покреће сваког дана када је берза активна или када корисник жели да тргује и служи као саветник приликом трговања. То значи да он неће аутоматски купити или продати деонице него ће предложити са колико деоница одређене акције је најисплативије трговати тог дана. Корисник затим одлучује да ли ће послушати савет или га одбити. Уколико корисник прихвати савет, број деоница одређене акције ће се променити као и новац којим се располаже што се може видети на главном прозору апликације. Све измене се складиште у Excel табели и могуће им је приступити и стећи увид у трансакције и остале детаље у било ком тренутку. Када се програм сутрадан покрене, тренутно стање остаје на њему па није потребно отварати историјску табелу да би се корисник подсетио како је завршио јучерашње трговање. На слици 4 је приказан изглед програма при првом покретању.



Слика 4. Изглед главног прозора при покретању

6. ЗАКЉУЧАК

Предвиђање правца кретања цена на берзи је веома значајно за развој стратегије трговања. То је компликован задатак који велики број трговаца све чешће покушава да олакша коришћењем ВНМ.

У овом раду је представљено апликативно решење као помоћ приликом трговине акцијама на Београдској берзи. Приказани су резултати тестова и начин на који апликација одлучује која је акција најисплативија. Трговање се вршило над пет акција које се у тренутку истраживања налазе на prime listing-у. За предвиђање кретања цена акција коришћена је ВНМ са простирањем сигнала унапред и остварена је зарада од 23,7% на уложена средства у динарима на годишњем нивоу.

Правци даљих истраживања могу бити тестирање ефикаснијих техничких индикатора у циљу повећања квалитета мреже и зараде, тестирање осталих алгоритама за обучавање неуронске мреже, као и олакшавање одабира којим акцијама корисник жели трговати. Како би апликација могла да тргује и другим акцијама са Београдске или неке иностране берзе, потребно је обучити неуронску мрежу историјским подацима за жељене акције.

7. ЛИТЕРАТУРА

- [1] A. D. Kachare, R. R. Kharde and A. D. Dongare, "Introduction to Artificial Neural Network," *International Journal of Engineering and Innovative Technology*, vol. vol. 2, July 2012..
- [2] M. Y. Qiu and Y. Song, "Predicting the direction of stock market index movement using an optimized artificial neural network model," 2016.
- [3] S. Slović, I. Raonić and I. Pejović, "Upravljanje ponudom i tražnjom na finansijskom tržištu pomoću tehničkih indikatora i oscilatora," *Pravno-ekonomski pogledi*, vol. 1, 2012.
- [4] S. Poletanović, *Sistem za podršku u odlučivanju kod berzanskog trgovanja zasnovan na veštačkim neuronskim mrežama*, Novi Sad: Univerzitet u Novom Sadu, 2017.
- [5] T. M. Mitchell, *Machine Learning*, McGraw-Hill Science/Engineering/Math, 1997.
- [6] V. Kecman, *Learning and Soft Computing*, The MIT Press, 2001.

Кратка биографија:



Борко Цвејић рођен је у Шапцу 1993. год. Основне студије Рачунарства и аутоматике на Факултету техничких наука уписао је 2012. год, а дипломску науку уписао је 2016. год. Исте године одбранио је мастер студије Рачунарство и аутоматика на Факултету техничких наука у Новом Саду.

UTICAJ DISTRIBUTIVNIH GENERATORA NA REGULACIJU NAPONA U DISTRIBUTIVNOJ MREŽI**IMPACT OF DISTRIBUTED GENERATION ON VOLTAGE REGULATION IN DISTRIBUTION NETWORK**

Uglješa Zelić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je razmatran uticaj distributivnih generatora na klasičnu regulaciju napona distributivne mreže, koja je realizovana preko regulacionog transformatora sa automatskim regulatorom napona. Pre svega, razmatra se uticaj odstupanja vrednosti stvarne struje potrošnje i struje koja se izmeri na strani sekundara regulacionog transformatora. Analiza posledica odstupanja te dve struje, lažna slika o opterećenju distributivne mreže, predstavlja osnovnu temu ovog rada. Program korišćen za realizaciju ovog rada razvijen je u programskom okruženju Microsoft Visual Studio 2010 i programskom jeziku FORTRAN.

Abstract – In this paper impact of distributed generation on voltage regulation is presented. It is done with regulation transformer by automatic voltage regulator. Difference in real load current and current that is measured and used as entry data for automatic voltage regulator is considered. Analysis of effects these two currents cause, bad image of distribution network load, is the main subject of this paper. The software tool for model verification is developed in Microsoft Visual Studio 2010 and programming language FORTRAN.

Ključne reči: regulacija napona, automatski regulator napona, regulacioni transformator, distributivni generatori.

1. UVOD

Potrošnja električne energije je sve veća. Nažalost, nju prati i sve veća zagađenost životne sredine. Da bi se zahtevi za povećanom potrošnjom ispunili, a emisija CO₂ smanjila, svetska elektroenergetika okrenula se prema obnovljivim izvorima električne energije. Njihovi izvori energije (vetar, sunce, plima, itd.) su praktično neiscrpn, njihova emisija štetnih gasova je zanemarivo mala. Ove elektrane se nalaze na distributivnim područjima, čime su značajno smanjeni gubici prenosa električne energije. Ovakav vid proizvodnje električne energije, zove se distributivna proizvodnja [1,2]. Prethodno navedene osobine predstavljaju osnovne razloge zašto broj instalacija obnovljivih izvora električne energije radikalno raste.

Nedostaci obnovljivih izvora su posledica pre svega davno izgrađene električne infrastrukture. Mreže su građene za prenos električne energije na velike razdaljine, od velikih udaljenih elektrana koje su izgrađene na malom broju lokacija. Pri tom, podrazumevalo se da je tok

električne energije uvek od prenosne mreže prema distribuciji, gde su krajnji potrošači. U skladu sa tim instalirani su i praktično podešeni svi koncepti i svi elementi elektroenergetskog sistema (regulacija napona, merenje, zaštita, itd.) [1,2]. Instalacijom distributivnih generatora (DG) u distributivnu mrežu (DM), ona od tradicionalno pasivne postaje aktivna mreža. U takvim okolnostima, višak proizvedene energije može da krene u drugom smeru, od distributivne ka prenosnoj mreži. Ovakvi, nepredviđeni režimi mogu da imaju za posledicu niz problema [1,2,3].

Regulacija napona DM predstavlja veliki i dobro poznat problem. Osnovni kriterijum pri rešavanju ovog složenog problema je ostvariti režim u kojem će svi potrošači imati napone najbliže moguće svojim nominalnim vrednostima, a da su pritom ti naponi u unapred definisanim granicama. Klasična regulacija napona se zasniva na automatskom regulatoru napona (ARN), koji u zavisnosti od izmerene struje na sekundaru regulacionog transformatora (RTr), u skladu sa zakonom regulacije, šalje impuls RTr da se promeni pozicija regulacione sklopke. Na ovaj način garantuje se da su naponi na potrošačima u DM bez DG, za sve unapred predviđene režime, u dozvoljenim granicama. Ako se u takvim DM priključi DG, ARN ima lažnu sliku o ukupnom opterećenju mreže za koju treba da izvrši regulaciju napona (struja izmerena na sekundaru RTr nije jednaka struji opterećenja DM). Ovaj problem je razmatran u većem broju radova [1,2,3].

Nakon Uvoda u drugoj glavi definisani su osnovni pojmovi vezani za razmatrani problem (potrošnja i klasična regulacija napona). Matematički model problema dvosmernog toka snage i porasta napona na izvodu sa DG razmatrani su u trećoj glavi. Njegova verifikacija izvršena je u četvrtoj glavi. Za proračune tokova snaga korišćen je Shirmohammadi-jev algoritam [4]. U petoj glavi je dat zaključak, a u šestoj glavi je referentno navedena literatura koja je korišćena za izradu ovog rada.

2. OSNOVNI POJMOVI

U ovom delu predstavljeni su osnovni pojmovi sa kojima se susreće pri analizi razmatranog problema.

2.1 Potrošnja

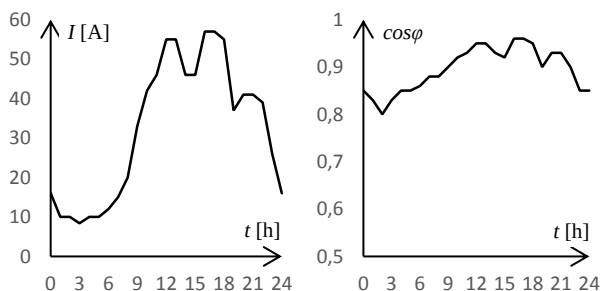
Pod potrošnjom se smatra ukupna potrošnja svih potrošača koji se napajaju sa niskonaponskih sabirnica transformatora. Njena vrednost definisana je sa dve veličine: modulom struje i faktorom snage, odnosno aktivnom i reaktivnom snagom. Te veličine, za razmatran period, mogu da se opišu sa [5]:

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je dr Goran Švenda, red.prof.

- minimalnim i maksimalnim vrednostima,
- dnevnim hronološkim dijagramima,
- normalizovanim dnevnih hronološkim dijagramima.

U ovom radu potrošnja je opisana modulom struje i faktorom snage, što je prikazano na slici 1.

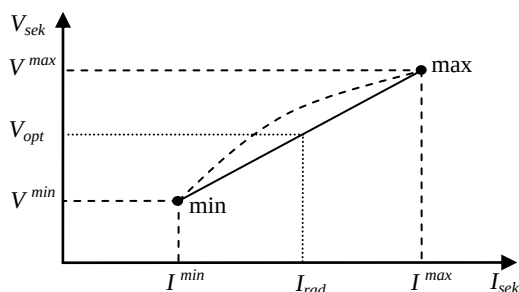


Slika 1 – Dnevni hronološki dijagrami: modula struje (levo), faktora snage (desno)

2.2 Klasična regulacija napona

Ideja klasične regulacije napona (KRN) se zasniva na održavanju napona na sabirnicama sekundara RTr, tako da naponi na potrošačima duž izvoda budu u dozvoljenim granicama [5,6].

Za režim sa maksimalnom (minimalnom) potrošnjom, kada pad napona na izvodu ima maksimalnu (minimalnu) vrednost, potrebno je podići (spustiti) napon na sekundarnoj strani RTr, tako da svi potrošači zajedno imaju najbolje moguće vrednosti. KRN se zasniva na zakonu regulacije na ARN. Primer zakona regulacije prikazan je na slici 2.



Slika 2 – Zakon regulacije ARN [5]

ARN, u skladu sa odnosnom izmerenih vrednosti struje i napona na sekundaru RTr i zakona regulacije, formira signal za promenu pozicije regulacionog otcepa na RTr.

Procedura automatske regulacije napona je sledeća [7]:

- prenos vrednosti struje i napona sa sekundara RTr u ARN;
- procena optimalne vrednosti napona V_{opt} na sekundaru RTr, na osnovu izmerene vrednosti struje I_{rad} , u skladu sa aktuelnim zakonom regulacije, slika 2;
- proračun odstupanja izmerene i željene vrednosti napona na sekundaru RTr;
- odluka: ako je odstupanje veće od unapred definisane vrednosti – inicirati delovanje ARN, koji inicira promenu pozicije regulacionog otcepa na RTr, s ciljem da se odstupanje napona smanji; ako je odstupanje manje od dozvoljene vrednosti – nastaviti sa nadzorom veličina struja i napona.

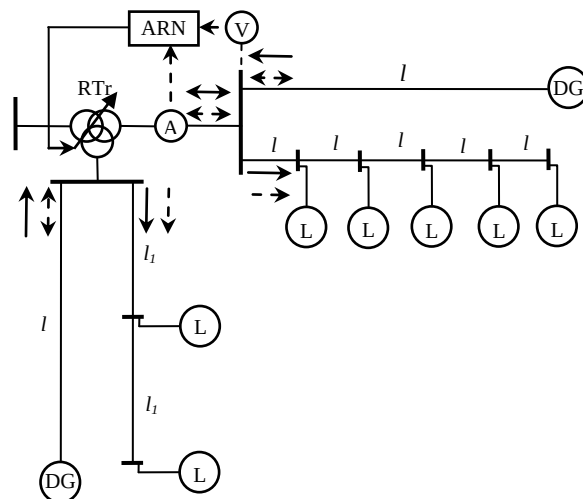
3. MATEMATIČKI MODEL

Priključenjem DG u DM, u zavisnosti od njegove proizvodnje manje ili više menja se ne samo vrednost pada napona na izvodu već i vrednost struje na sekundaru RTr. Osnovni problemi rada KRN, u takvim režimima, su:

- ARN nema korektnu sliku o opterećenju DM [1,2] i
- raste vrednost napona na mestu priključenja DG [8,9].

3.1 ARN nema korektnu sliku o opterećenju DM

Test DM sa 5 potrošača (L) na sekundaru i 2 na tercijeru, prikazana je na slici 3.



Slika 3 – Test DM

Na slici su prikazani mogući tokovi aktivnih snaga (pune strelice) i reaktivnih snaga (isprekidane strelice). Rad ARN i zakona regulacije definisan je samo za vrednosti napona i struje sa strane sekundara RTr. Kada je DG isključen, DM je pasivna, tokovi snaga su jednosmerni, ARN dobija korektne podatke o ukupnoj potrošnji DM, odnosno KRN radi kako je predviđeno. Međutim, kad se DG uključi, u zavisnosti od vrednosti njegovog injektiranja, izmerena struje može da se razlikuje više (DG više proizvodi) ili manje (DG manje proizvodi) od stvarne struje potrošnje DM. Pritom, uticaj DG je takav da ne menja samo vrednost struje na sekundaru RTr, već može da promeni i smer aktivne snage.

Posebno treba razmatrati DM sa tronamotajnim RTr koji na tercijeru imaju priključen DG (ali ne i na sekundaru). Za ovakav primer ARN nema informacije o promeni rada DG – kada se DG na tercijeru uključi, ili isključi, vrednost struje na sekundaru RTr je ista sa vrednošću struje potrošnje potrošača mreže koja se napaja preko sekundara RTr. Gledano sa strane potrošača na sekundaru RTr, ARN dobija korektne podatke i KRN dobro radi.

3.2 Raste vrednost napona na mestu priključenja DG

Drugi problem rada KRN u DM sa DG jeste porast napona na mestu ugradnje DG [8,9]. Kod pasivnih DM, DM bez DG, tokovi snaga su od RTr do potrošača. Vrednost napona je najveća u transformatorskoj stanici i opada do krajnjeg potrošača. U zavisnosti od tipa DG i njegovog režima, tokovi snaga mogu značajno da variraju. Samim tim značajno variraju i vrednosti padova napona na njegovom izvodu. Kada DG generiše više

snage nego što potrošnja zahteva, tok snage i pad napona na izvodu su suprotnog smera. Porast napona na izvodu može da se smanji povećanjem poprečnog preseka provodnika, apsorpcijom reaktivne snage od strane DG i pažljivim odabirom instalisane snage DG.

4. VERIFIKACIJA MATEMATIČKOG MODELA

Analiza uticaja DG na rad KRN u jednostavnoj test DM izvršena je na osnovu programa koji je razvijen u programskom jeziku FORTRAN.

4.1 Opis test DM

Test DM data je na slici 3, parametri mreže dati su u tabeli 1, a parametri RTr u tabeli 2.

Tabela 1 – Parametri test DM

l [km]	l_t [km]	l_G [km]	r [Ω/km]	x [Ω/km]	g [S/km]	b [S/km]
2	5	10	0.410	0.360	0	0.000100

Tabela 2 – Podaci o tronomotajnom RTr

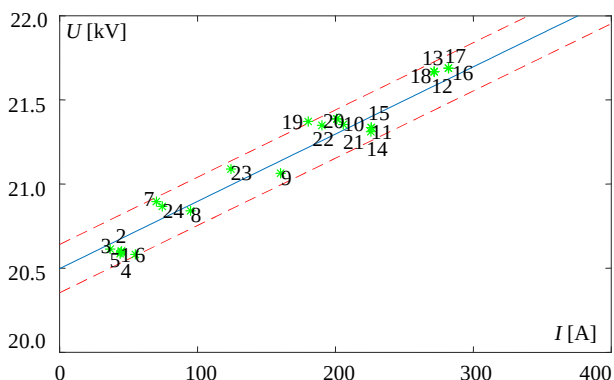
Prenosni odnos	(110 ± 10 × 1.64/21/10.5) kV/kV/kV
Naponi kratkog spoja	(14.8/8.3/2.7) %
Sn1	31.5 MVA
Sn2	31.5 MVA
Sn3	10.5 MVA

U nastavku su razmatrani sledeći primeri rada test DM:

- bez DG;
- DG na sekundaru generiše 10 MW;
- DG na tercijeru generiše 10 MW.

4.2 Test DM bez DG

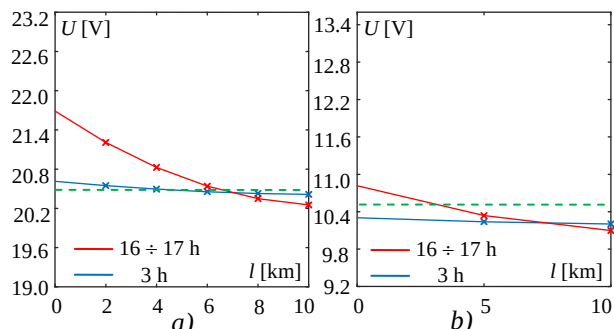
Razmatra se primer u kojem su DG u test DM isključeni. KRN, ARN i njen zakon regulacije su podešeni za ovaj slučaj. Zakon regulacije je prikazan na slici 4. Ostvarene vrednosti napona na sekundaru RTr, za 24 časa, naznačene su zelenim zvezdicama. Minimalna potrošnja je u 3h, dok je maksimalna u 16 ÷ 17h. Sa slike se može uočiti da RTr radi ispravno, u skladu sa zakonom regulacije.



Slika 4 – Zakon regulacije ARN bez DG

Na slici 5, prikazani su profili napona na izvodu sa potrošačima, koji su ostvareni za maksimalno (crvena linija) i minimalno opterećenje (plava linija). U toku minimalnog opterećenja, u trenutku 3h, profil napona ima blagi nagib. U toku maksimalnog opterećenja, period

16÷17h, ostvaren je maksimalni pad napona na izvodu sa potrošačima, crvena linija. U tom periodu potrebno je podići napon na sabirnicama sekunda RTr. Zelenim isprekidanim linijama prikazani su optimalni naponi za sekundar i tercijer.

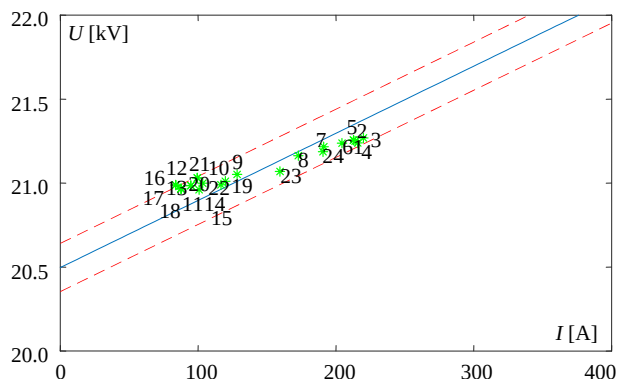


Slika 5 – Vrednosti napona za minimalnu i maksimalnu potrošnju: a) sekundar, b) tercijer

Iako ARN ne obuhvata potrošače na tercijeru, napon na sabirnicama tercijera se menja u skladu sa njegovim radom (sa promenom pozicije regulacione sklopke).

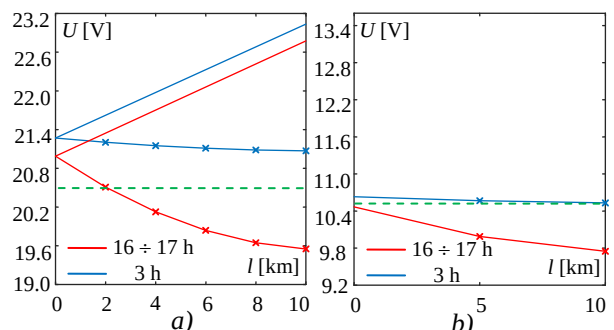
4.3 DG na sekundaru generiše 10 MW

Rezultati primene regulacije napona DM za primer u kojem DG na sekundaru RTr proizvodi 10 MW prikazani su na slici 6.



Slika 6 – Zakon regulacije ARN za test DM sa DG na sekundaru 10MW

Za razmatran primer, na slici 7, prikazani su profili napona ostvareni za maksimalno i minimalno opterećenje.



Slika 7 – Vrednosti napona za minimalnu i maksimalnu potrošnju: a) sekundar, b) tercijer

Za ovaj primer rad KRN je praktično neprihvatljiv:

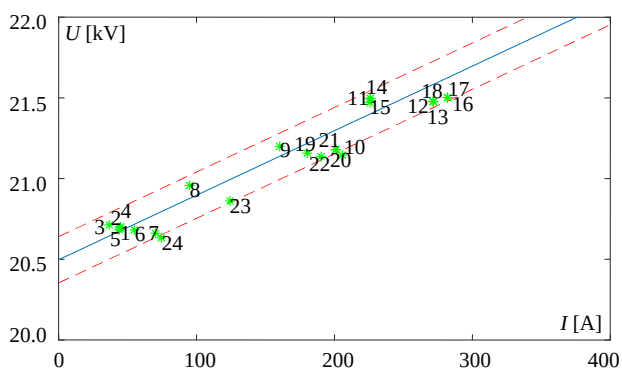
- ARN i zakon regulacije "vide" režim DM sa maksimalnom potrošnjom kao režim sa minimalnom potrošnjom, i obrnuto, "vide" režim DM sa

minimalnom potrošnjom kao režim sa maksimalnom potrošnjom;

- u periodu dana kada je potrebno povećati napon, ARN ga snižava i obrnuto;
- najveća šteta na potrošačima je u periodu niskog opterećenja (kada je potrebno spustiti napon, ARN ga povećava);
- naponske prilike su loše kako na sekundaru, tako i na tercijeru;
- zbog injektiranja velike snage, 10 MW, javlja se veliki porast napona na izvodu sa DG.

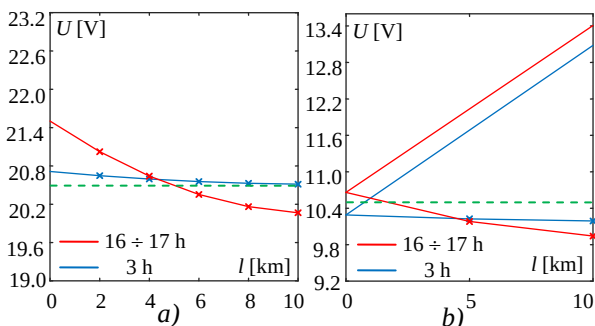
4.4 DG na tercijeru generiše 10 MW

Rezultati primene regulacije napona DM za primer u kojem DG na tercijeru RTr proizvodi 10 MW prikazani su na slici 8.



Slika 8 – Zakon regulacije ARN sa DG na tercijeru 10MW

Za razmatran primer, na slici 9, prikazani su profili napona ostvareni za maksimalno i minimalno opterećenje.



Slika 9 – Vrednosti napona za minimalnu i maksimalnu potrošnju: a) sekundar, b) tercijer

Uočava se da se postavljanjem DG na tercijer RTr, ne remeti značajno regulacija napona – deo 4.2, zbog jednosmernog toka snage na sekundaru RTr i korektnih ulaznih podataka za ARN. Šteta ostvarena na svim potrošačima test DM, u zavisnosti od rada DG, prikazana je u tabeli 3. Najveća šteta je za primer rada DG na sekundaru RTr.

Tabela 3 – Šteta na svim potrošačima

DG sec	0 MW	1 MW	10 MW	0 MW	0 MW
DG ter	0 MW	0 MW	0 MW	1 MW	10 MW
Šteta [n.j]	4.554	5.889	15.369	4.568	4.650

5. ZAKLJUČAK

U ovom radu analizirani su problemi klasične regulacije napona u DM nakon priključenja DG. Kada se DG priključi, vrednost struje koja je izmerena na sekundaru RTr razlikuju se od stvarne potrošnje u DM. Posledica je da ARN nema korektnu sliku o potrošnji DM. U skladu sa tim klasična regulacija napona ne može da izvrši svoju osnovnu ulogu, da optimalno podesi napon na sabirnicama sekundara RTr.

Priključenjem DG u DM, tradicionalno pasivne DM postaju aktivne. Klasični elektroenergetski sistemi, po raznim osnovama, nisu pripremljeni za režime sa promenjenim smerom tokova snaga, iz DM prema prenosnim mrežama. Posledica je da ARN mogu da koriguju napon u suprotnom smeru od optimalnog.

Klasična regulacija napona nema informacije o radu DG na tercijeru RTr, tako da će nezavisno od njegovog rada nastaviti da kontroliše napon na strani sekundara i regulacija napona će raditi kako je predviđeno. Porast napona na izvodu sa DG je siguran zbog suprotnog toka snage.

6. LITERATURA

- [1] F.A.Viawan: Voltage Control and Voltage Stability of Power Distribution Systems in the Presence of Distributed Generation, PhD Thesis, Chalmers University of technology Göteborg, Sweden, 2008.
- [2] G.Chao: Voltage Control in Distribution Networks using On-Load Tap Changer Transformers, PhD, University of Bath, UK, 2013.
- [3] L.Ingmar: Integration of Wind Power in Medium Voltage Networks, PhD, Lund University, Sweden, 2011.
- [4] D.Popović, D.Bekut, V.Treskanica: Specijalizovani DMS algoritmi, Novi Sad, 2004.
- [5] G.Švenda, V.Strezoski, R.Bibić: Regulacija napona u eksploataciji distributivnih mreža, Elektroprivreda, Br. 3, str. 28-42, 2008.
- [6] G.Švenda, R.Bibić: Unapređen matematički model za regulaciju napona u distributivnim mrežama, Elektroprivreda, Br.2, str. 120-126, 2005.
- [7] D.Đanić: Primena URN u ED Sombor, master rad, Fakultet tehničkih nauka – Novi Sad, Novi Sad, Srbija, 2011.
- [8] M.A.Mahmud, M.J.Hossain, H.R.Pota: Analysis of Voltage Rise Effect on Distribution Network with Distributed Generation, 18th World Congress The International Federation of Automatic Control Milano, Italy, 2011.
- [9] T.Xu, P.C.Taylor: Voltage Control Techniques for Electrical Distribution Networks Including Distributed Generation, 17th World Congress The International Federation of Automatic Control, Seoul, Korea, 2008.

Kratka biografija:



Uglješa Zelić rođen je u Kninu 1993. godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Elektroenergetski sistemi odbranio je 2016. god. Iste godine upisao se na master studije na istom smeru.

DSL I GENERATOR KODA ZA AUTOMATIZACIJU RAZVOJA PHP APLIKACIJA

DSL AND CODE GENERATOR FOR AUTOMATING THE DEVELOPMENT OF PHP APPLICATIONS

David Milivojev, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – U radu je predstavljeno projektovanje i implementacija jezika specifičnog za domen (*Domain Specific Language – DSL*) i generatora koda za automatizaciju razvoja PHP aplikacija.

Abstract – This document presents the design and implementation of *Domain Specific Language (DSL)* and code generator for the automation of the development of PHP applications.

Ključne reči: *DSL, PHP, generatori koda*

1. UVOD

Zadatak ovog rada je projektovanje i implementacija jezika specifičnog za domen (*Domain Specific Language – DSL*) i generatora koda za automatizaciju razvoja PHP aplikacija. Generatori koda smanjuju redundantnost koda i vreme izrade i povećavaju nivo apstrakcije, što dovodi do efikasnijeg razvoja i boljeg kvaliteta programskog koda ciljane aplikacije.

Prilikom implementacije generatora, neophodno je ostaviti mogućnost dorade generisane aplikacije dodavanjem ručno pisanog koda, pošto generisanje kompletne aplikacije obično nije moguće.

Rad je organizovan na sledeći način.

Drugo poglavlje sadrži opis jezika.

Treće poglavlje daje primer korišćenja datog jezika i generatora koda za implementaciju web aplikacije za podršku rada konferencija.

Četvrto poglavlje je zaključak rada.

2. JEZIK

U ovom poglavlju je opisana gramatika jezika i njegov meta-model.

2.1. Gramatika

Definisanjem ključne reči “WebApp” (listing 2.1) kreira se model aplikacije iz kog se navodi ime modela sa zagradama “[“ i “]” (koji čini i ime foldera) iz koga se kreira telo modela. Opis modela se definiše unutar navodnika, a upotreba “camel 47case” načina pisanja nije dozvoljena.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila dr Gordana Milosavljević.

```
EntityModel:
  'WebApp' name=ID '['
    (shortDesc=STRING)?
      classes+=Class
  ']'
;
```

Listing 2.1. Pravilo za kreiranje modela

Kreiranje se nastavlja definisanjem ključne reči „class“ sa zagradama “{“ i “}” unutar tela modela, čime se kreira telo klase u kome se može nalaziti jedan ili više entiteta (listing 2.2). Nakon ključne reči “class” opciono se može navesti ime baze podataka tako što se unese ključna reč “+db” i navede njeno ime.

```
Class:
  'class' (dbase='+db' name=ID)? '{'
    entities+=Entity
  '}'
;
```

Listing 2.2. Pravilo gramatike jezika koje se odnosi na kreiranje klase i baze podataka

Definisanjem ključne reči “entity” kreira se entitet iz kog se navodi ime entiteta, a definisanjem zagrada “{“ i “}” kreira se telo (listing 2.3).

U modelu samo jedan entitet iz imena mora sadržati ključnu reč “!m”, čime se označava primarni entitet, tj. glavna - početna stranica. Osim početne stranice, mogu postojati i „obične“ stranice pridružene entitetu, kao i *master-details* stranice. Obične stranice nemaju dodatak posle imena, a *master-details* se označavaju ključnom reči “!d”.

Svaki entitet mora imati jedan ili više obeležja i tačno jedan servis.

```
Entity:
  'entity' name=ID (main='!m' | detail='!d')?
  '{'
    properties+=Property
    services+=Service
  '}'
;
```

Listing 2.3. Pravilo za kreiranje entiteta.

Obeležje se kreira tako što se navede njegovo ime sa nizom opcija (listing 2.4). Obeležje koje je primarni ključ se označava ključnom reči “\$key” ispred imena. Samo jedno obeležje može biti ključ entiteta.

```
Property:
  (key='$key')? (title='^' | sort='>')? (name=ID
  (':' types=Type)? (init='=' (numb=INT)? )?) |
  (addlink='+' linkname=[Entity] '<'
  linkid=[Property] '>' showby=[Property])?
;
```

Listing 2.4. Pravilo za specifikaciju obeležja

Tip “input” označava da se polje koristi za unos teksta manjeg obima (string). Tip “desc” upotrebljava se za unos

polja kod kojih se očekuju opširni opisi (veća količina teksta). Upotreba “date” koristi se za unos datuma i vremena, dok upotreba “time” služi samo za unos vremena.

```

Type:
  input='input' | textarea='desc' | date='date'
  | time='time'
;

```

Listing 2.5. Pravila za tipove obeležja

Servis se definiše pomoću ključne reči “#actions” iza koje se navode uglaste zagrade “[“ i “”]”, čime se kreira telo servisa (listing 2.6). U telu servisa nalaze se akcije. Servis mora sadržati svih pet akcija koje su opisane u nastavku poglavlja.

```

Service:
  action='#actions' '['
    actions+=Action
  ']'
;

```

Listing 2.6. Pravilo za definisanje servisa

Pojedinačne akcije se definišu pomoću ključne reči “@” i imena akcije, nakon kojeg se navodi “.” i referencira ime metode (listing 2.7). Redosled akcija nije bitan.

```

Action:
  '@' name=ID ':' meths=Method
;

```

Listing 2.7. Pravilo za specifikaciju pojedinačnih akcija

Predefinisane akcije su: “view, views, create, edit, delete” i “!” iza kog se navodi ime (listing 2.8). Metoda “view” predstavlja prikaz jednog sloga, “views” predstavlja prikaz svih slogova, “create” predstavlja kreiranje sloga, “edit” izmenu sloga, “delete” brisanje sloga, “!” sa imenom akcije – predstavlja ručno pisanu akciju koja se uključuje u generisani kod.

```

Method:
  ga='views' | de='delete' | ad='create' |
  ed='edit' | go='view' | (custom='!'
  name=ID) ?
;

```

Listing 2.8. Prikaz pravila gramatike jezika koji se odnosi na metode.

U datom primeru, kreiranje entiteta se odnosi na {{entity.name}} u okviru šablona na listingu 2.9, prikaz comobox-a i kreiranje objekta se odnosi na {{props.name}}, a metode na {{prop.name}}.

```

<tr>
  <td>{{entity.name|capitalize}}:</td>
  <td>
    <select name="{{entity.name|capitalize}}">
      <?php
        $curl = curl_init('http://localhost/{{entity._model_name}}/services/{% for ser in entity.services %}{% if prop.meths.ga %}{%prop.name|lower}}{% endif %}{% endfor %}{% endfor %}');
        curl_setopt($curl,
CURLOPT_RETURNTRANSFER, true);
        $response = curl_exec($curl);
        $data = json_decode($response);
        {% for ser in entity.services %}{% if prop.meths.ga %}{%prop.name|lower}}{% endif %}{% endfor %}{% endfor %} = array();

        for ($i=0; $i<=count($data)-1;$i++)

```

```

        ${{entity.name|lower}} = new
        {{entity.name|capitalize}}();
        ${{entity.name|lower}}->
        >jsonDeserialize($data[$i]);
        array_push($% for ser in
        entity.services %){% for prop in ser.actions
        %}{% if prop.meths.ga %}{%prop.name|lower}}{%
        endif %}{% endfor %}{% endfor %},
        ${{entity.name|lower}});
      }

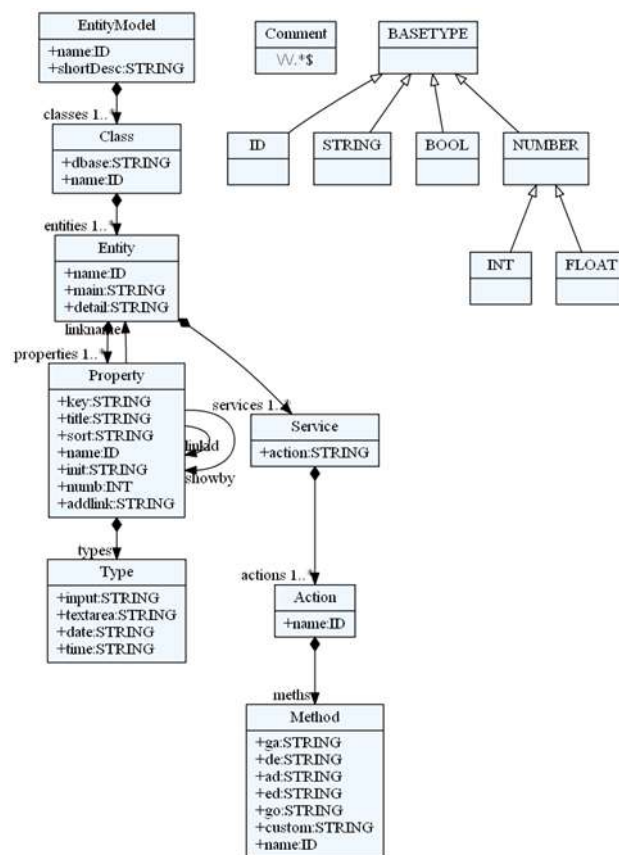
      foreach($% for ser in entity.services
      %){% for prop in ser.actions %}{% if
      prop.meths.ga %}{%prop.name|lower}}{% endif %}{%
      endfor %}{% endfor %} as $k):
        ?>
        <option value="<?php echo $k->get_{%
        for props in entity.properties %}{% if props.key
        %}{%props.name}}{% endif %}{% endfor %}(); ?>">
        <?php echo $k->get_{% for props in
        entity.properties %}{% if props.title
        %}{%props.name}}{% endif %}{% endfor %}(); ?>
        </option>

        <?php
        endforeach;
        ?>
      </select>
    </td>
  </tr>

```

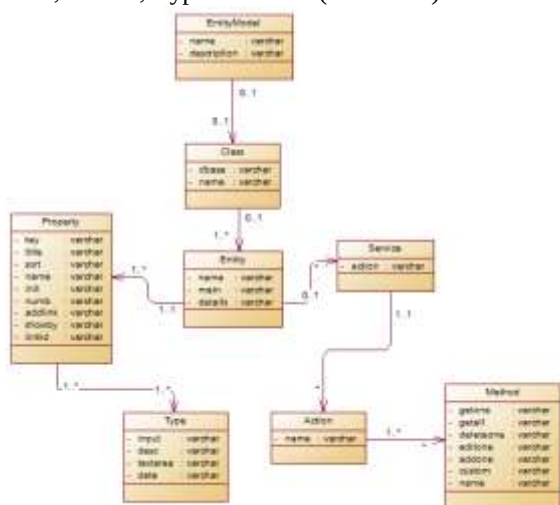
Listing 2.9. php_actionViews.template šablon

Meta-model na apstraktan način opisuje moguću strukutru modela definišući pri tome koncepte domena i njihove veze kao i ograničenja i pravila modelovanja [3]. Dati meta-model je projektovan tako da pokriva veliki broj situacija do kojih može doći u programskom jeziku PHP [4] (slika 2.10). Meta-model je dobijen na osnovu gramatike jezika.



Slika 2.10. Meta-model jezika za opis PHP aplikacija

Python klase koje čuvaju instance elemenata modela i koje su kreirane na bazi meta-modela jezika za opis PHP aplikacija su: EntityModel, Class, Entity, Property, Service, Action, Type i Method (slika 2.11).



Slika 2.11. Dijagram klasa

3. Primer izgenerisane aplikacije

Na listingu 3.1 je dat model koji specificira aplikaciju za konferencije. Na osnovu prethodno objašnjenih pravila, u modelu se definišu tri entiteta koja ujedno predstavljaju i tabele u bazi podataka: „Konferencija“ je entitet početne stranice, „Rang“ entitet obične stranice. „Predavanje“ je *master-details* stranica Ime baze podataka je „Konferencija2018“.

```

WebApp Konferencije[
  "Ovo je sajt na temu konferencija"

  class +db konferencija2018{

    entity Konferencija !m {
      $key idKonferencija =
      ^naziv : input = 55
      opis : desc = 20
      +Rang <idRang> nazivrang
      #actions[
        @konferencije : views
        @konferencija : view
        @dodajkonferenciju : create
        @izbrisikonferenciju : delete
        @izmenikonferenciju : edit
        @konferencijepretraga
          !PrikaziKonferencijePretraga
      ]
    }

    entity Rang {
      $key idRang =
      nazivrang : input =
      #actions[
        @rangovi : views
        @rang : view
        @izbrisirang : delete
        @dodajRang : create
        @izmenirang : edit
        @rangpretraga : !RangPretraga
      ]
    }

    entity Predavanje !d {
      $key idPredavanje =
      ^naziv : input =
      predavaci : desc =
      >pocetak : date =
  
```

```

    kraj : date =
    +Konferencija <idKonferencija> naziv
    #actions[
      @predavanja : views
      @predavanje : view
      @izbrisipredavanje : delete
      @dodajpredavanje : create
      @izmenipredavanje : edit
      @predavanjepretraga : !PredavanjePretraga
    ]
  }
}
]
  
```

Listing 3.1 Model za aplikaciju „Konferencija“

Na slici 3.1 prikazana je izgenerisana glavna stranica - forma „Konferencija“ koja sadrži spisak entiteta prikazanih u meniju i pretragu.



Slika 3.1. Prikaz početne stranice „Konferencija“

Na slici 3.2 prikazana je forma za unos podataka o novoj konferenciji, koja sadrži polja sa combobox-om koji prikazuje nazive rangova (iz entiteta „Rang“) od kojih je moguće samo jedan izabrati. Pritiskom dugmeta „snimi“ se kreira data konferencija.



Slika 3.2. Stranica za kreiranje konferencije

Na slici 3.3 prikazana je izgenerisana *master-details* stranica „Predavanje“ koja ima i opciju za filtriranje podataka. Filtriranje se vrši na osnovu naziva ponuđenih konferencija. Prilikom izbora konferencije i pritiskom dugmeta „filtriraj“, biće prikazana konferencija sa listom predavanja za datu konferenciju.



Slika 3.3. Primer master-details stranice „Predavanje“

3. ZAKLJUČAK

U okviru ovog rada je projektovan tekstualni jezik specifičan za domen za specifikaciju PHP aplikacija i implementiran je generator koda. Jezik je kreiran korišćenjem TextX okruženja, a generator je implementiran na bazi Jinja2 obrađivača šablona. Korišćenjem jezika i generatora je automatski izgenerisana PHP aplikacija za stručne i akademske konferencije. Upotreba generatora je pokazala da ovo rešenje pomaže programerima tako što poboljšava kvalitet softvera, smanjuje vreme izrade i mogućnost pojave grešaka.

Pravci daljeg razvoja će ići ka unapređenju jezika i generatora.

Jezik se može proširiti tako da podržava kompleksnije PHP aplikacije, ne samo one bazirane na CRUD formama i bazičnim operacijama. Mogu se podržati različite vrste formi, predefinisane vrste filtriranja, pretrage, dodatne transakcija i sl.

Generator se može unaprediti tako da mu ulaz bude i šema baze podataka, a ne samo model na datom jeziku. Takođe, može se dodati mogućnost biranja predefinisanih tema, kombinacije boja i stila, kao podrška za više izlaznih uređaja (*responsive design*). Takođe, može se obezbediti da korisnik odabere željeni poznati PHP framework ili da izostavi tu opciju.

4. LITERATURA

- [1] Igor Dejanovic, 2008. Metamodel, Editor modela i generator poslovnih aplikacija
- [2] Lorenzo Bettini, 2016. Implementing Domain-Specific Language with Xtext and Xtend, Packt Publishing
- [3] Markus Voelter, 2013. DSL Engineering: Designing, Implementing and Using Domain-Specific Languages
- [4] Dragoljub Ilić, 2017. Generator koda za sisteme za dinamičko upravljanje sadržajem, master rad, FTN, Novi Sad
- [5] Nenad Todorović, 2017. Arhitektura generatora koda za softverske proizvodne linije, master rad, FTN, Novi Sad
- [6] Aleksandar Lukić, 2017. Primena „Roslyn“ kompajlerske platforme i proširenog XML parsera za generisanje koda u softverskim proizvodnim linijama, master rad, FTN, Novi Sad
- [7] Igor Dejanović, 2016. Jezičke sintakse i meta-modelovaje, <http://www.igordejanovic.net/courses/jsd/jezicke-sintakse-i-metamodelovanje.html#1>
- [8] Igor Dejanović, 2016. Jezici specifični za domen, <http://www.igordejanovic.net/courses/jsd>
- [9] Jinja2, <http://jinja.pocoo.org/>
- [10] PHPMaker, <http://www.hkvstore.com/phpmaker/default.asp>
- [11] Form Generator, PHP & MySQL Generator (FPMG), <http://www.fpmgonline.com/index.php>

Kratka biografija:



David Milivojev rođen je u Zrenjaninu 1993. god. Osnovne studije završio na tehničkom fakultetu “Mihajlo Pupin”, smer Informacione tehnologije – inženjerstvo, završio je 2016. Godine. Trenutno student master studija na smeru softversko inženjerstvo i informacione tehnologije.

UTICAJ KONFIGURACIJE MERENJA NA TAČNOST ANALIZE LOŠIH PODATAKA U STATIČKOJ ESTIMACIJI STANJA**IMPACT OF MEASUREMENTS' CONFIGURATION ON ACCURACY OF BAD DATA ANALYSIS METHODS IN STATE ESTIMATION**

Milica Jović, Vladan Krsman, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je prikazano kako konfiguracija merenja utiče na stepen korelacije merenja, kao i na efikasnost detekcije i identifikacije loših podataka u statičkoj estimaciji stanja. Cilj ovog rada je da se utvrdi da li klasični algoritmi za analizu loših podataka prenosnih mreža mogu biti korišćeni u sub-prenosnim mrežama. Verifikacija rezultata izvršena je na IEEE mreži od 14 čvorova pri različitim konfiguracijama merenja i vrednostima redundanse merenja.

Ključne reči: Detekcija i identifikacija loših podataka, Korelacija podataka, Sub-prenosne mreže

Abstract – This paper presents how the measurement configuration influences the degree of measurements correlation, as well as the efficiency of bad data detection and identification in the state estimation. The goal of this paper is to determine whether traditional bad data analysis algorithms for the transmission networks can be used in sub-transmission networks. Verification of results was carried out on an IEEE network with 14 nodes at different measurements configuration and redundancy.

Keywords: Bad data detection and identification, Correlation of measurements, Sub-transmission

1. UVOD

Poznavanje aktuelnog režima je osnova za većinu proračuna u vezi sa analizom, eksploatacijom i upravljanjem elektroenergetskim sistemima. Od interesa je poznavanje stanja elektroenergetskog sistema u nekom diskretnom vremenskom trenutku, ali ne i način na koji je sistem dospao u taj režim, pa se estimacija stanja naziva statičkom.

Upravo iz tog razloga, statička estimacija predstavlja jednu od osnovnih funkcija za najveći broj proračuna u navedenim oblastima.

Statički estimator stanja transformiše merne podatke elektroenergetskog sistema u procenu vektora promenljivih stanja. Glavni cilj nije dobijanje vrlo „tačne“, već pouzdane i u datom trenutku „najbolje“ procene promenljivih stanja, čak i kada su merenja jako poremećena, zbog prisustva šumova i drugih grešaka [1]. Ukoliko postoji dovoljna redundansa merenja, realno je očekivati da se greške filtriraju.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je dr Vladan Krsman, docent.

U slučaju da se jave velike greške merenja, one se teško filtriraju i mogu stvoriti ozbiljne probleme u dobijanju kvalitetne estimacije stanja elektroenergetskih sistema (EES-a), što implicira postojanje alata za analizu loših merenja. U skladu sa gore navedenim, detekcija i identifikacija loših merenja predstavlja veoma važnu funkciju statičkog estimatora stanja.

2. SUB-PRENOSNE MREŽE

Sub-prenosne mreže predstavljaju deo elektroenergetskog sistema koji povezuje podsistem distribucije i podsistem prenosa. Podsistem prenosa čini prenosna mreža, koju pre svega, čine vodovi i transformatori. Distributivna mreža prvenstveno služi za napajanje konkretnih potrošača.

Između prenosnih i distributivnih mreža postoje jasne razlike, ali će za potrebe ovog rada biti izdvojene samo određene. Prenosne mreže su upetljane, tj. pored uzdužnih veza postoji i niz stalno uključenih poprečnih veza, dok su distributivne mreže radijalne. Značaj redundanse merenja za statičku estimaciju stanja je veliki, pa je bitno pomenuti da prenosne mreže imaju veliku redundansu merenja koja se kreće od 1.8 pa do 3, dok se kod distributivnih mreža nalazi uvek ispod 1, a najčešće je to vrednost oko 0.1-0.3.

Osnovni problem sub-prenosnih mreža, jeste to što su one po nekim karakteristikama sličnije prenosnim, dok su po drugim sličnije distributivnim mrežama i ne postoji jedinstvena definicija po kojoj bi se u jednom EES-u mogle detektovati. Za sub-prenosne mreže karakteristična je njihova upetljanost, ali ne u tolikoj meri kao u čisto prenosnim mrežama. Kada se posmatra redundansa merenja evidentno je da je kod sub-prenosnih mreža manja nego kod prenosnih i upravo se zbog toga dovodi u pitanje mogućnost korišćenja algoritama namenjenih proračunima prenosnih mreža.

3. STATIČKA ESTIMACIJA STANJA

Ulazni podaci za statičku estimaciju stanja su analogna merenja i signalizacije statusa aparata, kao i baza podataka, iz koje se očitavaju parametri elemenata mreže. Statička estimacija stanja je proces u kojem se određuju nepoznate vrednosti promenljivih stanja, na bazi redundantnih merenja iz sistema [1].

Za proračun statičke estimacije stanja koriste se tri skupa ulaznih podataka: merenja iz sistema, podaci o konfiguraciji (topologiji mreže) i podaci o parametrima modela mreže.

Vektor promenljivih stanja opisuje režim posmatranog EES-a, on predstavlja n -dimenzioni vektor promenljivih stanja ($\mathbf{x}$).

Elementi vektora promenljivih stanja su:

- V_i – moduli fazora napona u čvorovima ($i = 1, 2, \dots, N$)
- θ_i – uglovi fazora napona u čvorovima ($i = 1, 2, \dots, N$), osim referentnog balansnog,
- a_{ij} – nenominalni odnosi transformacije klasičnih regulacionih transformatora ($ij = 1, 2, \dots, L_t$),
- φ_{ij} – nenominalni odnosi transformacije faznih regulacionih transformatora ($ij = 1, 2, \dots, L_{ps}$).

$$\mathbf{x} = [\mathbf{V} \ \boldsymbol{\theta} \ \mathbf{a}_l \ \boldsymbol{\varphi}_l]^T. \quad (1)$$

Vektor merenja ($\mathbf{z}$) povezan je sa vektorom promenljivih stanja ($\mathbf{x}$) preko nelinearne jednačine [3]:

$$\mathbf{z} = \mathbf{h}(\mathbf{x}) + \mathbf{e}, \quad (2)$$

gde su:

- $\mathbf{z}$ – M-dimenzioni vektor merenja,
- $\mathbf{h}(\mathbf{x})$ – M-dimenzioni vektor funkcija promenljivih stanja,
- $\mathbf{x}$ – n-dimenzioni vektor promenljivih stanja,
- $\mathbf{e}$ – M-dimenzioni slučajni vektor grešaka merenja.

Merenja koja se koriste u proračunu statičke estimacije stanja su:

$$\mathbf{z} = [\mathbf{P}_l^T \ \mathbf{Q}_l^T \ \mathbf{I}_l^T \ \mathbf{P}^T \ \mathbf{Q}^T \ \mathbf{I}^T \ \mathbf{V}^T \ \mathbf{a}_l^T \ \boldsymbol{\varphi}_l^T]^T, \quad (3)$$

gde su:

- $\mathbf{P}_l$ – vektor tokova aktivnih snaga po granama,
- $\mathbf{Q}_l$ – vektor tokova reaktivnih snaga po granama,
- $\mathbf{I}_l$ – vektor tokova struja po granama,
- $\mathbf{P}$ – vektor injektiranih aktivnih snaga u čvorovima,
- $\mathbf{Q}$ – vektor injektiranih reaktivnih snaga u čvorovima,
- $\mathbf{I}$ – vektor injektiranih struja u čvorovima,
- $\mathbf{V}$ – vektor modula napona u čvorovima,
- $\mathbf{a}_l$ – vektor nenominalnih odnosa transformacije klasičnih regulacionih transformatora,
- $\boldsymbol{\varphi}_l$ – vektor uglova faznih regulacionih transformatora.

Sva merenja se mogu podeliti na telemetrisana merenja, pseudo-merenja i virtuelna merenja. Telemetrisana merenja su merenja koja se dobijaju od SCADA sistema. Pseudo-merenja su pretpostavljene vrednosti nekih veličina. Služe za povećanje redundanse merenja (i da bi se neobservabilni delovi mreže učinili observabilnim [1]). Pored stvarnih i pseudo-merenja, postoje i virtuelna merenja i njihova vrednost je poznata u svakom trenutku sa velikom sigurnošću.

Varijansa merenja (σ_m^2) daje indikaciju kvaliteta pojedinačnog merenja. Velike varijanse (mali težinski faktori) pokazuju da je to merenje nepouzdanije (manje tačno), i obrnuto. Vrednost težinskog faktora, u zavisnosti od varijanse merenja, može se kretati od nekoliko hiljada, pa do milion [1]. Najveći težinski faktori se dodeljuju virtualnim merenjima, dok se najmanji težinski faktori dodeljuju pseudo-mereljima.

Varijansa (težinski faktori) merenja definišu se preko matrice kovarijansi merenja ($\mathbf{R}$) [1]:

$$\mathbf{R} = \text{cov}\{\mathbf{e}\} = \mathbf{E}\{\mathbf{e}^T \mathbf{e}\} = \text{diag}\{\sigma_m^2\}, \quad (4)$$

Težinski faktor predstavlja recipročnu vrednost varijanse merenja.

Redundansa merenja definiše se kao odnos broja raspoloživih (M) i minimalno potrebnog broja nezavisnih merenja koji praktično odgovara broju promenljivih stanja sistema (n), pri čemu je $M > n$:

$$R_d = \frac{M}{n}. \quad (5)$$

Metod minimuma sume otežanih kvadrata odstupanja - WLS (Weighted Least Squares) metod, je najčešće korišćeni algoritam u statičkoj estimaciji stanja.

Moguće je definisati problem statičke estimacije stanja kao: pronaći vrednosti promenljivih stanja, pri kojima se ima minimalna suma kvadrata odstupanja između merenih i stvarnih vrednosti veličina (koje su izražene kao funkcija vektora promenljivih stanja), pri čemu se svakoj razlici mora dodeliti odgovarajuća verodostojnost (težina). Iz teorije estimacije je poznato da se ovo čini podelom razlike merenja sa odgovarajućom varijansom, tako da merenja sa malom greškom imaju veliku težinu u navedenoj sumi i obrnuto.

Ovako definisani problem u matematičkoj formi može se izraziti kao:

$$\min_{\mathbf{x}} \{J(\mathbf{x}) = [\mathbf{z} - \mathbf{h}(\mathbf{x})]^T \mathbf{R}^{-1} [\mathbf{z} - \mathbf{h}(\mathbf{x})]\}, \quad (6)$$

ukupna suma kvadrata reziduala merenja koja se minimiziraju. Rešavanjem problema (6) raspolaže se estimiranim vektorom promenljivih stanja, koji predstavlja optimalnu procenu promenljivih stanja [5].

4. KORELACIJA, DETEKCIJA I IDENTIFIKACIJA LOŠIH MERENJA

Korelacija merenja predstavlja međusobnu zavisnost dva merenja, bez obzira na vrstu i tip merenja [5]. Ukoliko su dva merenja korelisana, tada su i njihove greške korelisane, što je od velikog značaja za proces detekcije i identifikacije loših merenja.

Korelacija merenja se ispituje preko matrice korelacije reziduala merenja date u sledećem obliku:

$$\mathbf{R}_r^n = \text{diag}\{\boldsymbol{\Omega}\}^{-1/2} \cdot \boldsymbol{\Omega} \cdot \text{diag}\{\boldsymbol{\Omega}\}^{-1/2}, \quad (7)$$

gde su:

- $\mathbf{R}_r^n$ – matrica korelacije reziduala merenja i
- $\boldsymbol{\Omega}$ – matrica kovarijansi reziduala merenja.

Po prirodi problema matrica korelacije reziduala je simetrična matrica, a njeni dijagonalni elementi su jednaki 1 jer je svaka promenljiva u punoj korelaciji sama sa sobom. Vandijagonalni elementi imaju vrednosti koje se kreću od -1 do 1, u zavisnosti od smera i stepena korelacije.

Detekcija loših merenja predstavlja jednu od najznačajnijih funkcija statičke estimacije stanja. Njena suština je da detektuje postojanje podatka sa greškom koji može dovesti do smanjenja tačnosti estimiranog stanja sistema. Ona se sprovodi nakon izvršene statičke estimacije stanja i vrši se pomoću sledeća dva testa [5]:

1. Hi-kvadrat test, odnosno poređenje estimiranog optimizacionog kriterijuma (indeks performansi) sa odgovarajućim pragom.
2. Test najvećeg normalizovanog reziduala merenja, odnosno poređenje normalizovanih reziduala merenja sa odgovarajućim pragom.

Ukoliko je bilo koji od ova dva testa pozitivan, smatra se da su greške prisutne.

Hi-kvadrat test polazi od kriterijuma minimizacije sume otežanih kvadrata odstupanja, datog izrazom:

$$J(x_{est}) = \sum_{m=1}^M \left(\frac{z_m - h_m(x_{est})}{\sigma_m} \right)^2, \quad (8)$$

$J(x_{est})$ – vrednost optimizacionog kriterijuma u tački optimuma (x_{est}),

$h_m(x_{est})$ – vrednost funkcije m -tog merenja u tački optimuma (x_{est}),

$r_m = z_m - h_m(x_{est})$ – estimirani rezidual m -tog merenja u tački optimuma (x_{est}) i

σ_m – varijansa m -tog merenja.

Vrednost praga se obeležava kao vrednost χ^2 funkcije raspodele, za odgovarajuću verovatnoću pouzdanosti i broj stepeni slobode. Ako je $J(x_{est}) \geq \chi^2$ test je pozitivan, odnosno detektovano je postojanje lošeg podatka, a ukoliko je $J(x_{est}) < \chi^2$ test je negativan, odnosno nije detektovano postojanje lošeg podatka.

Identifikacija loših merenja je proces u kom se pronalazi merenje koje je loše i narušava tačnost estimacije stanja. Ukoliko je moguće, to merenje se izbacuje iz skupa merenja, u suprotnom se zamenjuje odgovarajućim merenjem koje može biti istorijska vrednost ili korigovana vrednost identifikovanog lošeg merenja. Identifikacija loših merenja se kao i detekcija, sprovodi nakon izvršene statičke estimacije stanja.

Najčešće korišćen pristup identifikaciji se bazira na analizi reziduala merenja [5]. U okvirima analize reziduala, široko korišćeni klasični algoritmi su [5]:

1. sukcesivna eliminacija ili kompenzacija merenja sa najvećim rezidualom i
2. identifikacija testiranjem hipoteze (*Hypothesis Testing Identification*).

Vrednost i -tog člana vektor normalizovanih reziduala izračunava se kao:

$$r_i^N = \frac{|r_i|}{\sqrt{\Omega_{ii}}}, \quad i = 1, \dots, m, \quad (9)$$

gde su:

r_i – vrednost reziduala merenja i -tog merenja i

Ω_{ii} – vrednost i -tog dijagonalnog elementa u matrici kovarijansi reziduala merenja (Ω).

U odsustvu loših merenja, normalizovani reziduali imaju normalnu normiranu raspodelu. U praksi postoji nekoliko različitih pristupa za odabir praga poređenja najvećeg normalizovanog reziduala [5]. Najčešće korišćen pristup je da se za gornju granicu normalne normirane funkcije raspodele uzima konstantna vrednost $c = 3$. U postupku identifikacije loših merenja, nakon dobijenog vektora normalizovanih reziduala, potrebno je naći najveći rezidual od svih. Pretpostavka je da k -to merenje ima najveću vrednost reziduala. Ukoliko se dobije da je $r_k^N > c$, onda se smatra da je k -to merenje loše merenje, u suprotnom nema merenja sa greškom.

Nakon identifikovanog lošeg merenja, pristupa se njegovoj eliminaciji i postupak se ponavlja ispočetka.

5. REZULTATI

U ovom radu izvršena je verifikacija rešenja na IEEE test sistemu koji se sastoji od 14 čvorova koji su povezani preko 15 vodova i 3 transformatora [6].

Sva merenja za ovu mrežu koja su korišćena u radu su data u tabeli 1 i u tabeli 2:

Tabela 1 – Podaci o merenjima injektiranih snaga i modula napona

Čvor	$P_{inj}[r.j.]$	$Q_{inj}[r.j.]$	$V[r.j.]$
1	1.0386	0.2404	1.0607
2	-0.0379	-0.0873	1.0355
3	-1.2563	0.0254	0.9928
4	-0.3834	-0.0838	1.0285
5	0.2501	-0.0966	1.0360
6	-0.0563	-0.0262	1.0427
7	0.0000	0.0000	1.0417
8	0.4866	0.2533	1.0800
9	-0.2106	-0.1263	1.0272
10	-0.0438	-0.0652	1.0310
11	0.3375	0.0710	1.0570
12	-0.0354	-0.0226	1.0401
13	0.0957	0.1390	1.0500
14	-0.1125	-0.0668	1.0190

Tabela 2 – Podaci o merenjima tokova snaga i struja

Čvor 1	Čvor 2	$P[r.j.]$	$Q[r.j.]$
1	2	0.8281	0.1816
2	4	0.1174	0.0011
2	3	0.6716	0.0946
3	4	-0.6049	0.0536

U prvom slučaju su korišćena sva merenja iz skupa i sva su smatrana realnim, tako da je redundansa telemetrisanih takođe 1.85. Za prag korelacije dva merenja je uzeta vrednost od 0.5.

Tabela 3 – Rezultati u prvom slučaju

Korelacija merenja	6.11700	
Estimirana vrednost optimizacionog kriterijuma - Jx	0.0007	
Prag poređenja funkcije cilja	31.4104	
Broj stepeni slobode	20	
Vrednost merenja bez greške	1.0386	Telemetrisano
Vrednost merenja sa greškom	1.13860	Detektuje
Estimirana vrednost optimizacionog kriterijuma kada postoji greška - Jx	82.6753	
Vrednost reziduala za loše merenje	9.09254	

Kada je redundansa merenja velika i kada su merenja telemetrisana, koristeći test za detekciju i identifikaciju loših merenja, uspešno je identifikovano loše merenje. Primećuje se da je korelacija merenja mala, te je očekivano da su klasični algoritmi efikasni.

U drugom slučaju se posmatra sub-prenosna mreža visoke redundanse ukupnih merenja, ali male redundanse telemetrisanih merenja.

Tabela 4 – Rezultati u drugom slučaju

Ukupna redundansa	1.62963	
Redundansa telemetrisanih merenja	0.40741	
Korelacija merenja	14.0534	
Estimirana vrednost optimizacionog kriterijuma - J_x	0.000032	
Prag poređenja funkcije cilja	23.6848	
Broj stepeni slobode	14	
Vrednost merenja bez greške	1.0386	Pseudo
Vrednost merenja sa greškom	1.13860	Ne detektuje
Estimirana vrednost optimizacionog kriterijuma kada postoji greška - J_x	0.92700	
Vrednost reziduala za loše merenje	0.96278	
Vrednost merenja bez greške	0.8281	Telemetrisano
Vrednost merenja sa greškom	0.7281	Ne detektuje
Estimirana vrednost optimizacionog kriterijuma kada postoji greška - J_x	12.07550	
Vrednost reziduala za loše merenje	1.50646	
Vrednost merenja sa greškom	0.0281	Detektuje
Estimirana vrednost optimizacionog kriterijuma kada postoji greška - J_x	145.93730	
Vrednost reziduala za loše merenje	11.89980	

Ukupna redundansa je i dalje velika ali sada skup merenja čine dominantno pseudo merenja, samim tim, test za detekciju i identifikaciju ne radi sa tačnošću kao u prvom slučaju. Broj stepeni slobode i prag poređenja funkcije cilja su manji, pa je potrebno da greška bude veća da bi bila uočena. Takođe, uočava se da je estimirana vrednost optimizacionog kriterijuma kada postoji greška veća u slučaju kada je greška na telemetrisanom, a ne na pseudo merenju, što znači da se realna loša merenja lakše identifikuju. Stepenn korelacije merenja je porastao sa smanjenjem redundanse i broja telemetrisanih merenja.

U trećem slučaju analizira se sub-prenosna mreža niske ukupne redundanse sa dominantnim pseudo merenjima.

Tabela 5 – Rezultati u trećem slučaju

Ukupna redundansa	1.14815	
Redundansa telemetrisanih merenja	0.11111	
Korelacija merenja	56.40390	
Korelacija merenja – prag 0.9	32.01970	
Estimirana vrednost optimizacionog kriterijuma - J_x	0.000055	
Prag poređenja funkcije cilja	3.8415	
Broj stepeni slobode	1	
Vrednost merenja bez greške	1.0386	Pseudo
Vrednost merenja sa greškom	1.13860	Ne detektuje
Estimirana vrednost optimizacionog kriterijuma kada postoji greška - J_x	0.01860	
Vrednost reziduala za loše merenje	0.10512	
Vrednost merenja bez greške	0.8281	Telemetrisano
Vrednost merenja sa greškom	0.7281	Ne detektuje
Estimirana vrednost optimizacionog kriterijuma kada postoji greška - J_x	0.01710	
Vrednost reziduala za loše merenje	0.04142	
Vrednost merenja sa greškom	2.2281	Detektuje ali ne identifikuje
Estimirana vrednost optimizacionog kriterijuma kada postoji greška - J_x	11.94060	
Vrednost reziduala za loše merenje	2.99456	

Uočava se da test za detekciju i identifikaciju loših merenja nije dovoljno robustan da može da radi kada se redundansa ukupnih merenja približava jedinici, a redundansa telemetrisanih merenja je približno 0.1 (kao u distributivnim mrežama). Detekcija loših merenja nije efikasna kao u prethodnim slučajevima, a i kada je moguće detektovati merenje sa greškom, teško ga je sa tačnošću identifikovati. Korelacija merenja je drastično porasla, smanjenjem redundanse, čime su i greške merenja mnogo više korelisane.

6. ZAKLJUČAK

U radu je korišćen Hi-kvadrat test za detekciju loših merenja i test najvećeg normalizovanog reziduala za identifikaciju loših merenja koji su razvijani za prenosnu mrežu, a primenjeni na različitim konfiguracijama merenja sub-prenosne mreže. Iz prikazanih rezultata se može videti da kad je ukupna redundansa merenja velika algoritam za detekciju i identifikaciju loših merenja može da izoluje merenje sa greškom, iako je redundansa telemetrisanih merenja mala. Kada se ukupna redundansa smanji i postane bliska redundansi distributivne mreže, ovaj algoritam više nije dovoljno robustan da detektuje loše merenje sa dovoljnom tačnošću. Merenja postaju više korelisana, pa samim tim i njihove greške što rezultuje identifikovanjem tačnog merenja kao lošeg i obrnuto.

7. LITERATURA

- [1] A. Abur, A. G. Expósito, „*Electric Power System State Estimation: Theory and Implementations*“, Marcel Dekker, New York, 2004.
- [2] V. Strezoski, „*Analiza elektroenergetskih sistema I*“, skripta, Fakultet Tehničkih Nauka, Novi Sad, 2011.
- [3] V. Strezoski, „*Analiza elektroenergetskih sistema II*“, skripta, Fakultet Tehničkih Nauka, Novi Sad, 2011.
- [4] A. Monticelli, „*State estimation in electric power systems, a generalized approach*“, University of Campinas, Norwell, 1999.
- [5] V. Krsman, „*Specijalizovani algoritmi za detekciju, identifikaciju i estimaciju loših podataka u elektrodistributivnim mrežama*“, doktorska disertacija, Fakultet Tehničkih Nauka, Novi Sad, 2017.
- [6] https://www2.ee.washington.edu/research/pstca/pf14/pg_tca14bus.htm, IEEE 14 Bus Test Case

Kratka biografija:

Milica Jovičić rođena je u Novom Sadu 1992. godine. Master rad je odbranila 2018. godine na Fakultetu tehničkih nauka iz Elektrotehnike i računarstva – Elektroenergetski sistemi.

Vladan Krsman rođen je u Sarajevu 1985. godine. Doktorsku disertaciju odbranio je 2017. godine na Fakultetu tehničkih nauka iz oblasti Elektroenergetski sistemi.

**RAZVOJ VEB BAZIRANOG SISTEMA ZA PRORAČUN TOKOVA SNAGA
DISTRIBUTIVNE MREŽE****DEVELOPMENT OF THE WEB BASED SYSTEM FOR THE LOAD FLOW
CALCULATION IN DISTRIBUTION NETWORK**

Milivoje Popovac, *Fakultet tehničkih nauka, Novi Sad*

**Oblast – INFORMACIONO – KOMUNIKACIONI
SISTEMI**

Kratak sadržaj – U ovom radu dat je pregled potencijalnih tehnologija za razvoj veb aplikacije koja će da vrši proračun tokova snaga distributivne mreže. Razvijena je predložena aplikacija koristeći neke od danas najpopularnijih tehnologija (Angular, Node.js, MongoDB). Prikazani su postupci projektovanja i razvijanja funkcionalnosti. Iznet je zaključak o realizovanoj aplikaciji i opisane mogućnosti za njen dalji razvoj.

Ključne reči: Tokovi snaga, veb aplikacija, Angular, Node.js, MongoDB

Abstract – This paper gives an overview of the potential technologies for developing a web application that will perform the calculation of the load flow of the distribution network. The proposed application has been developed using some of today's most popular technologies (Angular, Node.js, MongoDB). Procedures for designing and developing functionalities are presented. The conclusion about the implemented application and the described possibilities for its further development was made.

Keywords: Load Flow, Web application, Angular, Node.js, MongoDB

1. UVOD

Od nastanka prvog mikroprocesora - početkom 1970-ih godina - do danas, nauka i tehnologija su prešle značajan put. Brz razvoj tehnologije uticao je na sve sfere života toliko da se poslovanje danas ne može zamisliti bez iste. Uloga čovjeka se značajno promenila, mašine se već uveliko koriste da olakšaju fizički rad, a pametni računari polako zamjenjuju i čovekove intelektualne sposobnosti, stoga treba i iskoristiti sve mogućnosti koje nam se pružaju. Procesi i sistemi se usložnjavaju i neophodno je da tehnologija bude u skladu sa poslovnim ciljevima, te je potrebno uvesti automatizaciju gdje god je to moguće. Ni elektroenergetski sistemi nisu ništa drugačiji. Softverski alati u mnogome poboljšavaju performanse elektroenergetskog sistema, i trenutno živimo u vremenu kada imaju ogroman značaj, a očekuje se da će taj značaj samo još više da raste.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Srđan Popović.

Softverski alati su našli veliku primenu u planiranju, održavanju i radu jednog elektroenergetskog sistema. Danas najmoćnije kompanije koje upravljaju radom elektroenergetskih sistema su upravo one koje su čvrsto spregnute sa softverom kao takvim. Automatizacija distributivnih mreža (DM) i primena složenih softverskih alata dovela je do niza pogodnosti.

2. PRORAČUN TOKOVA SNAGA

Problem tokova snaga u suštini se odnosi na problem bilansa električne energije u elektroenergetskim sistemima prenosno-proizvodnih, cijelih i delova DM. Proračun tokova snaga predstavlja proračun promenljivih stanja (režima) mreže, gde su poznati izvor napajanja mreže i potrošnje u svim čvorovima mreže [1]. Na osnovu vrednosti modula i faznih stavova napona proračunavaju se preostale vrednosti koje nisu unapred zadate (npr. aktivne i reaktivne snage injektiranja u čvorovima, gubici snage, itd.). Na ovaj način definisan je celokupan režim DM.

Distributivne mreže (DM), za razliku od prenosnih mreža, karakteriše radijalan pogon i relativno slaba upetljanost. Odnos R/X kod vodova i kablova u DM je daleko viši nego u prenosnim mrežama.

Iz ovih razloga standardne metode za proračun tokova snaga u prenosnim mrežama imaju znatno manju efikasnost kada se primenjuju u DM. Za proračun tokova snaga distributivne mreže razvijeni su specijalizovani algoritmi koji su orjentisani ka granama.

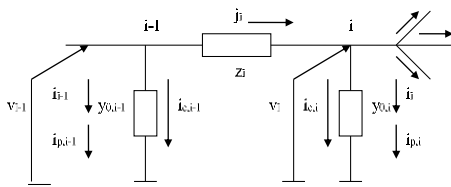
Jedan od njih jeste i algoritam sumiranja struja, poznatiji kao Shirmohammadi-ev algoritam.

2.1. Algoritam sumiranja struja

Shirmohammadi-ev algoritam predstavlja iterativni postupak za proračun tokova snaga u radijalnim DM. Algoritam počinje sa inicijalizacijom postupka, odnosno učitavanjem podataka o distributivnom mreži, numeraciji grana i čvorova po lejerima i setovanja indeksa iteracija na početnu vrednost $h=1$. Nakon same inicijalizacije započinje iterativni postupak. Svaka iteracija sastoji se od tri koraka:

1. Proračun injektiranih struja,
2. Proračun struja po granama,
3. Proračun napona po čvorovima.

Radi lakšeg razumevanja sledećeg dela izlaganja, na slici 1 je prikazana ekvivalentna šema jedne grane radijalne DM sa usvojenim referentnim smerovima.



Slika 1. Ekvivalentna šema grane [2]

U prvom koraku (proračun injektiranih struja) proračuni se izvode za svaki čvor, počevši od čvora u prvom lejeru, kao u izrazu (1):

$$i_i^{(h)} = \left(\frac{S_{p,i}}{v_i^{(h-1)}} \right)^* + y_{0,i} * v_i^{(h-1)}, i = 1, \dots, n_{cv} \quad (1)$$

U drugom koraku vrši se proračun struja po granama pri čemu se ovaj proračun izvodi za svaku granu, počevši od grana u posljednjem lejeru, prema relaciji (2):

$$j_i^{(h)} = i_i^{(h)} + \sum_{j \in i} j_j^{(h)}, j = n_{gr}, \dots, 1 \quad (2)$$

Poslednji korak se sastoji od proračuna napona u čvorovima, pri čemu se proračun izvodi za svaki čvor, počevši od čvora u prvom lejeru, prema izrazu (3):

$$v_i^{(h)} = v_{i-1}^{(h)} - z_i * j_i^{(h)}, i = 1, \dots, n_{cv} \quad (3)$$

Nakon svake iteracije ispituju se uslovi konvergencije koji su zadovoljeni kada je vrednost maksimalnog debalansa aktivne i reaktivne snage manja od unapred specificirane vrijednosti (relacije (4) i (5)):

$$D_p^{(h)} < \varepsilon \quad (4)$$

$$D_q^{(h)} < \varepsilon \quad (5)$$

Ukoliko su uslovi konvergencije ispunjeni, iterativni postupak se završava. U suprotnom, prelazi se na sledeću iteraciju.

3. PREGLED KORIŠĆENIH TEHNOLOGIJA

Tehnologije korišćene u ovom radu, za izradu veb aplikacije koja će da vrši proračun tokova snaga distributivne mreže su:

1. Angular
2. Node.js
3. Express
4. MongoDB

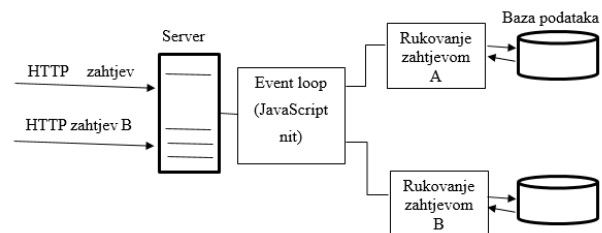
Angular je moderna platforma za veb aplikacije koja obezbeđuje programerima rad sa sveobuhvatnim skupom alata i mogućnosti za izgradnju velikih, robusnih aplikacija. Osnovna prednost Angular-a jeste da omogućava izgradnju aplikacija koje rade skoro na bilo kojoj platformi - bilo mobilnom, vebu ili desktopu. Angular tim se fokusirao na izgradnju mnogo više od samo robusnog aplikacionog okruženja, oni su izgradili čitav ekosistem.

Angular se koristi za kreiranje Single Page Aplikacija (SPAs) – samo jedan HTML fajl i skup JavaScript kodova dobijenih od strane servera. SPAs učitavaju jednu HTML stranicu i dinamički ažuriraju tu stranicu pošto korisnik stupa u interakciju sa aplikacijom. Sve što se vidi, svaka promena se renderuje u pretraživaču čime se daje korisniku reaktivno iskustvo. Međutim, to znači da se

veliki deo posla dešava na strani klijenta, u JavaScript-i [3].

Node.js je open source serverska platforma koja omogućava izgradnju brzih i skalabilnih mrežnih aplikacije koje koriste JavaScript. Node.js je izgrađen na vrhu V8, savremene JavaScript virtualne mašine koja upravlja Google-ovim Chrome veb pretraživačem. Node.js koristi event driven, ne blokirajući I/O model koji ga čini lakšim i efikasnim. Može upravljati višestrukim istovremenim mrežnim konekcijama sa malim troškovima, što ga čini idealnim za aplikacije koje zahtevaju podatke u realnom vremenu[4].

Node.js koristi Event loop (koristi se u GUI aplikacijama) koji omogućuje da se poziva funkcija kao reakcija na događaj, prikazan je na slici 2. Nakon izvršenja funkcije ili se poziva nova funkcija za događaj koji se već desio (ako je u queue-u) ili se čeka da se desi novi događaj.

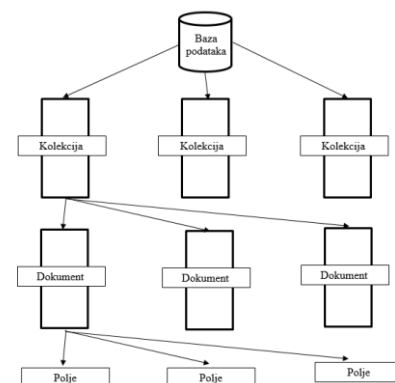


Slika 2. Princip rada Node.js-a[4]

Veb sajt Express-a opisuje Express kao „minimalno fleksibilno Node.js okruženje za veb aplikacije, pružajući robusan skup funkcija za izgradnju jednostranih i višestranih i hibridnih veb aplikacija“[5].

Express ima dizajn baziran na middleware-u. Middleware se opisuje kao „softverski lepak“, koristi se najčešće za softvere koji obezbeđuju komunikaciju i upravljanje podacima u distribuiranim aplikacijama. To su funkcije koje se pozivaju prije nego što se vrši rukovanje zahtevima. Neki od najčešće korišćenih su: body-parser, cookie-parser, multer itd.

MongoDB predstavlja cross-platform dokument orjentisanu bazu podataka napisanu u C++. Dokumenti koji se smeštaju u ovoj bazi se čuvaju u BSON formatu - binarna reprezentacija jednostavnih struktura podataka i asocijativnih listi (Binary JSON). Najpopularniji NoSQL DBMS, od marta 2017. peta po popularnosti DBMS (posle Oracle, MySQL, Microsoft SQL Server i PostgreSQL-a). Predstavlja odličnu podršku u Node.js aplikacijama (deo MEAN stack-a)[6]. Struktura MongoDB baze je prikazana na slici 3.



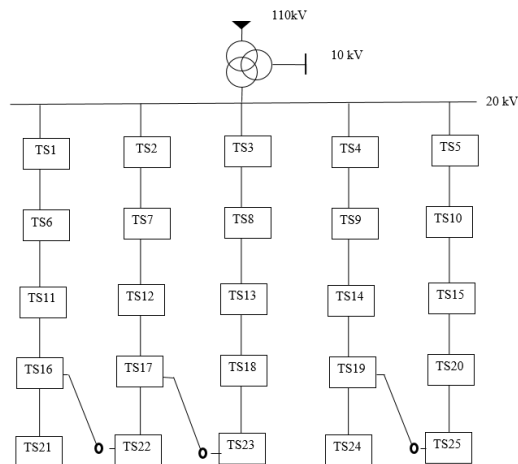
Slika 3. Struktura baze podataka u MongoDB

4. ARHITEKTURA SISTEMA

U ovom dijelu opisan je način na koji ćemo vršiti proračune tokova snaga distributivne mreže, pri tom koristeći veb baziran sistem.

4.1. Opis inicijalne test DM

Test mreža nad kojom će se vršiti, inicijalno, proračun tokova snaga prikazana je na slici 4. U ovom radu, osim proračuna tokova snaga biće opisan i način dodavanja novih potrošača DM, njihovo brisanje, kao i ažuriranje snaga potrošača.



Slika 4. Inicijalna test DM mreža[7]

U tabeli 1 dat je opis svih čvorova distributivne mreže.

Tabela 1. Čvorovi mreže[7]

ID čvora	P _p [r.j.]	Q _p [r.j.]	ID čvora	P _p [r.j.]	Q _p [r.j.]
100	0	0	115	0.7	0.14
101	0.5	0.10	116	0.5	0.10
102	0.8	0.16	117	0.8	0.16
103	0.9	0.18	118	0.8	0.16
104	0.5	0.10	119	0.4	0.08
105	0.7	0.14	120	0.7	0.14
106	0.5	0.10	121	0.5	0.10
107	0.8	0.16	122	0	0
108	0.9	0.18	123	0.3	0.06
109	0.5	0.10	124	0	0
110	0.7	0.14	125	0.2	0.04
111	0.5	0.10	126	0.4	0.08
112	0.8	0.16	127	0	0
113	0.9	0.18	128	0.2	0.04
114	0.5	0.10			

4.2. Izrada backend dela aplikacije

Osnova za dalji razvoj aplikacije jeste manipulisanje podacima iz baze podataka, u koju će se vršiti upisivanje, čitanje, modifikovanje i brisanje podataka. Ove operacije se nazivaju CRUD operacije (create, read, update, delete). Korišćenjem ovih operacija, automatski se dešavaju promene u bazi podataka. Da bi omogućili korišćenje ovih operacija, potrebno je razviti backend deo aplikacije. Prvo što treba uraditi jeste kreirati server pomoću jednog od standardnih paketa koji dolaze u sklopu instalacije

Node.js, http, a zatim definisati kakve zahtjeve će isti da prima, kakve odgovore će slati u zavisnosti od zahtjeva, kao i port na kome će dati server da osluškuj.

MongoDB baza spada u NoSQL baze, i njih karakteriše nepostojanje strukture podataka koji se skladište u istu. Međutim, u ovom radu svaka kolekcija koja se bude nalazila u bazi će imati neku strukturu (schemu), a paket koji to obezbeđuje jeste moongoose (dozvoljava definisanje objekata sa jasno određenim tipovima i šemom koja se mapira na MongoDB dokument). Na slici 5 prikazan je izmodelovan čvor mreže.

```
var mongoose = require('mongoose');
var Schema = mongoose.Schema;

var schema = new Schema({
  idNode: {type: Number, required: true},
  Pp: {type: Number, required: true},
  Qp: {type: Number, required: true}
});

module.exports = mongoose.model('Node', schema);
```

Slika 5. Modelovanje čvora DM

Slično, modeluje se i grana mreže. Osim prethodno navedenih modela, postoji i šema koja modeluje korisnika koji će se moći prijaviti kako bi koristio aplikaciju (prikazan na slici 6).

```
var mongoose = require('mongoose');
var Schema = mongoose.Schema;
var mongooseUniqueValidator = require('mongoose-unique-validator');

var schema = new Schema({
  username: {type: String, required: true, unique: true},
  role: {type: String, required: true},
  password: {type: String, required: true}
});

schema.plugin(mongooseUniqueValidator);

module.exports = mongoose.model('User', schema);
```

Slika 6. Model korisnika aplikacije

Kada pristigne zahtjev postavlja se pitanje kako će server da zna koji odgovor treba da vrati a da bude odgovarajući poslatom zahtjevu. Mehanizam koji nam ovo omogućava se naziva rutiranje. Rutiranje se odnosi na to kako aplikacija odgovara na klijentske zahtjeve u zavisnosti od endpoint-a, vrste HTTP zahtjeva (get, post.), URI-a itd.

U ovom dijelu biće opisane CRUD operacije nad kolekcijom koja skladišti podatke o čvorovima mreže.

```
router.get('/', function (req, res, next) {
  router.patch('/:id', function (req, res, next) {
    router.delete('/:id', function (req, res, next) {
      Node.findById(req.params.id, function (err, node) {
        if (err) {
          return res.status(500).json({
            title: 'An error occurred',
            error: err
          });
        }
        return res.status(200).json({
          message: 'Success',
          obj: node
        });
      });
    });
  });
});

router.post('/', function (req, res, next) {
  var node = new Node({
    idNode: req.body.idNode,
    Pp: req.body.Pp,
    Qp: req.body.Qp
  });
  node.save(function (err, result) {
    if (err) {
      return res.status(500).json({
        title: 'An error occurred',
        error: err
      });
    }
    res.status(200).json({
      message: 'Node saved',
      obj: result
    });
  });
});

router.patch('/:id', function (req, res, next) {
  Node.findById(req.params.id, function (err, node) {
    if (err) {
      return res.status(500).json({
        title: 'An error occurred',
        error: err
      });
    }
    if (!node) {
      return res.status(500).json({
        title: 'No Node Found',
        error: 'Node not found'
      });
    }
    node.Pp = req.body.Pp;
    node.Qp = req.body.Qp;
    node.save(function (err, result) {
      if (err) {
        return res.status(500).json({
          title: 'An error occurred',
          error: err
        });
      }
      res.status(200).json({
        message: 'Updated node',
        obj: result
      });
    });
  });
});

router.delete('/:id', function (req, res, next) {
  Node.findById(req.params.id, function (err, node) {
    if (err) {
      return res.status(500).json({
        title: 'An error occurred',
        error: err
      });
    }
    if (!node) {
      return res.status(500).json({
        title: 'No Node Found',
        error: 'Node not found'
      });
    }
    node.remove(function (err, result) {
      if (err) {
        return res.status(500).json({
          title: 'An error occurred',
          error: err
        });
      }
      res.status(200).json({
        message: 'Deleted node',
        obj: result
      });
    });
  });
});
```

Slika 7. Node Ruter

Kada pristigne zahtjev, pronalazi se odgovarajuća ruta a zatim se izvršava odgovarajuća callback funkcija. U sva četiri slučaja vrši se prvo provera da li je došlo do greške na serveru, ukoliko nije odgovor nastavlja svoj tok. zahtjev za čitanje podataka o čvorovima, pomoću metode find vrši se pronalaženje svih dokumenata unutar kolekcije nodes.

Post zahtjev na rutu node reprezentuje dodavanje novog potrošača u mrežu. Najpre se vrši kreiranje novog čvora (objekat tipa modela Node), a zatim i njegovo pamćenje u bazu podataka. Metode za brisanje i modifikaciju čvora vrše brisanje na osnovu prosleđenog id-a. Kao rezultat proračuna tokova snaga očekuju se promenljive stanja na osnovu kojih se može proračunati cjelokupni režim mreže. Naravno to predstavljaju naponi čvorova. Nakon iščitavanja svih čvorova i grana iz baze podataka vrši se proračun tokova snaga primjenom algoritma sumiranja struja. Implementacija algoritma je prikazana na slici 7.

```
for (var h = 0; h < maxIter; h++) {
  for (var i = 1; i < nodes.length; i++) {
    Inode[i] = math.divide(Sp_rj[i], prevNodeInRootv[i]), math.multiply(Vnode[i], prevNodeInRootv[i]);
  }

  for (var i = branches.length - 1; i >= 0; i--) {
    for (var j = 1; j < nodes.length; j++) {
      if (branches[i].node2ID == nodes[j].idNode) {
        jbranch[i] = Inode[j];
        for (var k = 0; k < branches.length; k++) {
          sum3branch[k] = math.complex(0, 0);
        }
      }

      for (var j = 1; j < nodes.length; j++) {
        if (branches[i].node2ID == nodes[j].idNode) {
          for (var k = 0; k < branches.length; k++) {
            if (branches[k].node1ID == nodes[j].idNode) {
              sum3branch[k] = math.add(sum3branch[k], jbranch[k]);
            }
          }
        }
      }
      jbranch[i] = math.add(jbranch[i], sum3branch[i]);
    }
  }

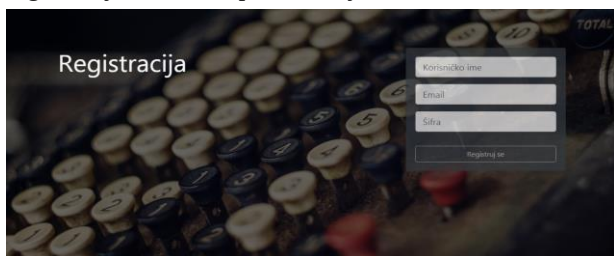
  for (var i = 1; i < nodes.length; i++) {
    for (var j = 1; j < nodes.length; j++) {
      if (branches[i-1].node1ID == nodes[j].idNode) {
        tempv[i] = Vnode[j];
      }
    }
    Vnode[i] = math.subtract(tempv[i], math.multiply(jbranch[i-1], jbranch[i-1]));
    prevNodeInRootv[i] = Vnode[i];
  }

  for (var i = 1; i < nodes.length; i++) {
    S[i] = math.subtract(math.multiply(Vnode[i], math.conj(Inode[i])),
      math.multiply(math.conj(Vnode[i]), math.multiply(math.abs(Vnode[i]), math.abs(Vnode[i]))));
    tempd[i-1] = math.re(S[i]) - math.re(Sp_rj[i]);
    tempdQ[i-1] = (math.im(math.subtract(S[i], Sp_rj[i])));
  }
}
```

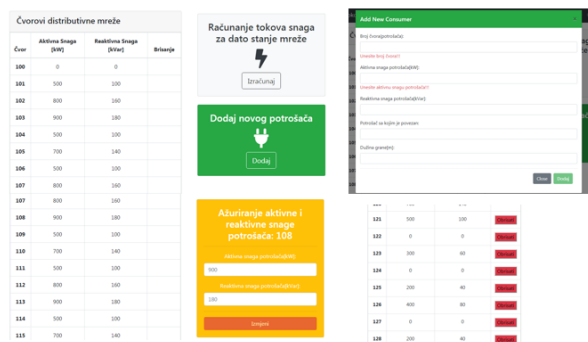
Slika 7. Proračun tokova snaga DM

4.3. Izrada frontend dela aplikacije

Ukoliko bilo koji korisnik želi da koristi aplikaciju, prethodno mora biti registrovan u bazi podataka. Takođe, postoje različite uloge, običan korisnik i administrator, pri čemu je administratoru omogućena potpuna manipulacija mrežom, dok običan korisnik može da samo dobija informacije o trenutnom stanju mreže. Stranica za registraciju korisnika prikazana je na slici 8.



Slika 8. Registracija korisnika



Slika 9. CRUD operacije nad čvorovima mreže

Nakon uspešne prijave, moguće je koristiti aplikaciju u svrhu kako manipulacije cijelom mrežom, tako i vršenja

proračuna tokova snaga. Na slici 9 su prikazane sve četiri CRUD operacije nad čvorovima(potrošačima) mreže.

Obezbeđena je validacija bilo kakvog unosa ili modifikacije, i bilo kakve izmjene ne smeju da utiču na narušavanje radikalne strukture mreže.

Promenljive stanja (naposljetku po čvorovima)				Stanje grana DM				Stanje grana DM			
Čvorovi	Stanje	Stanje	Stanje	Čvorovi	Stanje	Stanje	Stanje	Čvorovi	Stanje	Stanje	Stanje
1000	0	0	0	1000	0	0	0	1000	0	0	0
1001	100	100	100	1001	100	100	100	1001	100	100	100
1002	100	100	100	1002	100	100	100	1002	100	100	100
1003	100	100	100	1003	100	100	100	1003	100	100	100
1004	100	100	100	1004	100	100	100	1004	100	100	100
1005	100	100	100	1005	100	100	100	1005	100	100	100
1006	100	100	100	1006	100	100	100	1006	100	100	100
1007	100	100	100	1007	100	100	100	1007	100	100	100
1008	100	100	100	1008	100	100	100	1008	100	100	100
1009	100	100	100	1009	100	100	100	1009	100	100	100
1010	100	100	100	1010	100	100	100	1010	100	100	100
1011	100	100	100	1011	100	100	100	1011	100	100	100
1012	100	100	100	1012	100	100	100	1012	100	100	100
1013	100	100	100	1013	100	100	100	1013	100	100	100
1014	100	100	100	1014	100	100	100	1014	100	100	100
1015	100	100	100	1015	100	100	100	1015	100	100	100
1016	100	100	100	1016	100	100	100	1016	100	100	100
1017	100	100	100	1017	100	100	100	1017	100	100	100
1018	100	100	100	1018	100	100	100	1018	100	100	100
1019	100	100	100	1019	100	100	100	1019	100	100	100
1020	100	100	100	1020	100	100	100	1020	100	100	100

Slika 10. Rezultati proračuna tokova snaga

5. ZAKLJUČAK

U ovom radu prikazan je razboj veb aplikacije za proračun tokova snaga distributivne mreže. Aplikacija pruža mogućnost manipulacije nad cijelom distributivnom mrežom. Omogućeno je dobijanje informacija o trenutnoj konfiguraciji mreže (čvorovi, grane mreže). Takođe, obezbeđeno je dodavanje novih čvorova (potrošača) i grana, brisanje čvorova i grana kao i modifikacija aktivne i reaktivne snage potrošnje u čvorovima sistema. Cilj ovog rada jeste definitivno bio u implementaciji algoritma za proračun tokova snaga. Pokazano je i da se za takve kompleksne proračune mogu koristiti prethodno opisane tehnologije. Trenuto je na snazi prelazak sa desktop aplikacija na veb aplikacije, i sugurno je da će u neko dogledno vreme sve aplikacije izvršiti ovu tranziciju. Ni softeri koji omogućavaju rad sa elektroenergetskim sistemima nisu ništa drugačiji. Kao korak posle ovoga smatra se prelazak na Android aplikacije, zbog tendencije da se u poslednje vrijeme jako popularizuje upotreba mobilnih uređaja.

6. LITERATURA

- [1] V.C.Strezoski, "Osnovi Elektroenergetike" Fakultet tehničkih nauka u Novom Sadu, Novi Sad, 2014.
- [2] N.Katić, "Menadžment sistemi u elektroenergetici – DMS", skripta sa predavanja, Fakultet tehničkih nauka, Novi Sad, novembar 2013.
- [3] <https://angular.io/> (pristupljeno u maju 2018.)
- [4] <https://nodejs.org/en/> (pristupljeno u maju 2018.)
- [5] Jason Krol, "Web Development with MongoDB and Node.js", Packt Publishing Ltd, Birmingham, 2014
- [6] <https://www.mongodb.com/> (pristupljeno u maju 2018.)
- [7] D.Babić, "Optimalna rekonfiguracija distributivnih mreža, master rad", Fakultet tehničkih nauka u Novom Sadu, Novi Sad, 2016.

Kratka biografija:



Milivoje Popovac rođen je u Gacku 1993. god. 2012 godine upisuje osnovne akademske studije na FTN-u Novi Sad, smer Energetika, elektronika i telekomunikacije. 2016. godine upisuje master akademske studije na istom fakultetu, smer Elektroenergetika-Elektroenergetski sistemi.

UPRAVLJANJE AKTIVNIM ISPRAVLJAČEM ZASNOVANIM NA P-REZONANTNIM REGULATORIMA

P-RESONANT REGULATOR BASED ON ACTIVE RECTIFIER CONTROL

Vladimir Popov, Marko Vekić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Ovaj rad opisuje postepeni razvoj upravljačkog sklopa aktivnog ispravljača. Upravljački sklop je zasnovan na P-Rezonantnim regulatorima struje i omogućava tok sinusoidalnih struja prema mreži čak i u slučaju da je mrežni napon harmonijski izobličen.

Ključne reči: Pretvarač povezan na električnu mrežu, aktivni ispravljač, upravljanje, P-rezonantni regulator

Abstract – This paper presents the systematic development of active rectifier control design. The control structure is based on P-Resonant regulators. Proposed active rectifier control structure draws sinusoidal grid currents even in the case when utility voltage is polluted by higher harmonics.

Keywords: Grid-Connected Power Converter, Active Power Rectifier, P-Resonant controller

1. UVOD

Pretvarači energetske elektronike sve više dobijaju značaj u primenama koje su vezane za električnu mrežu. U početku je njihova primena bila ograničena na HVDC i u manjoj meri na statičke VAR kompenzatore (SVC). Međutim, od kasnih 1980-ih godina primene u proizvodnji, prenosu i distribuciji postaju sve značajnije, posebno u smislu kvaliteta isporučene i primljene električne energije. Uređaji energetske elektronike postaju nezaobilazan sprežni element obnovljivih izvora ka mreži (vetroelektrane, solarne elektrane itd), ali i daju podršku stabilnosti kako distribuciji (npr. D-STATCOM), tako i u prenosu (čitava klasa FACTS uređaja). S druge strane, pretvarači energetske elektronike u velikoj meri svojom masivnom primenom učestvuju u utiskivanju viših harmonika u mrežu. Najveći doprinos tome daju neupravljivi (diodni) ispravljači koji su često nezaobilazan sklop pogona i uređaja. Polazeći od preporuka i standarda IEEE519 i IEC-555 [1], koje određuju granice harmonijskog naprezanja u sabirnici priključenja pretvarača, jasno je da treba koristiti takve ispravljače koji ne unose dodatno izobličenje, a to su upravljivi ispravljači. Oni se mogu podeliti po načinu komutacije na mrežno-komutovane (tiristorske) i prinudno komutovane PWM ispravljače. Detaljnija podela i poređenja se mogu naći u [2]. PWM ispravljači omogućuju punu upravljivost što otvara mogućnost da se razmenjuju sinusoidalne struje s mrežom i nadzire sačinilac snage. Poseban izazov za upravljanje aktivnim ispravljačem predstavlja pojava viših harmonika mreže.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Marko Vekić, docent.

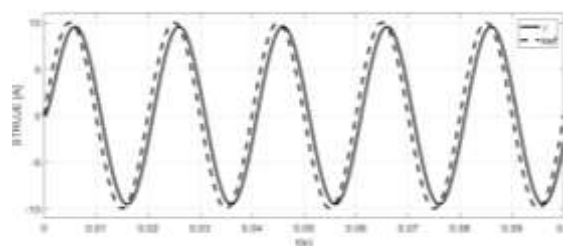
Tema ovoga rada jeste upravljanje aktivnim ispravljačem na način da povlači sinusoidalnu struju, čak i u uslovima visokoizobličenog napona mreže u sabirnici priključenja.

2. UPRAVLJANJE PRETVARAČEM NA MREŽI U $\alpha\beta$ OBLASTI

Upravljanje pretvaračem, bilo da je on deo nekog elektromotornog pogona ili povezan na električnu mrežu se obično vrši u tzv. obrtnom dq koordinatnom sistemu. U ovom radu će biti primenjena kontrola u mirujućoj $\alpha\beta$ oblasti čime se izbegava upotreba PLL-a. Osim toga, kako će se videti projektovanje regulacije se jednostavno nadograđuje za poništavanje ciljanih viših harmonika mreže. Običan PI regulator ne može da uspešno prati sinusnu referencu. Može se analitički pokazati da bi struja odziv struje koja teče kroz RL potrošač imala oblik *Iravani*:

$$i(t) = Ce^{\frac{-t}{\tau_i}} + \frac{I}{\sqrt{1 + (\omega_0 \tau_i)^2}} \cos(\omega_0 t + \varphi - \delta) \quad (1)$$

Gde su: $\delta = \tan^{-1}(\omega_0 \tau_i)$ i $\tau_i = \frac{K_p}{K_i}$, a K_p i K_i su redom proporcionalno i integralno dejstvo PI regulatora izabrani tako da je: $\frac{K_p}{K_i} = \frac{L}{R}$. Iz gornje jednačine je jasno da se postoji greška u ustaljenom stanju, kako u amplitudi, tako i u faznom stavu (slika 1) koja se može smanjivati povećanjem τ_i .

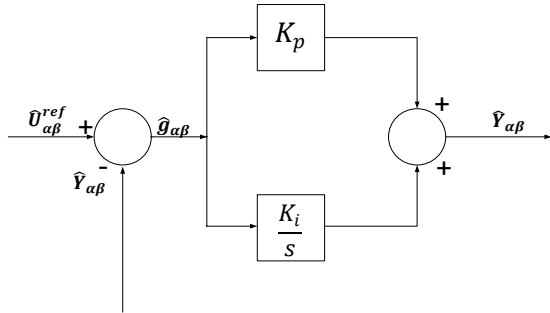


Slika 1. Greška ustaljenog stanja

Međutim, postoji jasna granica koliko daleko možemo ići sa smanjenjem, kako se ne bi propustio i šum PWM-a. Sve ovo govori u prilog tome da se mora još nešto uraditi kako bi se PI regulatorom u $\alpha\beta$ oblasti mogla pratiti sinusna referenca. Jedno rešenje jeste projektovanje kompenzatora koji bi imao rezonantne nule na svakoj neželjenoj učestanosti tako "zarobljavao" neželjene više harmonike, na sličan način kao što fizičko LC kolo "zarobi" datu rezonantnu učestanost. Takvo rešenje, iako sasvim ostvarljivo, nije dovoljno praktično u smislu jednostavne proširivosti na dodatne harmonike ukoliko je potrebno. Zato ćemo primeniti tzv. P-Rezonantne regulatore [3].

2.1. P-Rezonantni regulator

Da bismo došli do regulatora u mirujućem ($\alpha\beta$) sistemu, polazimo od standardnog PI regulatora u obrtnoj (dq) oblasti. Zbog pojednostavljivanja izvođenja, korist ćemo kompleksni zapis koji suštinski uvažava komponente po obeose (d i q , odnosno α i β).



Slika 2. Izvođenje P-Rezonantnog regulatora

Na gornjoj slici su: $U_{\alpha\beta}^{ref}$ ulaz, odnosno reference po obe ose (uslovno možemo to smatrati fazorom reference), $\hat{Y}_{\alpha\beta}$ je izlaz po obe ose, dok je $\hat{G}_{\alpha\beta}$ fazor greške, tj. Greška po obe ose. Na osnovu slike 2 se može napisati:

$$\frac{d\hat{Y}_{dq}}{dt} = K_p \frac{d\hat{G}_{dq}}{dt} + K_i \hat{G}_{dq} \quad (2)$$

U $\alpha\beta$ oblast prelazimo na osnovu izraza: $\hat{X}^{\alpha\beta} = \hat{X}^{dq} e^{j\theta}$, pa (2) postaje:

$$\frac{d(\hat{Y}^{\alpha\beta} e^{-j\theta})}{dt} = K_p \frac{d(\hat{G}^{\alpha\beta} e^{-j\theta})}{dt} + K_i \hat{G}^{\alpha\beta} e^{-j\theta} \quad (3)$$

Diferencirajući gornju jednačinu, a zatim primenjujući Laplace-ovu transformaciju i daljim sređivanjem, dobija se funkcija prenosa:

$$\frac{\hat{Y}^{\alpha\beta}}{\hat{G}^{\alpha\beta}} = \frac{K_p(s - j\omega) + K_i}{s - j\omega} = K_p + \frac{K_i}{s - j\omega} \quad (4)$$

Nakon racionalizacije imamo:

$$\frac{\hat{Y}^{\alpha\beta}}{\hat{G}^{\alpha\beta}} = K_p + \frac{K_i s}{s^2 + \omega^2} + j \frac{K_i \omega}{s^2 + \omega^2} \quad (5)$$

odnosno:

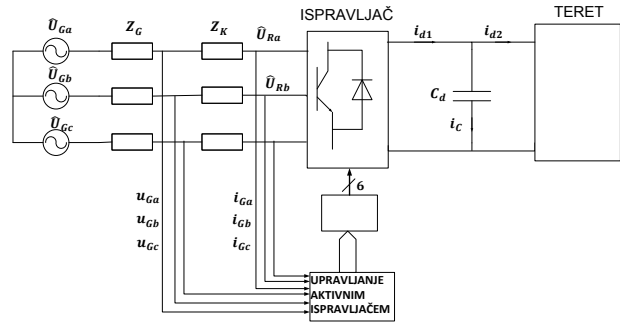
$$\begin{aligned} y_\alpha &= \left(K_p + \frac{K_i s}{s^2 + \omega^2} \right) g_\alpha - \frac{K_i \omega}{s^2 + \omega^2} g_\beta \\ y_\beta &= \left(K_p + \frac{K_i s}{s^2 + \omega^2} \right) g_\beta + \frac{K_i \omega}{s^2 + \omega^2} g_\alpha \end{aligned} \quad (6)$$

Oдавde se jasno vidi rezonantno dejstvo na učestanosti ω . Takođe se vidi izlaz regulatora po datoj osi zavisi od greške po toj osi, ali i greške po suprotnoj osi, na sličan način kao što je to slučaj u obrtnoj dq oblasti [4].

3. UPRAVLJANJE AKTIVNIM ISPRAVLJAČEM

Osnovni zadatak aktivnog ispravljača koji se u ovom radu razmatra jeste da obezbedi stalan jednosmerni napon na izlazu, ali uz dodatni uslov da struje koje povlači iz mreže budu što bliže sinusnom obliku, tj. da ispravljač ne prlja mrežu čak i u uslovima kada mrežni napon sadrži više harmonike. Možemo naslutiti da ostvarenje sinusnih struja u uslovima kada se pretvarač (ispravljač) upravlja postupkom PWM-anije teško ostvariti, pošto je način rada

uporediv, odnosno analogan slučaju kada se upravlja invertorom. Svrha upravljanja je naravno obrnuta, tj. inverzna ili invertovana (DC u AC), ali posledice takvog upravljanja su sinusne struje (ukoliko potrošač ima izraženu induktivnost). Na osnovu toga jasno je da će i u slučaju ispravljačkog rada mrežne struje (umesto potrošačkih) biti sinusne. Ovo međutim važi u slučaju da su naponi mreže simetrični i nezagađeni harmonicima. U slučaju prisustva viših harmonika, upitno je kako bi se standardni načini upravljanja borili sa ovakvim poremećajima. Upravo taj izazov će biti razmatran u ovom poglavlju.

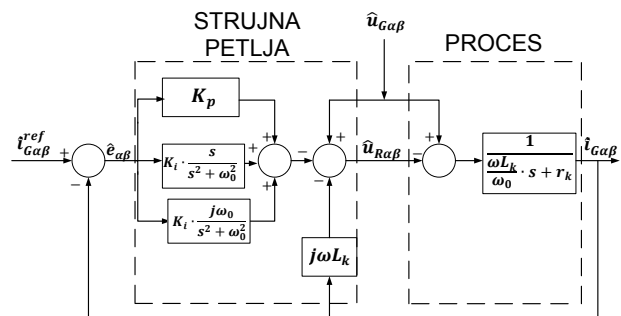


Slika 3. Aktivni ispravljač

Sklop aktivnog ispravljača se može opisati jednačinom:

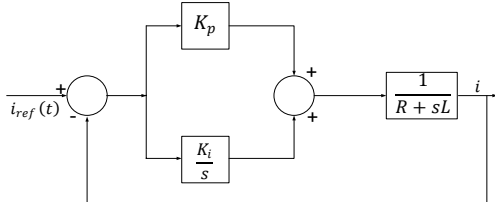
$$\hat{u}_{G\alpha\beta} = R_k \hat{i}_{G\alpha\beta} + L_k \frac{d\hat{i}_{G\alpha\beta}}{dt} + \hat{u}_{R\alpha\beta} \quad (7)$$

U izradi upravljačkog sklopa cilj je da se osnovni harmonik struje dovede na traženu (referentnu) vrednost, dok se viši harmonici trebaju poništiti, pa su im odgovarajuće reference ravne nuli. Referenca struje na osnovnom harmoniku u načelu dolazi iz regulatora napona jednosmernog međukola, pošto se upravo nadzorom struje postiže stabilna vrednost napona aktivnog ispravljača. Međutim, pri izradi same strujne petlje, posmatramo ju nezavisno od ostatka sklopa i onda se referenca zadaje proizvoljno da bi se proverili odzivi. Pri tom je proporcionalno dejstvo potrebno za ubrzanje prelaznog procesa, dok je integralno dejstvo potrebno da bi se greška u ustaljenom stanju svela na nulu. Rezonantna učestanost integralnog dejstva se bira tako da izdvoji harmonik koji se želi regulisati. Na slici 4 je prikazan upravljački sklop osnovnog harmonika koji sledi neposredno iz jednačine (6).



Slika 4. P-Rezonantni regulator osnovnog harmonika

Parametre regulatora možemo proračunati polazeći od činjenice da oni ostaju potpuno isti kao u dq sistemu. Zato na osnovu (6), pretpostavljajući napon mreže $\hat{u}_{R\alpha\beta}$ kao poremećaj, imamo regulacionu strukturu sa slike 5.



Slika 5. Sklop za proračun parametara regulatora

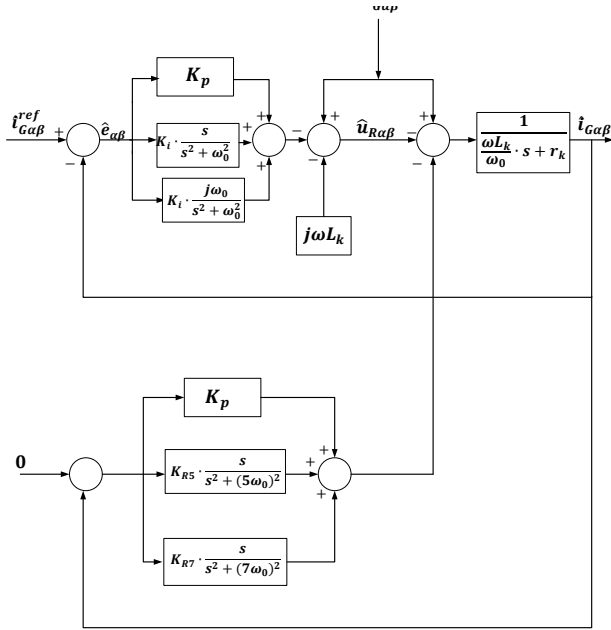
Parametre K_p i K_i biramo tako da se poništi pol koji unosi prenosna funkcija procesa, tj. Električne mreže, odnosno, kako je već rečeno: $\frac{K_p}{K_i} = \frac{L}{R}$. U povratnoj sprezi, prenosna funkcija je onda:

$$W(s) = \frac{1}{1 + s\tau_i} \quad (8)$$

gde je $\tau_i = \frac{R}{K_i}$. Konstantu τ_i biramo tako da bude red veličine sporija od učestanosti PWM-a, što je dovoljno brzo, a s druge strane ne dopušta prolazak šuma usled visokih harmonika PWM-a.

3.1. Suzbijanje viših harmonika

Cilj regulacije aktivnog ispravljača je da postigne sinusoidalnu struju koju on povlači iz mreže. U uslovima izobličenog napona, to znači da upravljački sklop mora aktivno suzbijati više harmonike mreže. To znači da moramo proširiti sklop sa slike 4 sa blokovima koji regulišu struje viših harmonika na vrednost 0.



Slika 6. Regulacija struje u uslovima izobličenog mrežnog napona

Sa slike 6 možemo uočiti da parametri regulatora K_p i K_i ostaju isti za više harmonike, peti i sedmi kao i za osnovni. Razlika je u referenci. Ovo je jasno iz strukture regulatora (6) kao i činjenice da se složena izobličena struja dobije kao zbir pojedinih harmonika. Jasno je dakle da su reference struje za 5-i i 7-mi harmonik ravne nuli, kako bi se oni suzbili. Referencu za osnovni harmonik ćemo, međutim dobiti iz nadređene petlje koja reguliše vrednost napona jednosmernog kola, pošto je zadatak ispravljača da obezbedi stabilan jednosmeran napon.

3.2. Nadzor napona jednosmernog kola

Jednačine koje opisuju jednosmerno kolo se mogu napisati na osnovu slike 3.

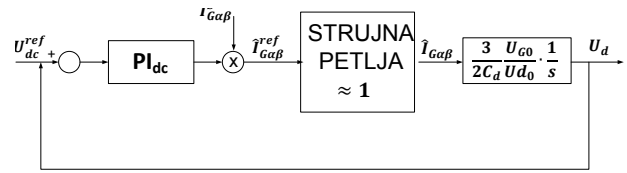
$$C \frac{du_d}{dt} = i_{d1} - i_{d2} \quad (9)$$

$$P_G = \frac{3}{2} \cdot (U_{Gd} \cdot i_{Gd}) = P_d = u_d \cdot i_{d1} \quad (10)$$

Iz (10) izražavamo struju i nakon linearizacije imamo:

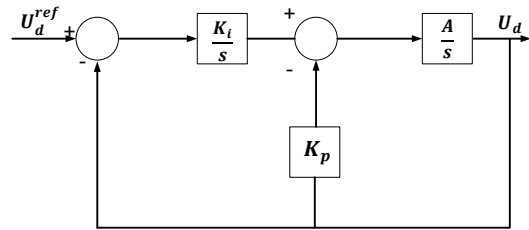
$$i_{d1} = \frac{3}{2} \frac{U_{Gd0}}{U_{d0}} i_{Gd} \quad (11)$$

Gornje jednačine su date u obrtnoj dq oblasti, pošto je bio cilj odrediti statičko pojačanje petlje, pa je otuda potpuno svejedno koji koordinatni sistem koristimo. Struju tereta i_{d2} smatramo za poremećaj, analogno mehaničkom momentu pri regulaciji brzine elektromotornog pogona. Šema regulacije napona jednosmernog kola data na slici 7.



Slika 7. Nadzor napona jednosmernog kola

Pri projektovanju regulatora napona jednosmernog kola, smatraćemo da je dinamika strujne petlje mnogo brža od dinamike napona, pa ćemo usvojiti pojačanje ove petlje ravne jedinici. Usvojicemo varijantu PI regulatora za regulaciju napona – tzv. IP regulator kojim se izbegava nula u funkciji spregnutog prenosa (slika 8)



Slika 8. PI regulator napona jednosmernog kola

Funkcija spregnutog prenosa je data izrazom:

$$\frac{U_d}{U_d^{ref}} = \frac{AK_i}{s^2 + AK_p s + AK_i} \quad (12)$$

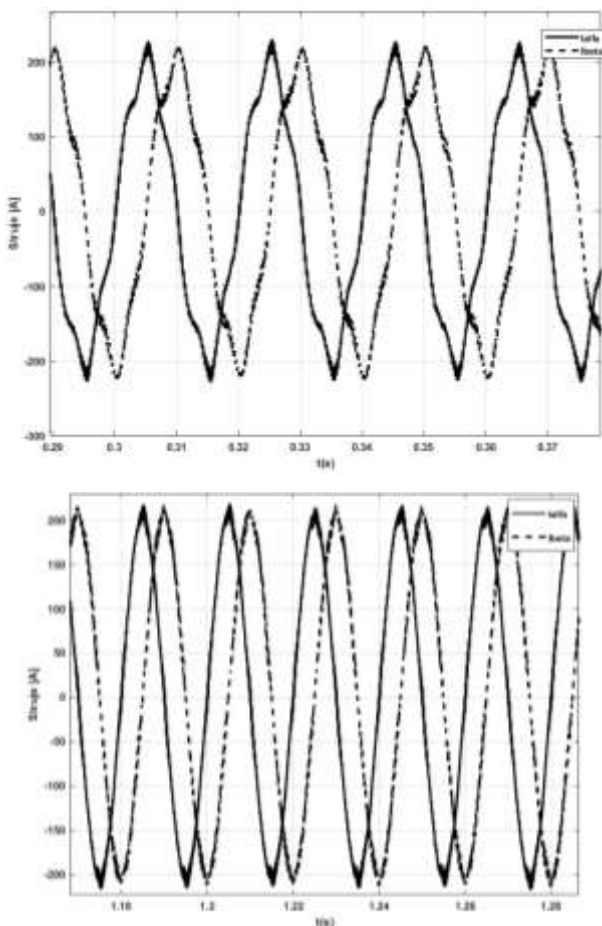
Gde je A statičko pojačanje, tj. $A = \frac{3}{2C_d} \frac{U_{G0}}{U_{d0}}$.

Vrednosti parametara K_p i K_i se dobiju na osnovu poređenja (11) i dobro poznate jednačine drugog reda $s^2 + 2\xi\omega_n s + \omega_n^2$ odakle je: $K_p = \frac{2\xi\omega_n}{A}$, $K_i = \frac{\omega_n^2}{A}$.

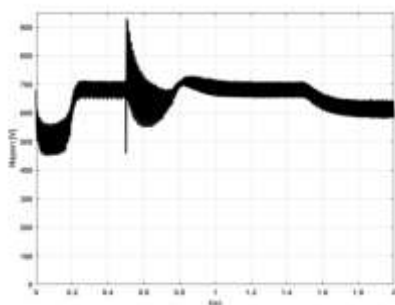
Prigušenjeξbiramo obično da bude $1/\sqrt{2}$ ili 1, dok prirodnu učestanost ω_n biramo na osnovu vremena smirenja [5]: $T_{set} = \frac{4}{\xi\omega_n}$. Na ovaj način su parametri regulatora jednosmernog kola u potpunosti određeni.

4. REZULTATI SIMULACIJA

Na slici 9 su dati rezultati simulacije odziva struje koju aktivni ispravljač povlači iz mreže pre i posle uključenja kompenzacije petog harmonika koji je prisutan u naponu mreže. Može se uočiti da su struje sinusoidalne, kako je to postavljeno u uslovima projektovanja aktivnog ispravljača.



Slika 9. Struje aktivnog ispravljača po α i β osi: pre (gore) i posle (dole) uključenja kompenzacije petog harmonika



Slika 10. Odziv napona jednosmernog kola

Na slici 10 je prikazan napon DC kola. Primećuje se prelazni proces, koji počinje u trenutku 0,5 sekundi, kada se uključi kontrola suzbijanja petog harmonika struje, a drugi prelazni proces nastupa u trenutku 1,5s, kada je smanjena referenca napona DC kola za 10%. U oba slučaja odziv napona jednosmernog kola zadovoljava uslov stabilnog napajanja potrošača sa odgovarajućom dinamikom.

5. ZAKLJUČAK

Ovaj rad je imao za cilj da analizira i predloži upravljački aktivnog filtra, pri čemu se pošlo od izrade regulatora u mirujućem koordinatnom sistemu. Pokazalo se da je ovaj pristup naročito pogodan u slučaju kada kruta mreža sadrži više harmonike, jer je upravljački sklop lako proširiv i daje veoma dobre rezultate, pri čemu nije potreban PLL. Zato je usvojena upravljačka šema koja se zasniva na tzv. P-Rezonantnim regulatorima. Glavne prednosti su jednostavna proširivost na proizvoljan broj viših harmonika, kao i jednostavan proračun parametara regulatora (proporcionalnog i integralnog dejstva) koji su potpuno isti za sve harmonike i jednaki odgovarajućim parametrima u obrtnoj, dq oblasti. Iz ovoga sledi i jednostavno projektovanje regulatora napona jednosmernog kola u cilju obezbeđivanja stabilnog napona aktivnog ispravljača. Rezultati simulacije su potvrdili da projektovani sklop aktivnog ispravljača povlači iz mreže sinusoidalne struje čak i pri izobličenom mrežnom naponu, pri čemu obezbeđuje brz odziv napona jednosmernog kola. Dalji rad bi mogao biti usmeren na primenu upravljačkog sklopa u slučaju aktivnih filtara.

6. LITERATURA

- [1] T. S. Key, J. S. Lai, "Comparison of Standards and Power Supply Design Options for Limiting Harmonic Distortion in Power Systems", IEEE Transactions on Industry Applications, vol. 29, no. 4, pp. 688-695, 1993.
- [2] J.W.Kolar, H.Ertl, "Status of the techniques of three-phase rectifier systems with low effects on the mains", The 21st International Telecommunication Energy Conference 9-9 June 1999, INTELEC '99, DOI: 10.1109/INTLEC.1999.794075
- [3] D.N.Zmood, "A Systematic Development of Improved Linear Regulators for Sinusoidal Power Converters", PhD thesis, Monash University, Victoria, Australia, June 2002.
- [4] A. Yazdani, R.Iravani, "Voltage-Sourced Converters in Power Systems", John Wiley & Sons, Inc. 2010, ISBN: 978-0-470-52156-4
- [5] L.Wang, et al, "PID and Predictive Control of Electrical Drives and Power Converters using Matlab Simulink" John Wiley & Sons Singapore Pte. Ltd. 2015, ISBN: 9781118339442.

Kratka biografija:



Vladimir Popov rođen je u Kikindi 1992. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranio je 2018.god.

kontakt: popovvl92@gmail.com



Marko Vekić – docent na Fakultetu tehničkih nauka u Novom Sadu, na Katedri za Energetsku elektroniku i pretvarace. Oblast interesovanja su mu energetska elektronika u prenosnim i distributivnim mrežama i kvalitet električne energije.

UPRAVLJANJE STATCOM-OM U SVRHU NADZORA NAPONA SABIRNICE U KOJOJ JE PRIKLJUČEN

STATCOM CONTROL FOR PCC VOLTAGE REGULATION

Jovo Vučetić, Marko Vekić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Ovaj rad se bavi postepenim razvojem sklopa STATCOM-a sa zadatkom regulacije napona u sabirnici u koju se STATCOM priključuje. Opisani su podsklopovi PLL-a, strujne petlje, regulacija napona DC kola i konačno nadzor napona STATCOM-a. Konačno je data i statička analiza, kao i rezultati simulacije.

Ključne reči: pretvarač na električnoj mreži, regulacija napona, STATCOM

Abstract – This paper describes the development of the STATCOM with the task of PCC voltage regulation. Sub-systems of PLL, current control, DC bus voltage control and PCC regulation are presented. Finally, system static analysis and simulation results are given.

Keywords: grid-connected converter, STATCOM, voltage regulation

1. UVOD

Ključna prednost naizmeničnih sistema prenosa i distribucije u odnosu na jednosmerne jeste smanjivanje gubitaka usled mogućnosti jednostavne transformacije naponskih nivoa. Međutim, naizmenični sistemi ipak imaju i nekoliko nedostataka [1] kao što su uravnotežavanje tokova snage u cilju očuvanja stabilnog napona u učestanosti. Ovi izazovi su se ranije rešavali kondenzatorima (kondenzatorskim baterijama) za kompenzaciju reaktivne snage, ali takva rešenja su bila veoma nefleksibilna. Fleksibilnije rešenje predstavljaju tzv. sinhroni kompenzatori, no oni su skupi, gabaritni i imaju srazmerno spor odziv. Međutim, sveobuhvatan napredak je postignut uključanjem komponenti i uređaja energetske elektronike [2] u prenosnu i distributivnu mrežu čime je nastala čitava skupina uređaja [3] pod imenom FACTS. Tako je energetska elektronika omogućila i veoma delotvornu zamenu složenog uređaja kakav je sinhroni kompenzator sa statičkim sinhronim kompenzatorom (eng: STATCOM). Dejstvo STATCOM-a [4], [5] je istovetno u smislu finog podešavanja utiskivanja/trošenja jalove snage, ali sa daleko boljom dinamikom i smanjenim gabaritima uređaja.

U ovom radu će biti predstavljen potanko rad pretvarača na mreži sa upravljačkim algoritmom STATCOM-a za održavanje napona u sabirnici priključenja.

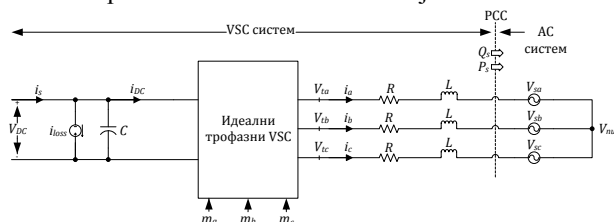
NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Marko Vekić, docent.

Odgovarajući upravljački sklop se sastoji od podsklopa za sinhronizaciju pretvarača s mrežom (PLL), strujne petlje, petlje napona jednosmernog kola i naponske petlje (nadzor napona u priključnoj sabirnici).

2. PRETVARAČ POVEZAN NA MREŽU

Pretvarač povezan na električnu mrežu je dat na slici 1.



Slika 1: Pretvarač na električnoj mreži

Jednačina koja opisuje stranu pretvarača ka mreži zapisana u fazorskom obliku u mirujućoj $\alpha\beta$ oblasti:

$$\hat{V}_{t\alpha\beta} = L \frac{d\hat{i}_{\alpha\beta}}{dt} + R\hat{i}_{\alpha\beta} + \hat{V}_{s\alpha\beta} \quad (1)$$

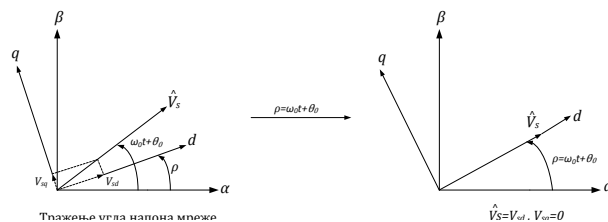
Nakon tzv dq transformacije:

$$f_d + jf_q = \hat{f}(t)e^{-j\rho(t)} = (f_\alpha + jf_\beta)e^{-j\rho(t)} \quad (2)$$

gde je ρ ugao dq sistema, daljeimamo:

$$L \frac{d}{dt}(i_{dq}) = -j \left(L \frac{d\rho}{dt} \right) i_{dq} - R i_{dq} + V_{tdq} - \hat{V}_s e^{j(\omega_0 t + \theta_0 - \rho)} \quad (3)$$

Cilj je da se ugao ρ izjednači s uglom fazora $\omega_0 t + \theta_0$ kako bi kontrolisane veličine imale jednosmerne vrednosti pogodne za PI regulatore.

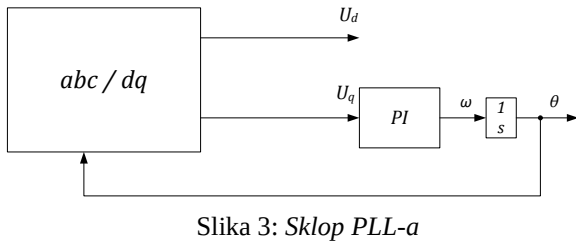


Slika 2: Dejstvo PLL-a

U tu svrhu koristimo sklop PLL-a (slika 2).

2.1. PLL

PLL je suštinski nadzorni sklop (slika 3) koji reguliše q komponentu napona na vrednost 0, time izjednačavajući ugao ρ sa uglom fazora $\omega_0 t + \theta_0$.



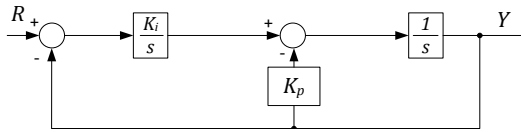
Slika 3: Sklop PLL-a

Jednačina (3) nakon toga, raspregnuta po d i q osi glasi:

$$L \frac{di_d}{dt} = \omega_0 L i_q - R i_d + V_{td} - V_s \quad (4)$$

$$L \frac{di_q}{dt} = -\omega_0 L i_d - R i_q + V_{tq} \quad (5)$$

Za proračun parametara PI regulatora koristimo šemu sa slike 4.



Slika 4: IP sklop regulatora

Ovo je zapravo u nekoliko izmenjen uobičajen sklop gde je proporcionalno dejstvo izmešteno u povratnu granu da bi se izbegla nula u funkciji povratnog prenosa, koja glasi:

$$\frac{Y}{R} = \frac{k_i}{s^2 + k_p s + k_i} = \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} \quad (6)$$

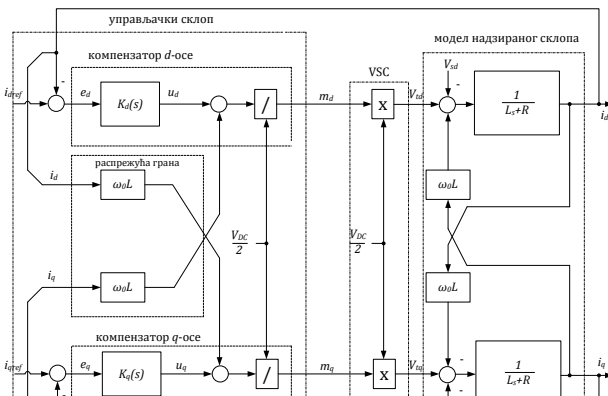
Izbor k_p i k_i , vršimo na osnovu ξ i ω_n , tako što prvo izaberemo prigušenje $\xi = 0.707$, a ω_n vodeći računa o vremenu smirenja: $t_s = \frac{4.5\xi}{\omega_n}$.

2.2. Strujna petlja

U projektovanju strujne petlje, dinamiku koju određuje mrežna induktivnost i otpornost treba zameniti PI regulatorom. Deo tog procesa je i tzv. raspregzanje po osama gde se mrežni napon i dejstvo struje po suprotnoj osi smatraju poremećajima:

$$\begin{aligned} u_d &= V_{td} - V_{sd} + \omega_0 L i_q \\ u_q &= V_{tq} - V_{sq} + \omega_0 L i_d \end{aligned} \quad (7)$$

Nadzorni sklop strujne petlje je dat na slici 4.

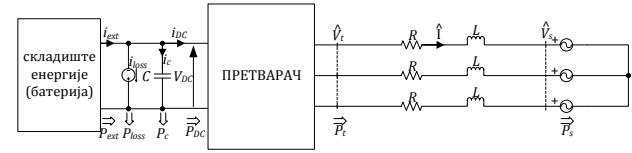


Slika 5: Nadzorni sklop strujne petlje

Pri projektovanju PI regulatora [5], primenjeno je načelo poništavanja dominantnog pola L/R kojeg unosi električna mreža, nulom PI regulatora, pa su parametri: $k_p = L/\tau_i$ i $k_i = R/\tau_i$.

2.3. Nadzor napona DC kola

Cilj nadzora jednosmernog međukola je održanje napona na stalnoj vrednosti da bi se postigao neometan tok snage (slika 6).



Slika 6: Tok snage pretvarača na mreži

Ravnoteža snaga je data izrazom:

$$P_{ext} = P_{loss} + P_c + P_{DC} \quad (8)$$

Zamenjujući $P_{loss} = V_{DC} \cdot i_{loss}$, $P_c = \frac{dE_c}{dt} = \frac{d}{dt} \left(C \cdot \frac{V_{DC}^2}{2} \right)$, $P_{DC} = V_{DC} \cdot i_{DC}$ kao i zanemarujući gubitke samog pretvarača, tj. $P_t = P_{DC}$, a zatim vršeći linearizaciju, dobija se:

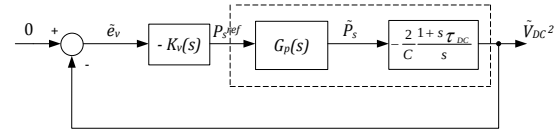
$$s \cdot \tilde{V}_{DC}^2 = \frac{2}{C} \cdot \tilde{P}_{ext} - \frac{2}{C} \left[\tilde{P}_s + \frac{2LP_{ext0}}{3V_s^2} \cdot s\tilde{P}_s + \frac{2LQ_{s0}}{3V_s^2} \cdot s\tilde{Q}_s \right] \quad (9)$$

Odakle se dobija prenosna funkcija nadzora jednosmernog kola:

$$G_{V_{DC}} = \frac{\tilde{V}_{DC}^2}{\tilde{P}_s} = -\frac{2}{C} \cdot \frac{1 + s\tau_{DC}}{s} \quad (10)$$

gde je $\tau_{DC} = \frac{2LP_{ext0}}{3V_s^2}$.

Nadzorna petlja je data na slici 7.



Slika 7: Nadzorni sklop napona DC kola

Blok $G_p(s)$ sa gornje slike, iako predstavlja funkciju prenosa snage, ona je ravna prenosnoj funkciji strujne petlje, pošto je:

$$P_s(t) = \frac{3}{2} V_{sd}(t) i_d(t) \text{ i } Q_s(t) = -\frac{3}{2} V_{sd}(t) i_q(t),$$

jer je $V_{sq} = 0$ u ustaljenom stanju PLL-a (slika 2). Dakle,

$$G_p(s) = G_i(s) = \frac{1}{1 + s\tau_i}, \text{ gde je } \tau_i = L/R.$$

Regulator, odnosno kompenzator koji nadzire napon jednosmernog kola k_v ćemo pretpostaviti da ima oblik:

$$k_v(s) = \frac{C}{2} \cdot \frac{H(s)}{s} \quad (11)$$

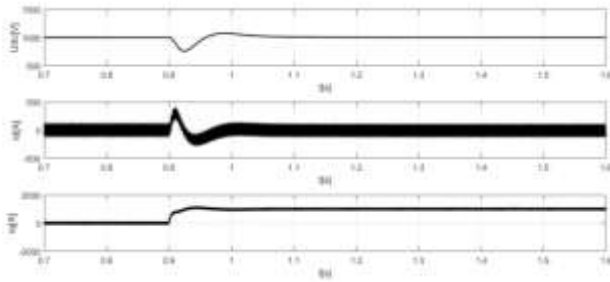
gde smo koeficijent $C/2$ usvojili zbog normalizacije (10), a integrator zbog nulte greške ustaljenog stanja. Preostalu nepoznatu prenosnu funkciju određujemo na osnovu željenog preteka faze. Analizom Bode-ovog dijagrama, dobija se:

$$H(s) = \frac{(\omega_c^{DC})^2}{s} \cdot \left(\frac{s+p}{s+\alpha} \right),$$

gde α dobijemo iz: $\delta_m = \arcsin \frac{\alpha-1}{\alpha+1}$,

dok je δ_m željeni pretek.

Na slici 8 su dati rezultati simulacije odziva napona DC kola kada se referenca struje po q osi (reaktivna komponenta) skokovito poveća u trenutku 1s sa 0 na 1000 A. Sa slike se vidi da se nakon prelaznog procesa kraćeg od pola sekunde, napon vraća na referentnu vrednost od 1000 V.



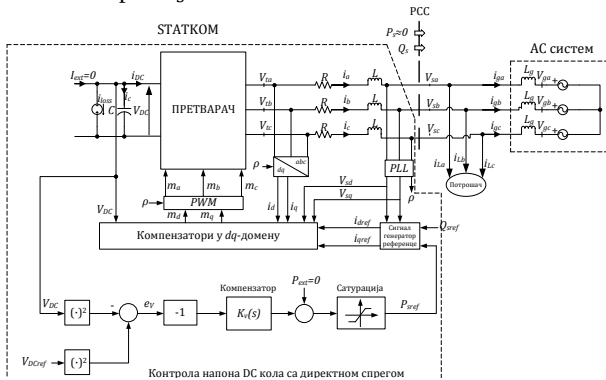
Slika 8: Rezultati simulacije odziva napona DC kola

3. NADZOR STATCOM-a

Jednačine koje opisuju napon $\hat{V}_s$ i struje u sabirnici priključenja (PCC) STATCOM-a u fazorskom obliku su:

$$\hat{V}_s = \hat{V}_g + L_g \frac{d\hat{i}_g}{dt}, \hat{i}_g = \hat{i} - \hat{i}_L \quad (12)$$

U istu sabirnicu (slika 9) priključen je teret (i_L) i kruta mreža (i_g). Zadatak STATCOM-a je da utiskuje reaktivnu struju u cilju kompenzacije pada napona na induktivnosti mreže koju uzrokuje struja tereta i na taj način održi stabilnim napon $\hat{V}_s$.



Slika 8: Nadzorni sklop STATCOM-a

Prelaskom u obrtnu (dq) oblast uz pretpostavku da nije nastupilo ustaljen stanje PLL-a, (12) postaje:

$$\begin{aligned} V_{sd} &= L_g \frac{di_{gd}}{dt} - \omega L_g i_{gq} + V_g \cos(\omega_0 t + \theta_0 - \rho) \\ V_{sq} &= L_g \frac{di_{gq}}{dt} + \omega L_g i_{gd} + V_g \sin(\omega_0 t + \theta_0 - \rho) \end{aligned} \quad (13)$$

Gornja jednačina je nelinearna pošto sadrži proizvod dve promenljive, struje i kružne učestanosti. Stoga vršimo linearizaciju na sledeći način:

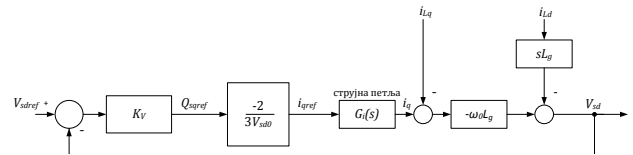
$$\begin{aligned} V_{sd} &= V_{sd0} + \tilde{V}_{sd}, \quad V_{sq} = 0 + \tilde{V}_{sq}, \quad i_{gd} = i_{gd0} + \tilde{i}_{gd}, \\ i_{gq} &= i_{gq0} + \tilde{i}_{gq}, \quad \frac{d\rho}{dt} = \omega_0 + \frac{d\tilde{\rho}}{dt} \end{aligned}$$

Razvijajući dalje (13) sa gornjim izrazima, dobija se:

$$\tilde{V}_{sd}(s) = \underbrace{-G_d(s)\tilde{I}_{Ld}(s) - G_q(s)\tilde{I}_{Lq}(s)}_{\text{efekat potrošača}} + \underbrace{G_q(s)\tilde{I}_q(s)}_{\text{efekat nadzora}} \quad (14)$$

gde su $G_d(s) = sL_g iG_q(s) = -\omega_0 L_g$.

U (14) se može uočiti da dve glavne komponente utiču na napon sabirnice: potrošač i nadzorni sklop STATCOM-a. Na slici 9 je prikazan nadzorni sklop utemeljen na (14).



Slika 9: Nadzor napona sabirnice

Prenosna funkcija traženog regulatora napona sabirnice data je blokom $k_v(s)$, pa je funkcija povratnog prenosa:

$$G_v(s) = k_v(s) \cdot \left(-\frac{2}{3V_{sd0}} \right) \cdot G_i(s) \cdot (-\omega_0 L_g) \quad (15)$$

Ako izaberemo presečnu učestanost petlje napona mnogo manju od presečne učestanosti strujne petlje u cilju njihovog raspredanja, npr. $\omega_c^{AC} = \frac{\omega_i}{15}$, onda je: $G_v(j\omega_c^{AC}) = k_v(j\omega_c^{AC}) \cdot \frac{2\omega_0 L_g}{3V_{sd0}} \cdot G_i(j\omega_c^{AC})$ i pošto je: $G_i(j\omega_c^{AC}) \approx 1$ zbog $\omega_c^{AC} \ll \omega_i$, onda (15) u oblasti učestanosti postaje:

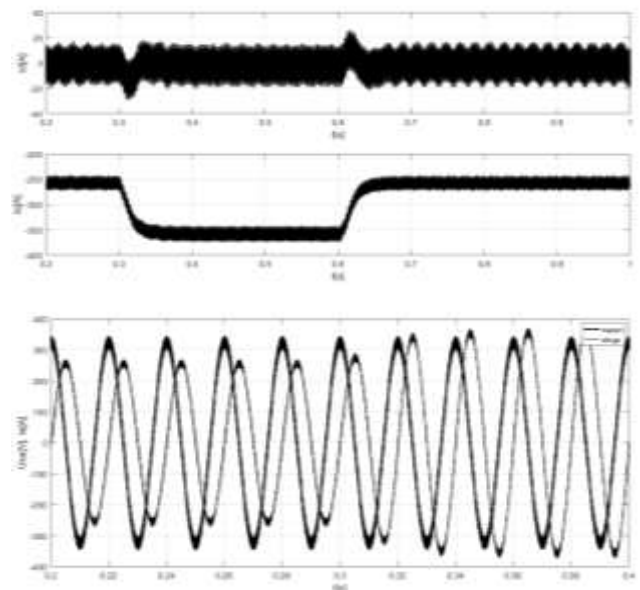
$$G_v(j\omega_c^{AC}) \approx \frac{2\omega_0 L_g}{3V_{sd0}} \cdot k_v(j\omega_c^{AC}) \quad (16)$$

Slično kao u (11), pretpostavimo da je: $k_v(s) = \frac{A}{s}$, pa je (16) dalje: $G_v(j\omega_c^{AC}) = \frac{2\omega_0 L_g}{3V_{sd0}} \cdot \frac{A}{j\omega_c^{AC}}$. Pošto tražimo da pojačanje nadzorne petlje bude 1 u ω_c^{AC} , onda iz gornje jednačine sledi vrednost koeficijenta

$$A \cdot \frac{2\omega_0 L_g}{3V_{sd0}} \cdot \frac{A}{\omega_c^{AC}} = 1 \Rightarrow A = \frac{\omega_c^{AC}}{\frac{2\omega_0 L_g}{3V_{sd0}}}$$

4. REZULTATI SIMULACIJA

Na slici su prikazani rezultati simulacija struja i_d i i_q , napona mreže U_{sa} i struje mreže i_a u slučaju povećavanja opterećenja (dodavanja tereta). U trenutku promene tereta potrošač povuče veću struju. Kao posledica toga poveća se struja mreže koja bi onda napravila pad napona na svojoj induktivnosti X_g . Da bi se to sprečilo STATCOM utiskuje reaktivnu komponentu struje i jedan deo te struje se grana prema mreži i kompenzuje pad napona na njenoj induktivnosti.

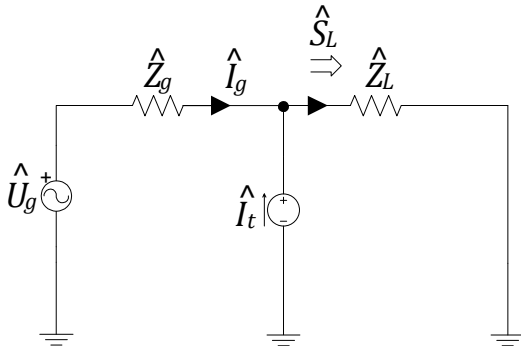


Slika 10: Nadzor napona sabirnice

5. STATIČKA ANALIZA

Postavlja se pitanje koje su mogućnosti STATCOM-a da kompenzuje pad napona razmenjujući samo reaktivnu snagu s mrežom (zanemarujući aktivne gubitke). Odgovor na to nam daje statička analiza rada koja utvrđuje odnos između strujnog i naponskog nivoa STATCOM-a, ztj. Njegovog kapaciteta i snage tereta u datoj sabirnici (slika 11). Pretpostavimo da je napon sabirnice (PCC) referentni, tj. $\hat{U} = Ue^{j0} = U$.

Pošto STATCOM razmenjuje samo reaktivnu struju, onda je: $\hat{I}_t = jI_t$. Neka je dalje prividna snaga tereta po fazi: $\hat{S}_L = S_L e^{-j\varphi} = \hat{U}^* \cdot \hat{I}_L = U \cdot \hat{I}_L$



Slika 11: Statička analiza STATCOM-a

Struja tereta je onda: $\hat{I}_L = \frac{\hat{S}_L}{U}$. Na osnovu gornje slike i Kirchhoff-ovih zakona dalje imamo: $\hat{U}_g - Z_g e^{j\theta} \cdot \hat{I}_g = \hat{U}$, $\hat{U}_g = U e^{j\delta}$ i $\hat{I}_g + \hat{I}_t = \hat{I}_L$. Izrazimo struju $\hat{I}_g$, pa imamo:

$\frac{U}{Z_g} \cdot \frac{e^{j\delta}-1}{e^{j\theta}} + jI_t = \frac{\hat{S}_L}{U}$ odakle sledi da je snaga $\hat{S}_L$:

$$\hat{S}_L = \frac{U^2}{Z_g} \cdot (e^{j(\delta-\theta)} - e^{-j\theta}) + jUI_t = P_L - jQ_L \quad (17)$$

Primenjujući Euler-ov obrazac na (15), možemo izraziti:

$$\cos(\delta - \theta) = \frac{Z_g}{U^2} P_L + \cos \theta, \text{ odnosno: } \sin(\delta - \theta) = \sqrt{1 - \left(\frac{Z_g}{U^2} P_L + \cos \theta \right)^2}.$$

Konačno, izražavajući struju STATCOM-a I_t imamo:

$$I_t = -\frac{Q_L}{U} - \frac{U}{Z_g} \left[\sin \theta + \sqrt{1 - \left(\frac{Z_g}{U^2} P_L + \cos \theta \right)^2} \right] \quad (18)$$

Gornja jednačina zapravo određuje strujni nivo STATCOM-a. Ukupna prividna snaga (kapacitet kompenzacije) je proizvod gornjeg izraza i napona sabirnice.

6. ZAKLJUČAK

Ovaj rad je imao za cilj da prikaže razvoj upravljačkog sklopa D-STATCOM-a. Pošlo se od analize i sagledavanja elemenata koji su svojstveni radu pretvarača energetske elektronike povezanog na električnu mrežu. Prvo je projektovan podsklop za sinhronizaciju pretvarača na mrežu – PLL. Pri analizi i izradi strujne petlje pošlo se od pretpostavke da je aktivna komponenta struje (q komponenta) srazmerna jalovoj snazi, a aktivna (d komponenta) radnoj snazi.

Podsklop nadzora napona jednosmernog kola je uspostavljen na osnovu jednačine ravnoteže snaga. Nakon što su uspostavljeni osnovni podsklopovi pretvarača povezanog na mrežu, prešlo se na primenu datih podsklopova za rad D-STATCOM-a u svrhu nadzora napona pretvarača u sabirnici priključenja. Dobijeni model je nelinearan, pa je nadzorni sklop projektovan u oblasti učestanosti nakon izvršene linearizacije. Rezultati simulacija su potvrdili da napon u sabirnici priključenja nakon kraćeg prelaznog procesa usled poremećaja izazvanog promenom tereta ostaje stabilan na željenoj vrednosti. Budući da izabrani D-STATCOM ne poseduje spoljne skladište energije, tj. nadzor napona vrši samo tokom reaktivne snage, bilo je neophodno izvršiti i statičku analizu rada da bi se sagledala ograničenja, odnosno neophodne strujne i naponske mogućnosti pretvarača. Statička analiza je dala zavisnost ukupne prividne snage D-STATCOM-a (ukupnog kapaciteta) u funkciji aktivne i reaktivne snage potrošača priključenog u posmatranu sabirnicu. Dalji pravci istraživanja bi mogli biti prošireni na STATCOM sa energetske skladištem u cilju povećanja mogućnosti kompenzacije.

7. LITERATURA

- [1] E. Acha, V.G. Agelidis, "Power Electronic Control in Electrical Systems", Newnes 2002, ISBN 0750651261
- [2] Edited by Y. H.Song and A. T. Johns, "Flexible AC Transmission Systems (FACTS)", The Institution of Engineering and Technology, 2008, ISBN: 978-0-85296-771-3
- [3] F. Shahnia, S. Rajakaruna, A. Ghosh, "Static Compensators (STATCOMs) in Power Systems", Springer 2015, ISBN: 978-981-287-281-4
- [4] A. Yazdani, R.Iravani, "Voltage-Sourced Converters in Power Systems", a John Wiley & Sons, Inc. 2010, ISBN: 978-0-470-52156-4
- [5] M.C. Falvo, L. Martirano, D. Sbordone, "D-STATCOM with energy storage system for application in Smart Micro-Grids", 2013 International Conference on Clean Electrical Power (ICCEP), 29 August 2013, ISBN: 978-1-4673-4429-6
- [6] L. Wang, et al., "PID and Predictive Control of Electrical Drives and Power Converters using Matlab Simulink" John Wiley & Sons Singapore Pte. Ltd. 2015, ISBN: 9781118339442.

Kratka biografija:



Jovo Vučetić rođen je u Bijeljini 1988. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranio je 2018.god.

kontakt: vuceticjovo@gmail.com



Marko Vekić je docent na Fakultetu tehničkih nauka u Novom Sadu, na Katedri za Energetsku elektroniku i pretvarace. Oblast interesovanja su energetska elektronika u prenosnim i distributivnim mrežama i kvalitet električne energije.

RAZVOJ NAMENSKOG JEZIKA ZA SPECIFIKACIJU MOBILNIH APLIKACIJA ZA DOMEN PRODAJE ODEĆE**A DOMAIN-SPECIFIC LANGUAGE FOR THE SPECIFICATION OF MOBILE APPLICATIONS IN THE DOMAIN OF CLOTHING SALES**

Nada Ignjić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu opisan je razvijeni jezik pomoću kojeg se kreira specifikacija aplikacija za mobilne telefone iz domena prodaje odeće. Na osnovu specifikacije generišu se prototipovi aplikacija u radnom okviru Ionic.

Ključne reči: *namenski jezici, razvoj softvera vođen modelima, aplikacije za mobilne telefone.*

Abstract – In this paper we describe a domain-specific language for creating specifications of mobile application prototypes in the domain of clothing retail. Based on these specifications, executable application prototypes are generated in the Ionic framework.

Key words: *Domain-Specific Language, Model-Driven Software Development, Mobile Applications.*

1 UVOD

Svedoci smo da su mobilni telefoni postali neizostavni deo naše svakodnevnice. Razlog za to je taj što mobilni uređaji više ne služe samo kao sredstvo za komunikaciju, već se koriste i kao sredstvo koje olakšava i štedi vreme prilikom obavljanja svakodnevnih zadataka. Zbog toga, prodaja mobilnih telefona u poslednjih nekoliko godina beleži rast. Dokaz za to je istraživanje prikazano u [1], koje pokazuje da je u 2017. godini prodato blizu 1.5 milijardi pametnih uređaja, što predstavlja rast od 2,47% u odnosu na 2016. godinu. U 2017. godini, 2.32 milijarde ljudi posedovalo je pametni mobilni uređaj [2].

Aplikacije koje se izvršavaju na mobilnim uređajima, postale su složenije ne samo u pogledu problema koji rešavaju, već i u smislu tehnologija koje su korišćene za njihovu izradu. Jedan od parametara koji prikazuje uspešnost mobilne aplikacije jeste i broj preuzimanja aplikacije u *online* prodavnicama. Da bi broj preuzimanja bio veliki, potrebno je napraviti aplikaciju koja je podržana na više različitih mobilnih platformi. Zbog specifičnosti svake platforme, aplikacija razvijena za određeni operativni sistem prirodno je nekompatibilna sa ostalim platformama. Ovaj nedostatak kompatibilnosti primorava programere da ponovo programiraju iste aplikacije za svaku ciljnu platformu povećavajući troškove i vreme isporuke aplikacije na tržište. Da bi se rešio ovaj nedostatak, kreiraju se hibridne mobilne aplikacije koje se programiraju u jednom programskom

jeziku nezavisnom od bilo koje platforme, koji se potom kompajlira u kôd koji je moguće izvršiti na željenoj platformi.

Jedan od primera najdostupnijih aplikacija u *online* prodavnicama su i mobilne aplikacije za prodaju odeće. Posmatrajući ove aplikacije, uočeni su zajednički koncepti koji su karakteristični za sve aplikacije iz ovog domena. Koristeći uočene koncepte, menadžeri i vlasnici radnji bi mogli da definišu izgled svoje aplikacije, ubrzao bi se ciklus pravljenja prototipa, i pojedini delovi razvoja aplikacije bi bili automatizovani generisanjem kôda.

Cilj rada je ustanovljenje pristupa koji koristi namenski jezik kako bi pomoću tipskih stranica specifikovali prototipove aplikacija. Iz te specifikacije moguće je generisati izvršive aplikacije za različite mobilne platforme. Ovakav pristup treba da omogući uštedu vremena prilikom razvoja prototipa aplikacije u odnosu na ručno pisanje kôda. U tradicionalnom razvoju neophodno je ručno pisati kôd, dok korišćenjem namenskog jezika generiše se određeni deo kôda na osnovu modela, prilikom čega se dobija na uštedi vremena.

Osim uvoda i zaključka, ovaj rad sadrži još četiri poglavlja. Poglavlje 2, pod nazivom *Arhitektura alata*, predstavlja opis delova razvijenog alata. *Pregled radnih okvira za razvoj hibridnih mobilnih aplikacija* predstavlja naziv poglavlja 3 u kojem je na osnovu rezultata evaluacije izabranih radnih okvira donesena odluka da se za potrebe ovog rada koristi *Ionic*. U poglavlju 4, *Namenski jezik za specifikaciju mobilnih aplikacija*, opisane su apstraktna i konkretna sintaksa razvijenog namenskog jezika. U poglavlju 5, *Primer generisanog prototipa aplikacije*, prikazane su pojedine stranice koje predstavljaju deo prototipa aplikacije koji je generisan uz pomoć generatora kôda.

2 ARHITEKTURA ALATA

U ovom poglavlju predstavljena je arhitektura razvijenog softverskog alata, kao i uloge svih komponenti koje čine gradivne elemente alata.

Za razvoj namenskog jezika korišćeno je okruženje EMF (*Eclipse Modeling Framework*) sa skupom pripadajućih alata. Prvi korak u razvoju namenskog jezika predstavlja kreiranje meta-modela. Kreiranje meta-modela predstavlja proces kreiranja konceptata i veza među konceptima budućeg jezika. Meta-model ujedno predstavlja i apstraktnu sintaksu namenskog jezika.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila dr Slavica Kordić, docent.

Na osnovu meta-modela, kreira se konkretna sintaksa. Razvojem konkretne sintakse, korisniku se pruža mogućnost specifikiranja modela koji je u skladu sa odgovarajućim meta-modelom, i koji zadovoljava ograničenja koja su specifikirana primenom jezika OCL (*Object Constraint Language*).

Za specifikaciju generatora korišćen je jezik *Xtend*, koji predstavlja proširenje programskog jezika *Java*, pomoću kojeg se specifikiraju pravila i ograničenja prilikom generisanja kôda. Generisanje kôda podrazumeva generisanje statičkog i dinamičkog kôda. Statički kôd ne zavisi od sadržaja jezičke skripte, dok dinamički kôd zavisi od definisanih koncepata i njihovih odnosa. Dinamički deo dobija se na osnovu strukture modela zadatog na ulazu u proces generisanja programskog kôda. Na ulazu u proces generisanja programskog kôda zadaje se specifikacija modela kreirana u skladu sa odgovarajućim meta-modelom, dok se na izlazu dobija izvršivi programski kod za ciljnu platformu.

3 PREGLED RADNIH OKVIRA ZA RAZVOJ HIBRIDNIH MOBILNIH APLIKACIJA

Na osnovu istraživanja koje je sprovedeno za potrebe ovog rada, izabran je *Ionic* kao hibridni radni okvir koji će biti korišćen za izradu statičkog kôda prilikom generisanja prototipa aplikacija. Evaluacija izabranih radnih okvira rađena je u odnosu na potrebe korisnika aplikacija i u odnosu na potrebe programera koji razvijaju aplikacije koristeći izabran radni okvir. Parametri za evaluaciju u odnosu na potrebe programera su:

- **Licenca i troškovi** - da li je potrebno platiti za korišćenje radnog okvira? Maksimalna ocena koja može biti dodeljena radnom okviru je 2.
- **Dugoročna upotrebljivost** - da li je to radni okvir koji se aktivno razvija i koji obećava mogućnost dugoročne upotrebe? Maksimalna ocena koja može biti dodeljena radnom okviru je 3.
- **Kvalitet dokumentacije** - neophodno je da bude jasna, dobro struktuirana i jednostavna za korišćenje. Maksimalna ocena koja može biti dodeljena radnom okviru je 3.
- **Lakoća učenja izabranog radnog okvira** - radni okvir koji je u saglasnosti sa standardima, lako se razume i bolje koristi. Maksimalna ocena koja može biti dodeljena radnom okviru je 3.
- **Razvojni napor** - napor koji mora uložiti programer kako bi za najkraće vreme završio softverski proizvod na kome radi uz korišćenje izabranog radnog okvira. Maksimalna ocena koja može biti dodeljena radnom okviru je 3.
- **Veličina zajednice** - zajednica ljudi koja koristi izabran radni okvir treba da bude što veća i aktivna kako bi programeri, ukoliko naiđu na neki problem, lakše došli do potencijalnog rešenja. Maksimalna ocena koja može biti dodeljena radnom okviru je 5.
- **Dostupnost na platformama** – radni okvir koji podržava više platformi obezbeđuje veću upotrebljivost. Maksimalna ocena koja može biti dodeljena radnom okviru je 2.

U tabeli 1 prikazani su rezultati evaluacije u odnosu na potrebe programera. Prva kolona sadrži nazive parametara po kojima se vrši evaluacija određenog radnog okvira, dok ostale kolone sadrže nazive radnih okvira koji se porede kao i ocene koje su im dodeljene. Maksimalna ukupna ocena koju može osvojiti radni okvir je 3. Ukupna ocena dobija se na taj način što se izračuna srednja vrednost svih ocena koje su dodeljene radnom okviru.

	<i>Ionic</i>	<i>React Native</i>	<i>Onsen UI</i>	<i>Framework7</i>	<i>Xamarin</i>
Licenca i troškovi	2	2	2	2	2
Dugoročna upotrebljivost	3	3	2	2	3
Kvalitet dokumentacije	3	3	2	3	3
Lakoća učenja	2	1	3	3	1
Razvojni napor	3	2	3	3	3
Veličina zajednice	5	3	1	1	5
Dostupnost na platformama	2	1	1	1	2
Ukupno:	2.85	2.14	2.00	2.14	2.71

Tabela 1. Rezultati evaluacije radnih okvira u odnosu na potrebe programera

Parametri za evaluaciju u odnosu na potrebe korisnika aplikacije su:

- **Kvalitet korisničkog iskustva** - parametar koji pokazuje da li realizovani korisnički interfejs odgovara platformi na kojoj se aplikacija pokreće. Maksimalna ocena koja može biti dodeljena radnom okviru je 2.
- **Performanse** – elementi korisničkog interfejsa i animacije moraju brzo reagovati na interakciju korisnika, kako bi performanse bile zadovoljavajuće. Maksimalna ocena koja može biti dodeljena radnom okviru je 2.

U tabeli 2 prikazani su rezultati evaluacije u odnosu na potrebe korisnika aplikacije.

	<i>Ionic</i>	<i>React Native</i>	<i>Onsen UI</i>	<i>Framework7</i>	<i>Xamarin</i>
Korisničko iskustvo	2	2	2	2	2
Performanse	1	2	1	1	2
Ukupno:	1.50	2.00	1.50	1.50	2.00

Tabela 2. Rezultati evaluacije radnih okvira u odnosu na potrebe korisnika aplikacije

Na osnovu tabela 1 i 2 kao najbolji radni okviri izdvajaju se *Ionic* i *Xamarin*. Najveća prednost ovih radnih okvira je dobra dokumentacija, kao i velika zajednica ljudi koju

ovi radni okviri okupljaju, stoga je rad sa njima izuzetno olakšan.

Međutim, zbog jasne i dobro objašnjene dokumentacije, velike zajednice ljudi, kao i lakoće učenja novog radnog okvira, odlučeno je da će se za potrebe ovog projekta koristiti *Ionic*.

4 NAMENSKI JEZIK ZA SPECIFIKACIJU MOBILNIH APLIKACIJA

Specifikacija namenskog jezika podrazumeva razvoj apstraktne i konkretne sintakse jezika.

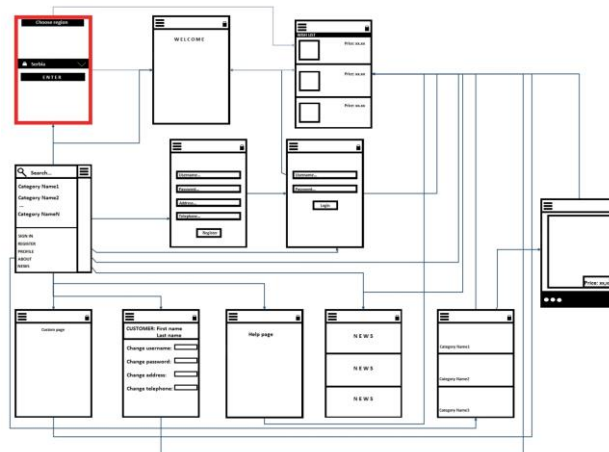
Apstraktna sintaksa predstavlja meta-model jezika, u kojem su predstavljeni koncepti i veze između koncepata. U nastavku su navedeni koncepti meta-modela jezika, kao i njihova objašnjenja:

- **App** - odgovara aplikaciji koja se kreira i predstavlja korenski koncept,
- **Menu** - odgovara meniju koji se pojavljuje u aplikaciji u vidu padajućeg tastera koji prikazuje listu linkova ka različitim stranicama,
- **Link** - odgovara vezi koja povezuje meni sa stranicama koje su kreirane, ili omogućuje prelaz sa jedne stranice na drugu,
- **Pages** - odgovara stranicama koje se kreiraju u okviru aplikacije,
- **EntryPoint** - odgovara stranici koja će biti prikazana kao početna stranica prilikom pokretanja aplikacije,
- **Region** - odgovara stranici koja se koristi za odabir regiona,
- **Login** - odgovara stranici koja se koristi za prijavu na sistem,
- **FirstPage** - odgovara stranici koja predstavlja kratku prezentaciju istaknutih ponuda ili viziju same kompanije za čije potrebe se kreira prototip aplikacije,
- **Category** - odgovara stranici koja sadrži različite kategorije odeće koje nudi robna kuća,
- **Register** - odgovara stranici koja služi za registrovanje korisnika koji se prvi put prijavljuju na sistem,
- **NonEntryPoint** - odgovara stranici koja ne može biti izabrana kao stranica koja će biti prva prikazana prilikom pokretanja aplikacije,
- **ShoppingCart** - odgovara stranici koja prikazuje detalje o korisničkoj korpi,
- **DetailPage** - odgovara stranici koja prikazuje detalje o nekom proizvodu,
- **HelpPage** - odgovara stranici koja prikazuje pomoć korisnicima aplikacije,
- **ProfileBuyer** - odgovara stranici koja prikazuje informacije o korisniku koji je ulogovan na sistem,
- **News** - odgovara stranici koja prikazuje novosti o trenutnim dešavanjima ili ponudama u svetu mode,
- **CustomPage** - odgovara opštoj stranici koja se modifikuje u zavisnosti od potreba vlasnika aplikacije za neki dopunski sadržaj (obaveštenja), i

- **About** - odgovara stranici koja prikazuje osnovne informacije o robnoj kući za čije se potrebe kreira prototip aplikacije

Razvojem konkretne sintakse, korisniku se pruža mogućnost specificiranja modela koji je u skladu sa odgovarajućim meta-modelom. Postoje dve vrste konkretne sintakse, prva koja se zasniva na upotrebi tekstualne notacije, dok je druga zasnovan na upotrebi grafičke notacije. U ovom radu korišćena je konkretna sintaksa zasnovana na upotrebi grafičke notacije.

Na slici 1 prikazan je primer modela prototipa aplikacije kreiran u razvijenom namenskom jeziku korišćenjem grafičke konkretne sintakse.

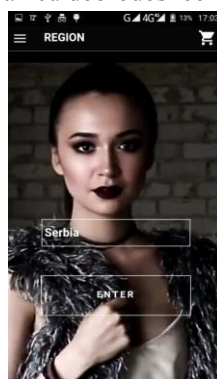


Slika 1. Primer grafičke konkretne sintakse

5 PRIMER GENERISANOG PROTOTIPA APLIKACIJE

U ovom poglavlju biće prikazane pojedine stranice koje predstavljaju prototip aplikacije koji je generisan na osnovu modela napravljenog pomoću *MobileFlow* namenskog jezika. Konkretna grafička sintaksa prikazana je na slici 1.

Prilikom pokretanja prototipa aplikacije, kao prva stranica prikazuje se stranica za odabir regiona. Ova stranica prikazana je na slici 2. Nakon odabira regiona, prelazi se na stranicu dobrodošlice koja je prikazana na slici 3.



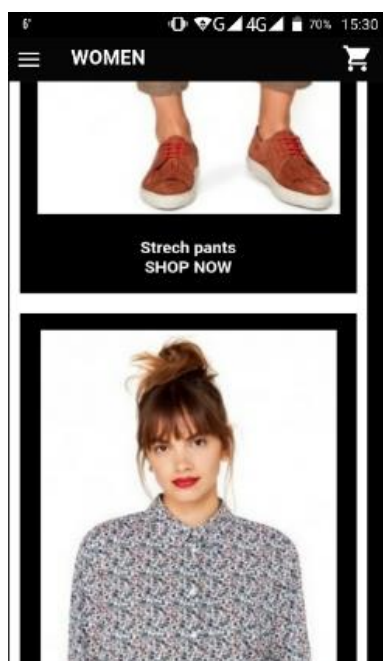
Slika 2. Stranica za odabir regiona



Slika 3. Stranica dobrodošlice

Stranica kojoj se može pristupiti iz menija, a koja je prikazana na slici 4, je stranica za prikaz kategorija odeće za žene.

Ova stranica sadrži link ka stranici za prikaz korisničke korpe, i sa ove stranice moguće je pristupiti meniju aplikacije.



Slika 4. Stranica za prikaz kategorija odeće za žene

6 ZAKLJUČAK

Upotreba mobilnih aplikacija iz domena prodaje odeće je veoma rasprostranjena. Zbog svoje rasprostranjenosti postoji potreba za brzim razvojem ovih aplikacija. Primenom MDE (*Model Driven Engineering*) pristupa razvoju softvera, vreme razvoja prototipa aplikacije za prodavnice odeće je znatno smanjeno, jer se pomoću generatora kôda dobija gotov prototip, bez potrebe za programiranjem. Generisan prototip dobija se na osnovu modela specifikiranog pomoću namenskog jezika.

Ovaj rad sadrži opis softverskog alata za razvoj prototipova aplikacija za domen prodaje odeće. Alat se sastoji iz namenskog jezika *MobileFlow* i generatora kôda. Razvijeni namenski jezik pored apstraktne sintakse, sadrži i konkretnu sintaksu jezika. Generator kôda sadrži statički i dinamički kôd. Statički kôd je unapred definisan od strane programera uz pomoć radnog okvira *Ionic*, dok se dinamički deo dobija na osnovu strukture modela zadatog na ulazu u proces generisanja programskog kôda. Na ulazu u proces generisanja programskog kôda zadaje se specifikacija modela kreirana u skladu sa odgovarajućim meta-modelom, dok se na izlazu dobija izvršivi programski kôd za ciljnu platformu. Moguće je razviti više različitih generatora koji će, na osnovu podataka dobijenih iz modela specifikiranog pomoću namenskog jezika, generisati različite prototipove mobilnih aplikacija za prodaju odeće.

Kako bi generisani prototipovi bili bolji, sa detaljnijim korisničkim interfejsom, neophodno je proširiti meta-model novim konceptima. Novi koncepti mogu biti koncept pretrage, koncepti plaćanja preko *PayPal*, koncepti učitavanja *product* kataloga, itd.

Pored proširenja meta-modela, potrebno je omogućiti učitavanje *product* kataloga odvojeno u prototip tako da se prototip pored domena prodaje odeće može koristiti u domenu prodaje bilo kog proizvoda.

Product katalog proizvoda je kolekcija ponuda proizvoda koja kombinuje informacije o cenama i dostupnosti sa specifikacijama proizvoda koji opisuju odnose između proizvoda, usluga koje se koriste za realizaciju proizvoda i potrebnih resursa.

Klikom na odgovarajuće dugme bilo bi omogućeno učitavanje bilo kog *product* kataloga, čime bi se doprinelo generalizaciji aplikacije, koja bi mogla da se iskoristi u domenu prodaje različitih proizvoda, a ne samo odeće.

LITERATURA

- [1] Dostupno na:
<https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/>, januar 2018.
- [2] Dostupno na:
<https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/>, januar 2018.

Kratka biografija:

Nada Ignjić rođena je 8.7.1992. godine u Novom Sadu. Školske 2011/2012. godine upisala je Fakultet tehničkih nauka u Novom Sadu, odsek Računarstvo i automatika. Osnovne akademske studije završila je 2015. godine. Školske 2015/2016 upisuje master studije na smeru Softversko inženjerstvo i informacione tehnologije, usmerenje Inženjering informacionih sistema. Položila je sve ispite propisane planom i programom.

IZAZOVI U IMPLEMENTACIJI RAČUNARSKE IGRE**CHALLENGES IN THE IMPLEMENTATION OF A VIDEO GAME**

Andrej Miletić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – PRIMENJENE RAČUNARSKE NAUKE I INFORMATIKA

Kratak sadržaj – Ovaj rad se bavi procesom razvoja jedne video igre. Opisuje se koje atribute treba video igra da sadrži, primere alata za laku implementaciju i rešenje nekih problema koje igra ovog tipa sadrži.

Abstract – This paper deals with the process of developing a video game. It describes which attributes a video game needs to have, an example of tools for an easy implementation, and the solution of some of the problems that a video game of this type contains.

Ključne reči: Video igra, playability, cloud computing, multiplayer, sinhronizacija

1. UVOD

Iako je industrija video igara relativno nova, ona se danas nikako ne može smatrati za malom. Postoji velik broj žanrova igara – RPG, strategy, hack&slash, shooter, MMO, MOBA, kao i još mnoštvo drugih i njihove kombinacije. Jedna stvar koja je zajednička za sve igre jeste da osobe koje su ih razvile žele da one budu uspešne, a da bi bile uspešne potrebno je da budu zabavne. Tako da je prvi korak u procesu razvoja jedne video igre razumeti šta privlači te igrače i osmisliti mehanike koje se sa tim poklapaju.

Nakon ovog je potrebno odrediti koji alati su nam potrebni za razvoj igre. Razvijanje sopstvenog alata je uglavnom isplativo samo za velike kompanije sa specifičnim potrebama tako da će sada biti odabrani neki od gotovih rešenja. Prva odluka je u kom game engine-u će igra biti pravljenja. Trebalo bi odabrati onaj koji nudi jednostavno rešenja za probleme koje predviđamo da ćemo imati. Ovde razvijana igra treba da podrži automatsko ponašanje protivničkih karaktra, kao i multiplayer koji zahteva sinhronizaciju igrača. Za multiplayer i sinhronizaciju bi nam idealno bilo cloud rešenje koje bi brinulo o stanjima na računaru svakog od igrača, a koje bi se uklapalo sa odabranim game engineom.

2. PLAYABILITY

Playability predstavlja set osobina koje opisuju iskustvo igrača (Player Experience) dok koristi određeni sistem za igru čiji glavni cilj je da donese uživanje i zanimaciju dok igrač igra sam ili u društvu [1].

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Aleksandar Kupusinać, van.prof.

Osobine sa kojima definišemo playability su [1]:

- **Satisfaction (Satisfakcija):** Ovo možemo definisati kao zadovoljstvo koje proističe iz igranja date video igre.

Satisfakciju dalje možemo okarakterisati osobinama kao **Fun (zabava)** – glavni cilj video igre je da zabavi igrača, tako da ukoliko igra kao celina ne ispunjava taj uslov ne može biti ni satisfakcije, **Dissapointment (razočarenje)** – video igra ne sme na neki način da izazove osećaj razočarenja ili nelagode, pošto u suprotnom mogu da prestanu da igraju, **Attractiveness (privlačnost)** – odnosi se na elemente video igre koji povećavaju satisfakciju igrača

- **Learnability (Mogućnost učenja):** Ovo definišemo kao igračev kapacitet da razume i savlada sisteme i mehanike igre (ciljeve, pravila, itd...). Ovo možemo dalje podeliti na: **Game knowledge (poznavanje igre)** – u koliko meri je igrač unapred upoznat sa igrom utiče na to koliko teško će biti da se savlada “learning curve” igre. **Skill (veština)** – prikazuje koliko dobro igrač igra video igru nakon što savlada osnovne koncepte te igre, kao i način na koji pristupa problemima koji su mu prezentovani za maksimalan učinak. **Difficulty (težina)** – kolika je težina zavisi od toga koliko je oštra “learning curve” u odnosu na nivo veštine igrača i koliko dugo igra. Veća težina može da dovede do dodatnog napora od strane igrača da nauči kako da ispravno igra, ili da dovede do frustracije. **Frustration (frustracija)** – ovo je uglavnom deo procesa učenja kada igrač još uvek nije savladao sve koncepte igre, ali ukoliko se ovo osećanje javlja kroz celu igru može postati problematično. **Speed (brzina)** – brzina kojom se novi koncepti i sadržaj ubacuje u igru direktno utiče na proces učenja te igre. **Discovery (otkrivanje)** – da su različiti delovi igre osmišljeni na način tako da je igraču potrebno sve manje vremena da usvoji nove i poboljša postojeće osobine kako bi ispunio ciljeve igre.

- **Effectiveness (efektivnost):** Ovo definišemo kao potrebne resurse i vreme pre nego što igrač krene da oseća zabavu dok ispunjava ciljeve igre. Dalje se sastoji od : **Completion (završavanje)** – Igra je efektivnija ukoliko je procenat završetka igre veći, tj. ukoliko igraču nijedan deo igre nije bio previše dosadan za završiti. **Structuring (struktuiranje)** – igra je dobro struktuirana ukoliko ima dobar balans između različitih ciljeva koje treba završiti, i različitih izazova za premostiti, tj. ukoliko igrač ne iskusi dosadu zbog prevelikog ponavljanja istih stvari.

- **Immersion (imerzija):** Možemo definisati kao sposobnost igre da sebe učini dovoljno uverljivom tako da igrač oseti određenu “vezu” sa tim virtualnim svetom. Dalje se može podeliti na: **Conscious Awareness (aktivna svesnost)** – na kom stepenu je igrač aktivno svestan kakve posledice po taj virtualni svet imaju njegove akcije u igri. **Absorption (apsorpcija)** – igrač koji je u

potpunosti “apsorbovan” u igru u potpunosti će se u potpunosti i posvetiti igri i savladavanju njenih izazova. **Realism (realizam)** – što je igra realističnija (u korišćenju kontrola, prezenovanja sadržaja, atmosfere...) to će igrač biti pod većom imerzijom. **Dexterity (spretnost)** – odnosi se na igračevu sposobnost da interaguje sa igrom pomoću ponuđenih kontrola (interaktivna spretnost) i izvršavanje različitih pokreta i akcija u virtualnom svetu. **Socio-Cultural Proximity (socio-kulturna blizina)** – igrač će biti manje ili više pod osećajem imerzije u zavisnosti od toga koliko on može da se poistoveti sa svetom koji mu je predstavljen. Na to utiču faktori kao što su starost, pol, poreklo, itd..

- **Motivation (motivacija):** Ovo definišemo kao set karakteristika igre koje podstiču igrača da vrši određene akcije sve dok ih ne završi. Motivaciju dalje određujemo sa: **Encouragement (ohrabrenje)** – koliko je igrač “ohrabren” da igra dalje zavisi od nivoa samopouzdanja koje ima kada se susreće sa novim izazovima koji su mu prezentovani. **Curiosity (radoznalost)** – može se proizvesti dodavanjem opcionalnih ciljeva i izazova koji igraču daju mogućnost da svojevolejno interaguje sa većim brojem elemenata igre. **Self-improvement (samo poboljšanje)** – kada igrač ili njegov karakter razvije novi set sposobnosti ili poboljša trenutne, ili kako bi mogao da pređe novi izazov, ili samo zato što uživa u njima. **Diversity (raznolikost)** - što igra ima više zanimljivih elemenata i mogućnosti da ponudi igraču, manja je šansa da će doći do osećaja monotomije.

- **Emotion (emocije)** – Ovo se odnosi na igračeve podsvesne reakcije na video igru koji pokreću određena osećanja. Dalje delimo na: **Reaction (reakcija)** – igrač reaguje na video igru zato što je ona izvor različitih stimulansi, i te reakcije izazivaju različita osećanja. **Conduct (ponašanje)** – video igre imaju mogućnost da utiču na ponašanje igrača dok igra igru tako što ga vode kroz niz željenih osećanja preko stimulansa koji proizvode. **Sensory Appeal (senzorna privlačnost)** – igra mora da u igraču može da izazove interes ili želju za njenim estetskim elementima kako bi pojačala igračevu privlačnost prema njoj. To može preko različitih senzornih kanala, npr. audiovizualnih – odgovarajuća ambijentalna muzika ili mesta u kojima igrač može da zastane i uživa u pogledu na virtualni svet.

- **Socialization (socijalizacija)** – Ovo definišemo kao set atributa, elemenata, i resursa koje igra ima i koji promovšu društvenu dimenziju igračevog iskustva u grupnom scenariju. Ovakvo društveno iskustvo može učiniti da da igrač ceni igru na drugačiji način, zahvaljujući vezama koje je izgradio sa drugim igračima (ili čak karakterima iz video igre). Socijalizacija se primenjuje i na veze sa karakterima iz video igre, npr. biranje karaktera sa kojim se igrač može poistovetiti, interakcija sa karakterima zarad dobivanja više informacija, traženje pomoći, trgovina, pregovaranje, uticanje na druge karaktere kako bismo pozitivno ili negativno uticali i na krajnji cilj. Socijalizaciju možemo dalje deliti na: **Social Perception (društvena percepcija)** - stepen društvene aktivnosti koju igrači postižu dok igraju igru u multiplayer okruženju umesto sami. **Group Awareness (svest o grupi)** - odnosi se na svesnost igrača da je deo tima koji ima iste ciljeve i da uspeh grupe zavisi koliko dobro tim sarađuje. **Personal Implication (lična**

implikacija) – igrač mora biti svestan da individualni uspesi dovode takođe do uspeha grupe. Takođe igra mora biti dizajnirana tako da igrači budu svesni svoje uloge unutar tima. **Sharing (deljenje)** – kada igrač igra u grupi resursi treba da se dele ravnomerno. Kontrola svih resursa u grupi od strane jednog igrala uglavnom dovodi do nepovoljnog ishoda. **Communication (komunikacija)** – multiplayer igre bi trebalo da imaju mehanizme za komunikaciju između igrača preko kojih mogu razmenjivati informacije. **Interaction (interakcija)** – u zavisnosti od pravila igre i uslova za pobjedu igrači će na različite načine interagovati jedni sa drugima. Možemo navesti tri načina interakcije: Kompetitavan (kada uspeh jednog igrača ili tima dovodi do poraza ostalih), sarađivački (uspeh pojedinca je zamenjen uspehom cele grupe, svi sarađuju ka zajedničkom cilju), i kooperativan (igrači imaju svoj pojedinačan cilj, ali opcionalno mogu sarađivati sa drugima kako bi poboljšali svoje šanse).

3. ODABIR ALATA

Iako postoji određena količina izbora kada je u pitanju odabir game engine-a (Unity, Unreal 4, CryENGINE, Source 2) za potrebe ovog projekta je sasvim pogodan Unity game engine. On je jedan od najpopularnijih engine-a na tržištu, a personal verzija je besplatna za upotrebu. Kao software je uspešan i popularan tako da se na njemu aktivno radi i konstantno se dodaju nove funkcionalnosti. Takođe je multiplatform i veoma lako može da se izgradi verzija igre, bilo za PC, android, iOS...

Sa obzirom da igra koju razvijamo treba da sadrži online multiplayer funkcionalnost potreban nam je sistem koji će moći da igrač poveže u realnom vremenu (nešto što Unity još uvek nema dovoljno dobro razvijeno). Kako ne bismo sami morali da razvijamo sistem koji bi nam ovo omogućio biće iskorišćeno cloud rešenje.

3.1. Cloud Computing

Cloud computing je model koji omogućava na zahtev, sveprisutan, pogodan pristup preko mreže deljenim, podesivim kompjuterskim resursima (mreže, server, aplikacije, servisi, itd.) koji mogu biti brzo snabdevani ili oduzeti uz minimalan napor ili interakciju provajdera. Zagovornici cloud-a tvrde da on omogućava kompanijama da izbegnu velike početne troškove infrastrukture (kao što je kupovina servera).

Neke od osobina koje cloud treba da iskazuje [2]:

- **Autonomnost** – Klijenti nisu svesni tehničkih kompleksnosti ponuđenih usluga. To može obuhvatati korišćene tehnologije, fizičku lokaciju, mreže, ljudske resurse koji su potrebni za održavanje, itd.

- **Predefinisani QoS (Quality of service)** – Cloud provajderi navode svoje QoS kako bi obavestili potencijalne klijente o nivou usluga koji mogu da očekuju. QoS daje mogućnost tim klijentima da odaberu od mogućih provajdera onog koji može da ispuni njihove tehničke zahteve.

- **Internet-based** – Neki smatraju da je ime za Cloud originalno došlo zato što se u IT-u oblik oblaka koristio da grafički prikaže internet. Cloud servisi se su hostovani često daleko od klijenta (može biti i drugi kontinent) i njima se pristupa preko interneta.

- **Laki za korišćenje** – Cloud provajderi bi trebalo da nude interfejs preko kojih će klijenti moći da sa lakoćom koriste nuđene usluge. Ovo obuhvata i GUI za administraciju i API za developere.

- **Skalabilnost** – Klijenti nisu ograničeni sa fiksnom količinom resursa, već mogu po potrebi da uzimaju više ili oslobađaju već zauzete resurse po trenutnoj potrebi. To bi trebalo da uključuje potrebe u toku dana (npr. ujutru i uveče ima manje prometa nego u toku popodneva).

- **Nisu skupi** – Cloud provajderi nude malim i srednje velikim preduzećima (koji ne mogu sami da priušte data centre) značajno jeftiniju opciju. Ova ušteda proizilazi i iz činjenice da resursi koje provajderi nude nisu namenjeni samo jednom klijentu, već se dele između više njih.

Model pretplate – Klijenti koriste usluge koje su im potrebne od provejdera, a zauzvrat im se uglavnom na kraju meseca naplaćuje račun u zavisnosti od potrošnje.

3.2. GameSparks

GameSparks je jedan od cloud BaaS (Backend as a Service) provajdera za game developere. Krenuo je sa radom u avgustu 2012. i danas je jedan od popularnijih izbora. Osnova GameSparksa je izgrađena na Microsoft Azure-u. GameSparks nudi SKD-ove za razna okruženja kao što su Unity (koje je iskorišćeno), Unreal, Android, Corona, itd. Nakon pravljenja naloga može se pristupiti web portalu za pravljenje novih i konfiguraciju postojećih igara. Podržava pravljenje i single player i multy player igara bez potrebe za integracijom sa drugim servisima, mada još uvek nema najbolju podršku za open-world igre. GameSparks takođe podržava spajanja naloga od različitih provajdera u jedan player account, čime ostvaruje cross-platform gaming, mada je najbolja podrška za Facebook i Twitter ali je moguće i za još. Ovim se omogućava prenošenje i liste prijatelja u igru.

Developerima se nudi na korišćenje Cloud Storage sa Mongo DB bazom podataka. Iz developer platforme se može pristupiti svim podacima u bazi za odabranu igru pomoću NoSQL notacije. Neke od kolekcija u bazi koje se automatski dobijaju i popunjavaju su za listu igrača, mečeva, tim četa, svih napravljenih leaderboard-a ili matchmaking-a, itd. Osim ovih developer može praviti proizvoljne kolekcije i pristupati mu preko Cloud Code-a što čini potencijal velikim za šta je sve moguće uraditi na bekendu. Svim podacima u bazi se može pristupiti preko web portala.

Veća kontrola nad bekendom se dozvoljava pomoću Cloud Code-a, kojim se može definisati zeljena backend logika. Piše se JavaScript-u i u web portalu, što ga čini cross platform tako da ukoliko bi došlo do izmene klijentskog okruženja backend bi ostao neizmenjen. Cloud Code se može koristiti na više načina. Jedan je pisanje koda u vidu Interceptor-a. Može se pisati kod koji će se izvršavati pre nego što se izvrši neki zahtev sa klijenta, pre nego što se pošalje odgovor klijentu, nakon pravljenja poruke ili pre nego što poruka stigne do određenog igrača. Drugi je pisanje proizvoljnih događaja na serveru koji se onda eksplicitno pozivaju sa klijentske strane i služi proizvoljnoj svrsi. Jedna od korisnih usluga koje nudi GameSparks vezana za Cloud Code i razvoj

igre je Test Harness. To je alat unutar web portala koji može da simulira klijenta, tj. dozvoljava se preko njega šalju API zahtevi direktno iz portala i da se vidi odgovor. Na ovaj način se može testirati novi ili menjati postojeći Cloud Code pre nego što se naprave ikakve izmene na klijentu.

GameSparks podržava matchmaking po određenom kriterijumu, kao i pravljenje leaderboard-a. Nudi laku implementaciju četa između igrača, kao i slanje notifikacija.

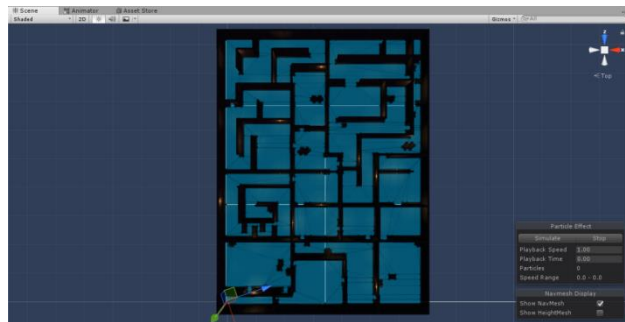
4. IMPLEMENTACIJA VIDEO IGRE

Pravljenje ove igre je zahtevalo rešenje dva veća problema – kretanje kompjuterski kontrolisanih karaktera i sinhronizaciju između više igrača. Srećom odabrani alati daju jednostavno rešenja za oba.

4.1. Kretanje kompjuterski generisanih karaktera

Protivničko kretanje je zamišljeno tako da se svaka instanca neprijatelja konstantno približava igraču sve dok ne bude dovoljno blizu da napadne. Unity ovako nešto u ogromnoj meri olakšava time što nudi ugrađenu podršku za pathfinding – NavMesh, skraćeno od Navigation Mesh. NavMesh se pravi pomoću procesa koji se zove NavMesh Baking, koji skuplja sve Render Mesh-ove i Terrain-ove svih objekata u sceni koje označimo za ovaj proces. Posle ovog procesa od Unity-a dobijamo aproksimaciju prohodnih oblasti na našoj sceni. Prilikom ovog procesa imamo slobodu podešavanja stvari kao što su koliko blizu ćemo dozvoliti da nam se karakter približi zidu, ili pod kojim uglom možemo da se penjemo uz padinu.

Nakon "Bake-ovanja" u našoj sceni će se generisati prohodna površina koji možemo da koristimo za naše protivnike kontrolisane od strane igre (Slika 4.1.1.).



Slika 4.1.1. Mapa sa prohodnim površinama (plavo) za protivničke karaktere

Kada smo ovo odradili potrebno je da svim game objektima za koje želimo da koriste ovaj NavMesh dodamo NavMeshAgent komponentu. Posle toga je preostalo samo da napišemo kratku skriptu u kojoj ćemo u svakom frejmu neprijateljskim karakterima proslediti igračevu poziciju kao novu tačku ka kojoj treba da se kreću (ukoliko i igrač i neprijateljski karakter imaju više od 0 života).

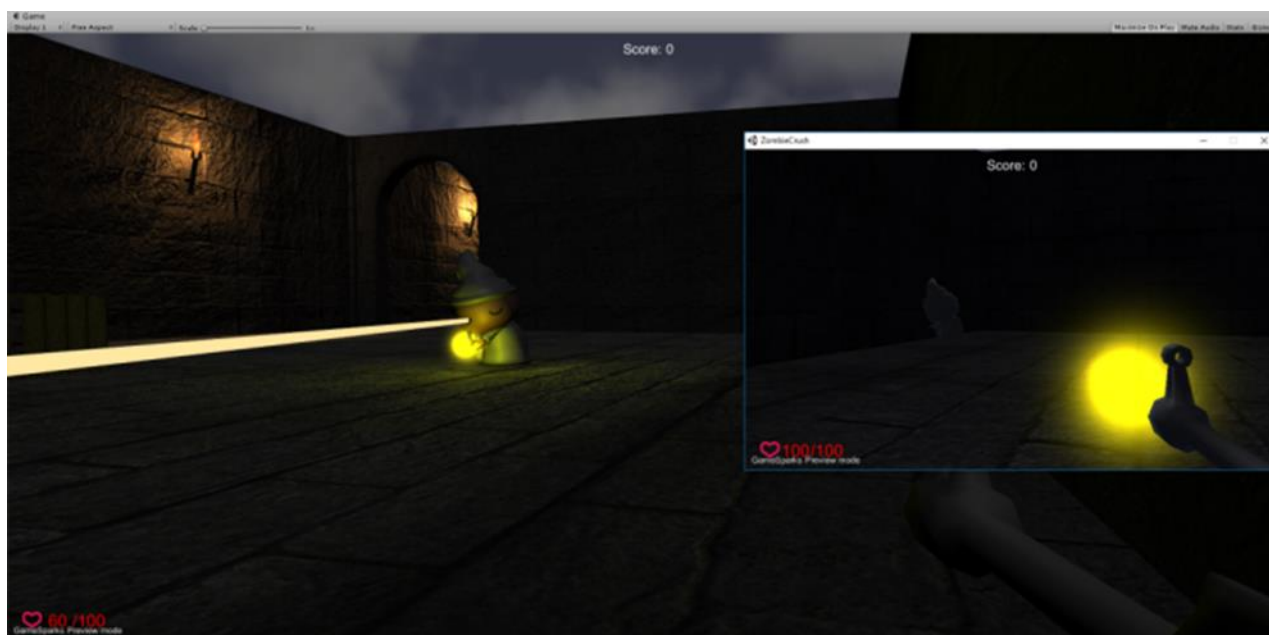
4.2. Sinhronizacija više igrača

Kretanje igrača se u igri izračunava iznova u svakom frejmu. Ukoliko bismo svakog frejma slali informacije o poziciji/rotaciji igrača vrlo lako bi se moglo desiti da se zaguši mreža sa obzirom da se igre ciljaju da budu 30fps(frames per second) za konzole i 60+ fps za PC.

Tako da ćemo mi slati te podatke rede (10 puta u sekundi u ovom primeru). Podaci koji se šalju vezano za kretanje su x i z pozicija igrača (u igri nema kretanje po y osi), trenutna rotacija igrača, i trenutni smer kretanja igrača kako bi se mogla vršiti predikcija kretanja između paketa. Napad (pucanje) igrača se obrađuje na malo drugačiji način.

S obzirom da se pucanje igrača ne dešava na redovnim intervalima, nego samo kada igrač odluči, onda se odmah po dešavanju šalje svim ostalim igračima.

Potrebne informacije za ovo su pozicija odakle se desio pucanj, pozicija gde je završio pucanj, i kog igrača je pogodio (ili informacija da nikog nije pogodio). Za kalkulaciju da li je drugi igrač pogođen je odlučeno da se dešava sa strane igrača koji je pucao. Ova odluka je zato što vizuelno gledano igrač koji je pucao će uvek videti kada je nekog pogodio, dok pogođeni igrač može imati okrenuta leđa.



Slika 4.2.1. Igrač koji je pucao u jednoj instanci igre (manji prozor) i prenet pucanj u drugu instancu igre (veći prozor)

Odabir dobrih alata može višestruko smanjiti troškove razvoja, dok odabir nepogodnih može imati suprotan uticaj.

Na primer situacija gde tek kasno u razvoju se shvati da backend servis nema odgovarajuću podršku za količinu igrača koji igraju datu igru.

Najkompleksnija stvar koju je trebalo implementirati unutar ove igre je bio multiplayer, gde je trebalo odlučiti koje informacije treba slati, i koliko često, a da se ne zaguši mreža, niti da se smanji „player experience“ zbog „seckanja“ karaktera, kao i da se minimizuje uticaj uvek prisutnog kašnjenja kada imamo udaljene servere.

Drugi problem, ponašanje neprijateljskih karaktera tako da se kreću ka igraču, srećom je bio lak za rešiti pomoću alata koje Unity nudi u vidu NavMesh-a.

Ideja je da će sa ovakvom odlukom biti najmanje „nije fer“ situacija. Kada je ovaj paket poslat biće obrađen kod drugih igrača koji će iscertati pucanj onog igrača koji je poslao paket, i skinuti životne poene onom igraču koji je označen kao pogođen (ukoliko neki jeste).

5. ZAKLJUČAK

Razvijanje jedne ozbiljnije video igre je velik projekat i zahteva veliki broj stručnjaka iz različitih oblasti (programeri – opet iz više oblasti, stručnjaci za zvuk, modeliranje, animaciju, itd.) koji moraju zajedno raditi i komunicirati kako bi posao bio dobro odrađen. Posle osnovnog koraka osmišljavanja šta igra treba da predstavlja potrebno je odlučiti koje alate koristiti za dati posao. Iako bi za svaku igru mogli razvijati poseban set alata to bi bilo vremenski i finansijski neisplativo, već je bolje ili jednom razviti opštije alata koje je moguće koristiti u više situacija, ili koristiti već napravljenje.

6. LITERATURA

- [1] Jose Luis González Sánchez, Natalia Padilla Zea, Francisco L. Gutiérrez, “From Usability to

Playability: Introduction to Player- Centred Video Game Development Process”, Lecture Notes in Computer Science, vol 5619. doi: 10.1007/978-3-642-02806-9_9

- [2] Qusay F. Hassan, “Demystifying Cloud Computing” The Journal of Defense Software Engineering (CrossTalk)

Kratka biografija:

Andrej Miletić rođen je u Kraljevu 1993. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Računarstvo i automatika odbranio je 2016.god

**UPOREDNA ANALIZA JAVASCRIPT BIBLIOTEKA I OKVIRA -
REACT, ANGULAR, VUE****A COMPARATIVE ANALYSIS OF JAVASCRIPT FRAMEWORKS AND LIBRARIES -
REACT, ANGULAR, VUE**

Aleksandra Sopka, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Tema rada i istraživanja je uporedna analiza između **JavaScript** okvira i biblioteke – **React-a**, **Angular-a** i **Vue-a** i predstavlja rezultat zainteresovanosti za trendom Internet aplikacija koje se oslanjaju na klijentsku stranu. U radu je opisan svaki od okvira/biblioteke kroz razne aspekte kao što su komponente, povezivanje podataka i upravljanje stanjima, rutiranje, itd. Nakon opisa sva tri, sledi deo gde su međusobno upoređeni.

Ključne reči: *React, Angular, Vue, JavaScript, biblioteka, okvir, klijent, server, veb čitač*

Abstract – The subject of this research is a comparative analysis of Javascript frameworks and libraries – **React**, **Angular** and **Vue**, and is inspired by the trend of client side Internet applications. The paper describes each of the frameworks / libraries through various aspects, such as components, data binding and state management, routing, etc. After that comes the part where they are compared.

Keywords: *React, Angular, Vue, JavaScript, library, framework, client, server, browser*

1. UVOD

Single Page Application (SPA) je aplikacija koja radi unutar veb čitača i nakon inicijalnog učitavanja stranice ne zahteva ponovna učitavanja stranice tokom korišćenja [1]. U vreme kada koncept **SPA** postaje popularniji, razvijaju se i programski jezici za izradu frontenda. **JavaScript**, kao i njegove biblioteke se trenutno razvijaju brže nego ikada, a nove tehnologije se često pojavljuju. Biblioteke i okviri imaju tendenciju da olakšaju razvoj **JavaScript-a** i prošire njegove mogućnosti.

2. JAVASCRIPT OKVIRI I BIBLIOTEKE

JavaScript okviri i biblioteke već dugo postoje. Oni su postali popularni po prvi put sa **jQuery** bibliotekom 2006. godine. Još robusniji **JavaScript** okviri su postali poznati javnosti negde između 2010. i 2011. godine - **AngularJS** i **Ember**. **AngularJS** je bio prvi pravi okvir koji je spojio vezivanje podataka, rutiranje i šabloniranje (**templating**) u jednom paketu.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Aleksandar Kupusinac, vanr.prof.

Ember je ubrzo došao i ponudio nekoliko poboljšanja u odnosu na **AngularJS**, npr. optimizovano rutiranje.

Poslednjih godina, kontinuirano su se pojavljivali potpuni, sveobuhvatniji i optimizovaniji **JavaScript** okviri. Popularnost okvira **JavaScript-a** je brzo rasla, ruku pod ruku sa trendom **SPA**. Trenutno ne postoji okvir koji se smatra najboljim, ali ima nekoliko dobrih opcija sa neznatno različitim funkcionalnostima. Prema tome, budući pravac **JavaScript** okvira je i dalje prilično neizvestan [2].

2.1. Okvir vs biblioteka

Okviri diktiraju kako će projekat biti strukturiran, a biblioteke su građevinski blokovi koji se mogu koristiti bilo gde.

Ključna razlika između biblioteke i okvira je inverzija kontrole (*Inversion of Control*). Pri pozivu metode iz biblioteke, programer ima kontrolu. Sa okvirom je kontrola obrnuta: okvir „poziva“ programera.

„Inverzija kontrole je ključni deo onoga što čini okvir drugačijim od biblioteke. Biblioteka je skup funkcija, obično organizovane u klase, koje se mogu pozvati. Svaki poziv obavi neku funkciju i vrati kontrolu klijentu. Okvir oličava neki apstraktni dizajn sa više ugrađenih ponašanja. Da bi se koristio, potrebno je da programer ubaci svoje „ponašanje“ na različita mesta u okviru kao podklasu ili uključivanjem sopstvenih klasa. Kod okvira tada poziva programerov kod na ovim tačkama [3].“

2.2 React, Angular, Vue – opšte karakteristike

React je **JavaScript** biblioteka namenjena za kreiranje kompozitnih korisničkih interfejsa. Podržava kreiranje ponovno iskoristivih komponenti, koje predstavljaju podatke koji se menjaju u toku vremena. Velik broj ljudi koristi **React** kao **V** u **MVC**. **React** izdvaja **DOM** od korisnika, nudeći jednostavniji programerski model i bolje performanse. **React** takođe može da se iscrta na serveru koristeći **Node**, i može pokretati native aplikacije pomoću **React Native-a**. Implementira jednosmerni tok podataka, što umanjuje **boilerplate** i lakše ga je razumeti u odnosu na tradicionalno vezivanje podataka.

Angular je platforma i okvir namenjen za izgradnju klijentskih aplikacija u **HTML-u** i **TypeScript-u**. **Angular** implementira osnovne i opcione funkcionalnosti kao set **TypeScript** biblioteka koje se unose u aplikacije. Pod **Angular-om** se misli na verzije **Angular-a** 2, 4, 5 i 6, i predstavlja potpunu preradu svoje preteče **AngularJS-a** (**Angular** 1).

Osnovni gradivni blokovi **Angular** aplikacije jesu moduli, koji pružaju kontekst za građenje komponenti. Ovi moduli objedinjuju srodne delove koda u funkcionalne celine; **Angular** aplikacija je definisana skupom modula. Aplikacija mora da sadrži bar jedan modul - a to je **root** (korenski) modul koji omogućava butstrapovanje. Osim **root** modula, obično sadrži i više funkcionalnih modula.

- Komponente definišu prikaze, koji predstavljaju skup elemenata ekrana koje **Angular** bira i menja prema logici i podacima programa.
- Komponente koriste servise koji pružaju određene funkcionalnosti koje nisu direktno povezane sa prikazima. Snadbevači servisima mogu da se injektuju u komponente kao zavisnosti, čineći kod modularnim, višekratnim i efikasnim.

Komponente i servisi su, zapravo, klase sa dekoratorima koji obeležavaju njihov tip i pružaju metapodatke koji govore **Angular**-u kako da ih upotrebi.

Vue je progresivan okvir za izgradnju korisničkih interfejsa. Za razliku od drugih monolitskih okvira, **Vue** je dizajniran, od dole do gore, da bude postepeno usvojen. Osnovna biblioteka je fokusirana samo na sloj za prikaz i integraciju s drugim bibliotekama i postojećim projektima. S druge strane, **Vue** se u kombinaciji sa modernim alatima i odgovarajućim bibliotekama koristi i kod **SPA**.

Vue je minimalan, ali moćan okvir. Omogućava programeru da brzo napravi prototip od manjih aplikacija, i neće stati na put prilikom strukturiranja velikih frontend sistema. Jedna od njegovih najvećih prednosti je to što je **lightweight** i lak za učenje. **Vue**, kao i **Angular**, podržava i jednosmerno i dvosmerno povezivanje podataka, podržava komponente, direktive, rutiranje, rukovanje događajima, itd [4].

3. REACT, ANGULAR, VUE – UPOREDNA ANALIZA

U ovom delu će prvo biti opisani okviri/biblioteka nakon poređenja na primeru jednostavne aplikacije – dodavanje, ažuriranje i brisanje članaka, kroz različite aspekte kao što su poređenje komponenti, načina povezivanja podataka, upravljanja stanjima, zatim načina rutiranja.

Nakon toga će biti upoređene performanse sva tri okvira/biblioteke na osnovu podataka preuzetih sa xxx.

3.1. Komponente

Većina modernih frontend **JavaScript** okvira/biblioteka pruža neku vrstu podrške za razvoj zasnovan na komponentama. Ovo je neverovatno važan korak u pravcu razvoja Interneta. Komponente pružaju način pisanja malih delova sa konzistentnim **API**-jem koji se lako mogu orkestrirati kao deo većeg ekrana, aplikacije ili sistema.

React pruža mogućnost definisanja komponenti kao klasa ili funkcija. Komponente definisane kao klase trenutno pružaju više mogućnosti. Klase se pišu u stilu **ES6**.

Komponente u **Angular**-u funkcionišu malo drugačije od onih u **React**-u. Komponente su definisane kao klasa sa

@Component dekoratorom, a metapodaci komponente su dati odmah. **Angular** aplikacija uvek sadrži jedan korenski modul koji ima svoju komponentu.

Komponente u **Vue** radnom okviru su slične onima u **Angular**-u. **Vue** ima veoma jednostavnu i lako razumljivu sintaksu. Kao i u **React**-u i **Angular**-u, komponente u **Vue**-u imaju svoje izolovano područje koje ima svoja svojstva.

Stil pisanja **React** komponenti je najbližiji **JavaScript**-u, a komponente u **Angular**-u i **Vue**-u pružaju malo više funkcionalnosti.

3.2. Povezivanje podataka i upravljanje stanjima

Povezivanje podataka (**data binding**) je proces koji uspostavlja vezu između prezentacionog sloja aplikacije i poslovne logike aplikacije. Kod **JavaScript** okvira/biblioteka povezivanje podataka može biti jednosmerno ili dvosmerno.

React podržava samo jednosmerno povezivanje podataka. Prednost modela jednosmernog povezivanja podataka jeste jednostavnost, ali je uglavnom neophodno da aplikacija može da menja stanje svoje roditeljeske komponente u nekom trenutku. Ovde je potreban inverzan protok podataka. Inverzan tok podataka predstavlja prenošenje podataka sa **HTML** polja za unos u komponentu. U **React**-u, to se postiže pomoću callback funkcija, vezivanjem upravljača događajima (event handler) za promene u **DOM** događajima, i preuzimanjem vrednosti **HTML** unosnog polja iz objekta događaja (**event object**).

Angular i **Vue** podržavaju dvosmerno povezivanje podataka. U ovom slučaju, upravljači događajima nisu potrebni. U suštini, upravljači događajima postoje, ali su upravljani od strane radnog okvira i sakriveni od programera.

I jednosmerno i dvosmerno povezivanje podataka imaju svoje prednosti i mane. Ono što dobije na jednostavnosti, gubi na upravljanju i performansama. Jednosmerni podaci su lakši za upravljanje i održavanje u jednosmernom smeru toka podataka jer se upravljači događajima postavljaju ručno. Kada je stanje komponente uvek poznato programeru, tok podataka ima manje šanse da izazove neočekivano ponašanje ili praznine u izvođenju.

3.3. Rutiranje

Rutiranje se može posmatrati kao neophodno svojstvo kod **SPA**.

React ne sadrži ruter (**Router**), niti sličnu funkcionalnost. Za rutiranje koristi posebnu biblioteku, od kojih je najpopularnija **React Router**. Najnovija verzija ove biblioteke podržava dinamičko rutiranje. Dinamičko rutiranje omogućava kreiranje responsive putanja, tako da se sadržaj stranice može prilagođavati različitim veličinama ekrana, bez potrebe da se osveži stranica.

Dok je u **React**-u neophodna eksterna biblioteka, **Angular** podržava ruter biblioteku prilikom osnovne instalacije. **Angular Router** ne obezbeđuje dinamičko rutiranje, ali postoje eksterne biblioteke za tu svrhu (npr. **Angular-dynamic-routing library**).

Kao i **Angular**, **Vue** podržava zvanični ruter u svojoj osnovnoj instalaciji.

3.4. Učinak

Upoređivanje različitih oblasti u sveobuhvatnim testovima bi dalo pouzdane dokaze o učinku. Ova evaluacija, naime, nije izvodljiva samo strogim poređenjem performansi čineći test aplikaciju sve većom i većom, jer postoje brojni načini za rast aplikacije. Dovoljan broj testova bi doveo do ogromnog radnog opterećenja, a rezultat manjeg broja testova ne bi bio relevantan pri evaluaciji. Bitnija stvar je procena koje opcije i alate daje okvir za ispitivanje i korekciju rupa u performansama (**performance gaps**).

Učinak okvira sastoji se od nekoliko manjih tehničkih faktora u strukturi aplikacije, koji se mogu podeliti u faze izvršavanja programa. Najvažnije od faza su vreme pokretanja i kreiranje i, kasnije, modifikacija **DOM** elemenata.

Rezultati merenja su preuzeti sa [8], gde se može više pročitati o samom merenju, kao i o konfiguraciji mašine koja je upotrebljena za ovo merenje. Poređeni su **Angular** verzija 5.0.0, **React** verzija 16.1.0, nativni **JavaScript** i **Vue** verzija 2.5.3.

Rezultati su ukazivali da su vremena pokretanja **React**-a i **Vue**-a tolerantna – malo veća u odnosu na nativni **JavaScript**, dok je za **Angular** nešto veće nego kod ostalih – skoro duplo veće od **React**-a i **Vue**-a, i skoro tri puta veće od nativnog **JavaScript**-a. U slučaju **React**-a, vreme pokretanja obično u mnogome zavisi od biblioteka koje su uključene u aplikaciju i kako te biblioteke „podnose“ inicijalizaciju. **Angular** pruža nekoliko načina za konfiguraciju početne inicijalizacije (startup initialization). Jedna od prednosti **Angular**-a je i to što se može birati između dve opcije kompajliranja - **just-in-time** kompajliranje (**JIT**) i **Ahead-of-time** (**AOT**) kompajliranje. **Vue** se naziva najbržim trenutno dostupnim **JavaScript** okvirom. **Vue**-ovo vreme pokretanja je veoma optimizovano samo po sebi.

Osim merenja vremena pokretanja, mereno je i vreme renderovanja kod svakog od tri. Rezultati merenja su pokazali da su i **Angular** i **React** i **Vue**, kada je reč o manipulaciji **DOM**-om, sličnih performansi. Postoji par različitih stilova rukovanja **DOM**-om među radnim okvirima/bibliotekama. **React** i **Vue** koriste virtuelni **DOM**, a **Angular** ima svoju strategiju detektovanja promena. Postoje male razlike u performansama, ali svaka od njih ima svoje prednosti. Strogo poređenje performansi radnih okvira u skriptovanju i iscrtaivanju daje određene smernice, ali razlike postaju značajnije samo u slučaju iscrtaivanja velikih struktura podataka.

3.5. Spoljašnje biblioteke

React ima velik broj dostupnih biblioteka. **React** zajednica veoma je aktivna u ovom aspektu, i često postoji grupa programera koji su fokusirani na neku specifičnu funkcionalnost. Ova rešenja mogu biti optimizovanja u poređenju sa **Angular**-om i **Vue**-om, koji pokušavaju da pokriju svu funkcionalnost unutar njih samih.

Za **Angular** postoji manji broj eksternih biblioteka, međutim sve uobičajeno potrebne funkcionalnosti su pokrivene u samom okviru.

Vue ima relativno veliki opseg eksternih biblioteka i ima nekoliko pratećih biblioteka kao što su **Vuex** i **Vue-router**, koje su zvanično podržane i uvek ažurirane kako bi ispoštovale osnovne zahteve **Vue**-a.

3.6. Lokalizacija

Za sva tri okvira, postoji mnogo mogućnosti za implementaciju lokalizacije. **Angular** je jedini koji pruža potpuno i proširivo rešenje kao deo svoje implementacije.

Kod **React**-a jedno od najpopularnijih rešenja jeste jednostavna biblioteka pod nazivom **React-localization**, a kod **Vue**-a za implementaciju ove funkcionalnosti koriste se npr. **Vuex-i18n** i **Vue-localize**.

4. ZAKLJUČAK

Nakon prolaska kroz svaki od okvira/biblioteka, stekao se utisak o karakteristikama svakog. Izabrali jedan, drugi ili treći, najviše zavisi od toga za šta, odnosno za kakvu aplikaciju se koristi.

Sva tri okvira/biblioteke su zasnovane na komponentama, sva tri okvira imaju dobru dokumentaciju i veliku podršku zajednice, podržane su i od strane velikih kompanija, kao što su **Google**, **Facebook**, i dr.

Odluka za upotrebu **Vue**-a može značiti brz proces učenja, čist kod, brzo renderovanje i malu veličinu, korišćenje šablona. Stoga, za potrebe male aplikacije i samostalnog programiranja, **Vue** može biti prvi izbor.

Angular je u poređenju sa **React**-om i **Vue**-om najteži za učenje, međutim on podržava objektno-orijentisano programiranje pa se može činiti kao primamljiviji izbor. Osim toga, **Angular** se pokazao kao odličan za velike projekte stoga je on popularan u mnogim kompanijama.

React ima ogroman ekosistem što znači da je Internet pun materijala, tutorijala, blogova o **React**-u, što olakšava posao programeru.

Kako je **React** biblioteka, ima podršku velikog broja eksternih biblioteka. Jedna od velikih prednosti **React**-a nad **Angular**-om i **Vue**-om jeste to što može biti namenjen radu u malom timu na malim aplikacijama, a isto tako može uspešno da prati rast aplikacije. Možda je upravo zato, **React** je trenutno, najpopularniji od svih **JavaScript** biblioteka/okvira.

4. LITERATURA

- [1] Fernando Monteiro: „Learning Single-page Web Application Development“, Packt Publishing, 2014
- [2] Philip Klauzinski, John Moore: „Mastering JavaScript Single Page Application Development“, Packt Publishing, 2016
- [3] <https://martinfowler.com/bliki/InversionOfControl.HTML> (pristupljeno u junu 2018)
- [4] <https://reactjs.org/> (pristupljeno u junu 2018)
- [5] <https://angular.io/docs> (pristupljeno u junu 2018)

- [6] Lukas Ruebbelke: „Building Web Applications in Angular 2“, Frontend Masters, 2016
- [7] [HTTps://Vuejs.org/v2/guide/](https://vuejs.org/v2/guide/) (pristupljeno u junu 2018)
- [8] [HTTps://www.stefankrause.net/js-frameworks-benchmark7/table.html](https://www.stefankrause.net/js-frameworks-benchmark7/table.html) (pristupljeno u maju 2018)

Kratka biografija:



Aleksandra Sopka rođena je u Subotici 1993. godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva odbranila je 2016. godine.
Kontakt: aleksandrasopka@gmail.com

OPTIMALNO/AUTOMATSKO PUNJENJE I PRAŽNJENJE SKLADIŠTA ELEKTRIČNE ENERGIJE**OPTIMAL/AUTOMATIC CHARGE AND DISCHARGE OF ENERGY STORAGE**

Lazar Bošnjaković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je razmatran problem smanjivanja pika opterećenja u distributivnoj mreži za unapred zadatu vrednost koristeći skladišta električne energije. Detaljnije su analizirana tri algoritma koja služe za dobijanje optimalnog rasporeda punjenja i pražnjenja skladišta električne energije u toku jednog dana. U cilju rešavanja problema korišćen je jedan od tri analizirana algoritma, adaptivni algoritam. Njegova verifikacija za kriterijum minimalnih gubitaka u toku dana je izvršena na primeru jednostavne distributivne mreže.

Abstract – Peak shaving in distribution networks with energy storages is analyzed in this paper. Three algorithms for obtaining optimal schedule for charging and discharging of energy storages are described and analysed. One of three described algorithms is used for peak shaving in this paper. Verification is performed on a simple distribution network.

Cljučne reči: Distributivna mreža, pik opterećenja, optimizacija, skladišta električne energije.

1. UVOD

Osnovni zadatak koji distributivna kompanija treba da ispuni je kvalitetno i kontinualno snabdevanje potrošača električnom energijom. Jedan od izazova sa kojim se susreće distributivna kompanije je i neravnomerna potrošnja u toku dana. Neravnomerna potrošnja prouzrokuje upotrebu malih elektrana, koje su skupe za upotrebu i predstavljaju trošak za distributivno preduzeće. Zbog toga se razvijaju strategije za smanjivanje pika opterećenja i prebacivanja opterećenja u druge delove dana. Jedna od najpopularnijih strategija je primena skladišta električne energije (SEE) [1]. Ideja koji stoji iza primene SEE je da se ono puni kad je opterećenje malo, a prazni kad je opterećenje veliko, i time smanjuje potrošnju u distributivnoj mreži. SEE mogu biti instalirana u bilo kom podsistemu elektroenergetskog sistema.

2. FORMULACIJA PROBLEMA

Cilj ovog rada je da se predloži algoritam za određivanje optimalnog rasporeda punjenja i pražnjenja SEE sa ciljem da se smanji pik opterećenja uz uvažavanje tehničkih ograničenja same baterije i mreže. Algoritam se svodi na postavljanje i rešavanje optimizacionog problema angažovanja SEE uz uvažavanje tehničkih ograničenja SEE u vidu maksimalnog i minimalnog kapaciteta SEE, kao i maksimalnog stepena brzine punjenja i pražnjenja SEE.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji je mentor bio dr Goran Švenda, red. prof.

Kriterijumska funkcija glasi: Maksimizovati benefit dobijen primenom algoritma smanjivanja pika opterećenja, tačnije maksimizovati razliku između uštede dobijene primenom ovog metoda i troškova primene ovog metoda [3]:

$$\begin{aligned} \text{Benefit}|_{T_e} \\ = \max (\text{Savings}(\text{Size}, \text{Operating schedule})|_{T_e} \\ - \text{Costs}(\text{Size}, \text{Operating schedule})|_{T_e}) \end{aligned}$$

gde je:

Savings – uštede postignute primenom SEE i zavisi od smanjene potražnje električne energije i rasporeda upravljanja SEE.

Costs – troškove ugradnje, upravljanja i održavanja SEE.

Size – kapacitet SEE.

T_e – vremenski period za koji se posmatra optimizacija.

3. KARAKTERISTIKE SKLADIŠTA ELEKTRIČNE ENERGIJE

Najzastupljeniji tip SEE danas predstavljaju elektrohemijska SEE. Postoje tri vrste elektrohemijskih skladišta električne energije [2]:

- Konvencionalne
- Baterije visoke temperature
- Flow baterije

Konvencionalne baterije su sastavljene od ćelija koje sadrže dve elektrode (katoda i anoda) i elektrolita koji se nalaze u zapečaćenoj kućištu. Baterije visoke temperature, ili baterije istopljenih soli, sadrže istopljene elektrode. Kod flow baterija, elektrolit se čuva u posebnom kontejneru odvojen od elektroda.

4. ALGORITMI ZA ANGAŽOVANJE SKLADIŠTA ELEKTRIČNE ENERGIJE

Algoritmi se razlikuju po metodama optimizacije koji se koriste. Te metode su:

- Dinamičko programiranje,
- Bin packing optimizacija
- Adaptivni algoritam.

4.1. Optimizacija – dinamičko programiranje

Prilikom primene ovog algoritma, za promenljivu stanja je uzeta trenutna količina energije u SEE. Količina energije injektirana u mrežu od SEE se može izračunati na osnovu razlike dva uzastopna stanja tj. količine energije sadržane u SEE. Optimizacija ovog problema se vrši za određeni vremenski period, tačnije za vremenski period angažovanja SEE. Tokom ovog perioda je poznata kriva potrošnje, kao jedan od ulaznih parametara optimizacije. Blok dijagram optimizacije problema

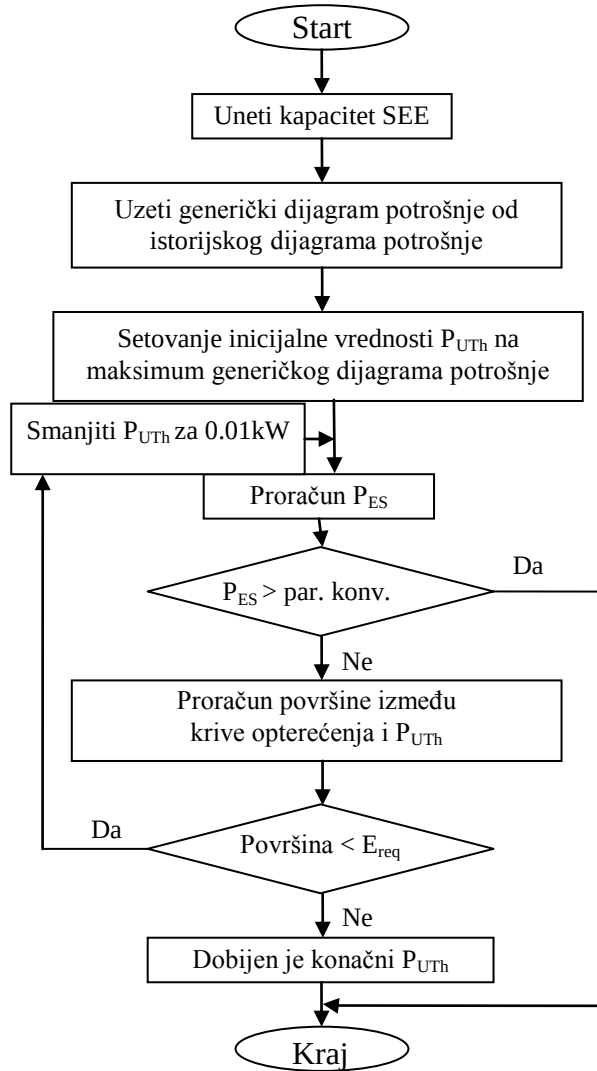
primenom dinamičkog programiranja je prikazan na slici 4.1 [4].

Za ovaj algoritam se kriterijumska funkcija definiše kao:

$$KF = \max \left| \sum_{j=1}^{n_j} [FCOST(PD_j) - FCOST(PD_j - PB_j)] \right| \quad (4.1)$$

4.2. Optimizacija – adaptivni algoritam

Prvi korak primene adaptivnog algoritma predstavlja određivanje granice u potrošnji koja se određuje pomoću algoritma koji analizira krivu potrošnje za posmatrani period optimizacije [5].



Slika 4.2 – Blok dijagram algoritma optimizacije za angažovanje SEE primenom adaptivnog algoritma

Drugi korak adaptivnog algoritma odnosi se na samo određivanje rasporeda punjenja i pražnjenja SEE na osnovu prethodno izračunatog P_{Tuh} . Snaga koja dolazi iz mreže (P_{grid}) jednaka je sumi snaga opterećenja (P_{load}) i snaga SEE (P_{SE}). Odnos te tri snage je:

$$P_{grid} = P_{load} - P_{ES} \quad (4.2)$$

Tipično smanjenje pika potrošnje uključuje samo jedan ciklus punjenja i pražnjenja u toku dana. SEE radi u tri moda: punjenje, pražnjenje i čekanje.

4.2.1. Punjenje SEE

Punjenje se odvija prema sledećoj formuli:

$$P_{ES}(t) = \begin{cases} \text{nominalna brzina,} & \text{za SOC} < 90\% \\ 3[0.7 \ln(t) - 2]kW, & \text{za SOC} \geq 90\% \end{cases} \quad (4.3)$$

4.2.2. Stanje pripravnosti

U stanju pripravnosti je:

$$P_{ES}(t) = 0 \quad (4.4)$$

4.2.3. Pražnjenje SEE

Pražnjenje SEE se odvija po pravilu:

$$P_{ES} = \begin{cases} P_{load}(t) - P_{UTH}(t) \text{ za} \\ P_{load}(t) - P_{UTH}(t) \leq P_{conv} \text{ i SOC} > 50\% \\ P_{conv} \text{ za} \\ P_{load}(t) - P_{UTH}(t) > P_{conv} \text{ i SOC} > 50\% \\ 0 \text{ za} \\ \text{SOC} < 50\% \end{cases} \quad (4.5)$$

4.3. Optimizacija – kombinovan Bin-packin i subset sum algoritam

Prvi korak u optimizaciji je kreiranje histograma od krive potrošnje primenom bin-packing optimizacije. Potrebno je podeliti zadati profil potrošnje u b jednakih vremenskih intervala, tako da zadovoljavaju jednačinu [6]:

$$\sum_{i=1}^b t_b = R \quad (4.6)$$

Drugi korak u optimizaciji je određivanje kapaciteta SEE primenom subset-sum algoritma. Važi sledeći izraz:

$$\sum_{i=1}^k e_i x_i = E \quad (4.7)$$

gde je $x_i = 0$ ili $x_i = 1$. Skup S predstavlja rešenje subset-sum problema ako su zadovoljene sledeće dve jednačine:

$$\begin{aligned} n &\geq |S| \geq (n - \varepsilon) \\ S &= \{e_i \mid \frac{e_i}{t} \leq P_{ESS}\} \end{aligned} \quad (4.8)$$

gde je ε dozvoljena vrednost odstupanja od potrebnog kardinaliteta skupa S , što predstavlja broj dozvoljenih punjenja i pražnjenja u toku jednog ciklusa. Kriterijumska funkcija se predstavlja u sledećem obliku:

$$\begin{aligned} KF &= \min(D_p), \\ D_p &= \max(\{D_1, \dots, D_b\}) \end{aligned} \quad (4.9)$$

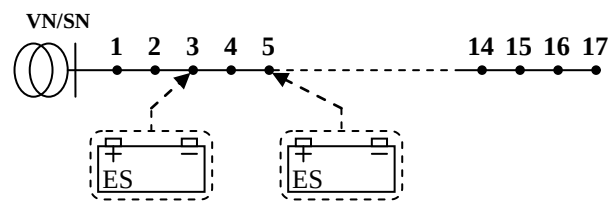
gde je:

D_p – normalizovani pik potrošnje

b – broj diskretnih vremenskih intervala

5. VERIFIKACIJA ALGORITMA

Verifikacija algoritma izvršena je na primeru jednostavne distributivne mreže (DM) koja je prikazana na slici 5.1.



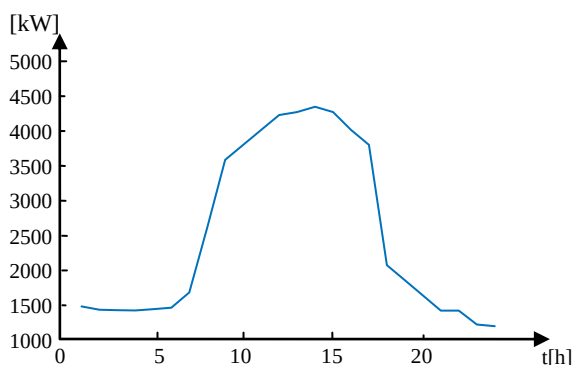
Slika 5.1 – Jednostavna distributivna mreža

Test mreža se sastoji od 17 čvorova i 16 grana. U svakom čvoru se nalazi ekvivalent potrošnje, prikazan kao 24-člani niz u kome je sadržana vrednost potrošnje za dati sat.

Na početku voda se nalazi VN/SN transformator prenosnog odnosa 110/21 kV/kV. SEE se priključuju u svaki čvor i analizira se režim

5.1. Primer 1 – Osnovni slučaj

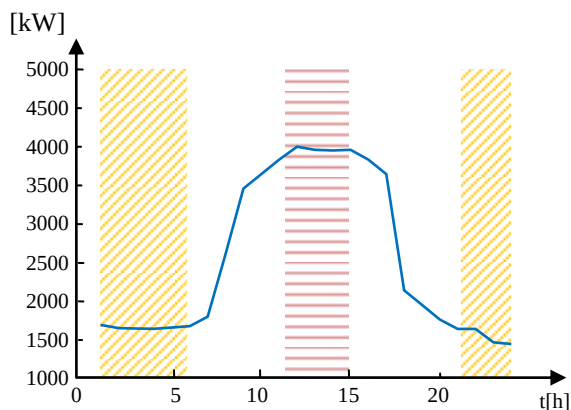
U ovom primeru razmatra se inicijalno stanje test mreže. Test DM ima približno ravnomernu raspodelu potrošnje, tačnije, svih 17 potrošačkih čvorova imaju približno jednake dnevne dijagrame opterećenja (DDO). Na slici 5.2 je prikazana ukupna potrošnja za ceo izvod u toku 24 časa:



Slika 5.2 – Promena ukupne potrošnje u toku dana

5.2. Primer 2 – Efekti jednog SEE

Kroz ovaj primer je urađena analiza delovanja jednog SEE u cilju smanjenja pika opterećenja primenom adaptivnog algoritma.

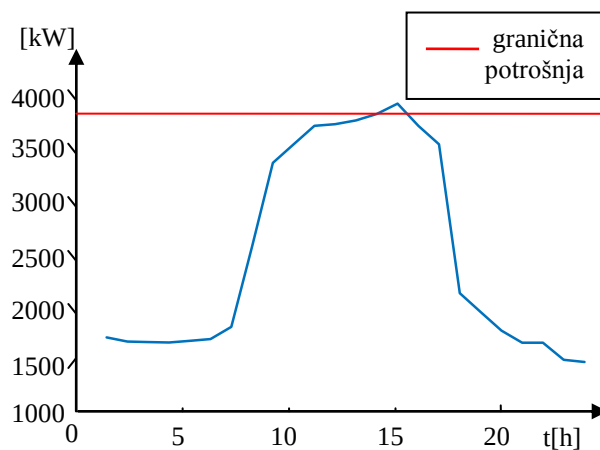


Slika 5.3 – Potrošnja u mreži nakon delovanja SEE

5.3. Primer 3 – Promena granične potrošnje adaptivnim algoritmom

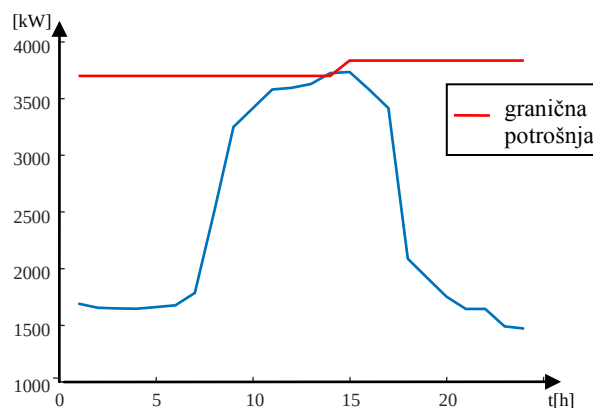
Pomoću ovog primera se pokazuje mogućnost adaptivnog algoritma da menja graničnu potrošnju nakon koje SEE počinje sa pražnjenjem.

Slučaj kad se ne koristi adaptacija granične potrošnje



Slika 5.5 – Kriva potrošnje posle delovanja SEE

Slučaj kad se koristi adaptacija granične potrošnje

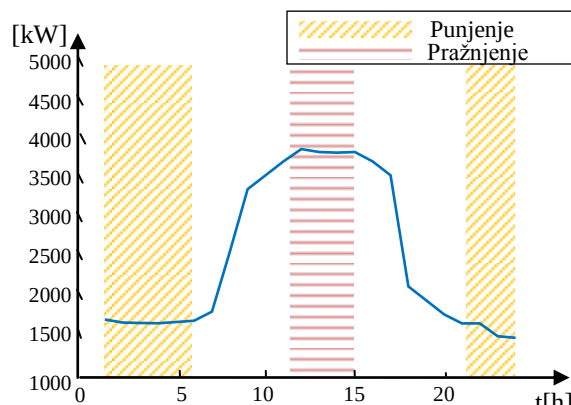


Slika 5.14 – Kriva potrošnje sa primenom promene granične potrošnje

5.4. Primer 4 – Efekti dve istovetne SEE

Kroz ovaj primer je urađena analiza delovanja dva istovetna SEE u cilju smanjenja pika opterećenja primenom adaptivnog algoritma.

Na slici 5.7 je prikazana potrošnja u mreži nakon delovanja SEE.

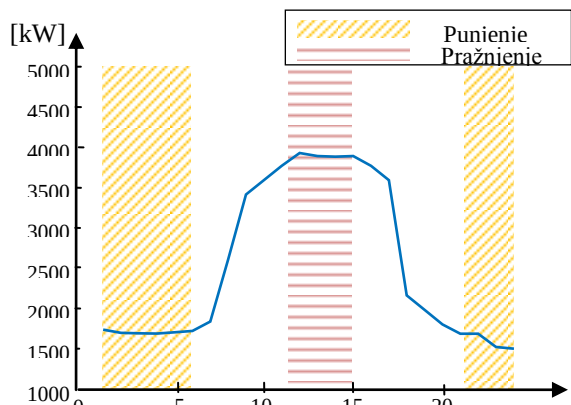


Slika 5.7 – Potrošnja u mreži nakon delovanja SEE

Optimalna pozicija oba SEE bira se na osnovu minimalnih gubitaka aktivne snage u toku dana.

5.5. Primer 5 – Efekti dva SEE različitih kapaciteta

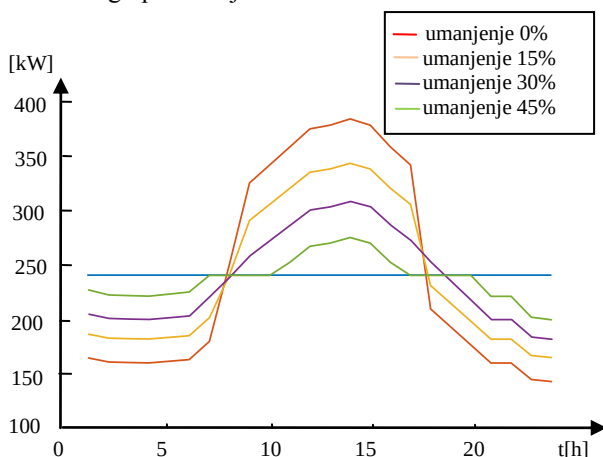
U ovom primeru je obrađen slučaj kada se na izvodu nalaze dva različita SEE, i analiziran je njihov učinak smanjenja pika opterećenja primenom adaptivnog algoritma. Pre određivanja rasporeda punjenja i pražnjenja, određeni su kapaciteti kao i stepeni punjenja i pražnjenja oba SEE u odnosu 7:3.



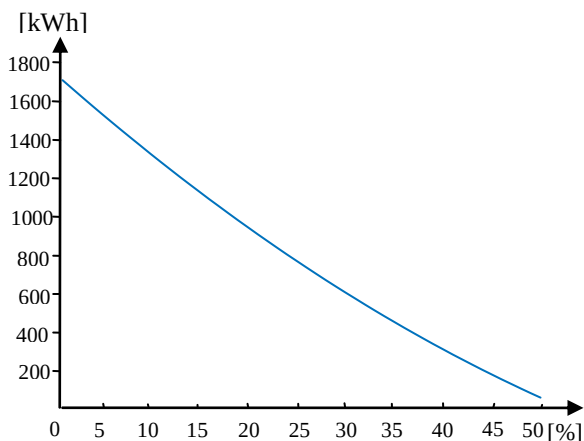
Slika 5.8. Potrošnja u mreži nakon delovanja oba SEE

5.6. Primer 6 – Efekti kapaciteta baterije u zavisnosti od odnosa maksimalnog i minimalnog opterećenja

U ovom primeru je urađena analiza promene kapaciteta baterije u zavisnosti od promene odnosa maksimalnog i minimalnog opterećenja u toku dana.



Slika 5.10. Promena odnosa maksimalne i minimalne potrošnje



Slika 5.11. Promena kapaciteta SEE

6. ZAKLJUČAK

Smanjivanje pika potrošnje u DM donosi višestruke koristi distributivnoj kompaniji. Pravilnim dimenzionisanjem i upravljanjem, smanjuje se potrošnja u distributivnoj mreži za unapred zadatu vrednost, a podiže se u trenucima niske potrošnje za takođe unapred zadatu vrednost. Kroz sve primere je pokazan značaj injektiranja SEE sa ciljem popravljavanja stanja u mreži, kroz smanjivanje potrošnje i smanjivanje ukupnih gubitaka u toku dana. Očekuje se da će buduća istraživanja u ovoj oblasti odnositi na analizu optimalnog broja i kapaciteta SEE za smanjivanje pika opterećenja, kao i studije isplativosti. Takođe, pored kriterijuma minimalnih gubitaka u mreži u toku dana, moguće je tražiti optimalnu poziciju SEE i na osnovu napona potrošača, kao i ograničenja struje u deonicama.

7. LITERATURA

- [1] M.Uddin, M.F.Romlie, M.F.Abdullah, S.Abd Halim, A.H Abu Bakar, T.C. Kwang: A Review on Peak Load Shaving Strategies; *Renewable and Sustainable Energy Reviews*; Vol. 82, Part 3, pp. 3323-3332 February 2018.
- [2] A.Oudalov, R.Charkaoui, A.Beguin: Sizing and Optimal Operation of Battery Energy Storage System for Peak Shaving Application; *Power Tech, 2007 IEEE Lausanne*, Switzerland; 1-5 July 2007;
- [3] D.K.Maly, K.S.Kwan: Optimal Battery Energy Storage System (BESS) Charge Scheduling With Dynamic Programming; *IEE Proc. – Sci. Meas. Technol.* Vol. 142, No. 6, November 1995.
- [4] K.Huat, C.Yun, S.Lim, S.Morris: Energy Storage System for peak shaving, *International Journal of Energy Sector Management*, Vol. 10, Iss 1; February 2016.
- [5] S. Agamah, L.Ekonomou: Peak Demand Shaving and Load-leveling Using a Combination of Bin-packing and Subset sum Algorithms for Electrical energy Storage System Scheduling; *IET Science, Measurement & Technology*, 2016, Vol. 10, Iss. 5, pp. 477-484.

Kratka biografija:



Lazar Bošnjaković rođen je u Novom Sadu 25.08.1992. godine. Gimnaziju "Svetozar Marković" završio je 2011. godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Elektroenergetski sistemi odbranio je 2016. god. Iste godine upisao je master studije.

METODOLOŠKI OKVIR ZA PREDIKCIJU AKVATIČNE VEGETACIJE PRIMENOM VEŠTAČKIH NEURONSKIH MREŽA**FRAMEWORK FOR THE PREDICTION OF AQUATIC VEGETATION USING ARTIFICIAL NEURAL NETWORKS**

Aleksandar Radošević, *Fakultet tehničkih nauka, Novi Sad*
Bojan Damjanović, *Prirodno-matematički fakultet, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Iskopine duž samog toka reke Drine nastaju usled obimne eksploatacije šljunka što rezultuje stvaranju veštačkih jezera. Obzirom da iskopine često nastaju slučajnim odabirom, ovaj rad ima za cilj da doprinese poboljšanju životne sredine i vodene vegetacije planskim iskopavanjem šljunka. Na osnovu dobijenih rezultata je moguće definisati tačan set vrednosti morfometrijskih varijabli koje treba primeniti prilikom kopanja da bi nastala jezera na taj način postala refugijumi, te doprinela diverzitetu glavnog toka reke Drine. U istraživanju su korišćene PCA i CCA analize radi smanjenja skupa podataka koji je prikupljan od 2015. do 2017. godine na veštačkim jezerima. Veštačka neuronska mreža je uzeta za razvoj rešenja, a tačnost rezultata od oko 55% treba da se uzme sa rezervom sa obzirom na mali skup podataka. Uspešnost predviđanja vrsta koje se ne pojavljuju na određenom staništu je oko 80% što može biti od značaja ekologima.

Ključne reči: Istraživanje i analiza podataka, veštačka neuronska mreža, restauracija, vegetacija, Drina.

Abstract – Excavations along the course of the river Drina are due to the extensive exploitation of gravel which results in the creation of artificial lakes. Since excavations are often caused by random selection, project aims improve environment and aquatic vegetation by determining biodiversity friendly excavation procedure. On the basis of the obtained results, it is possible to define the exact set of morphometric variable values to be applied during extraction, so that the lakes formed in this way after their floating, contribute to restoration of the Drina vegetation. During the research, PCA and CCA analysis were used in order to reduce the starting data set, gathered from 2015. until 2017. from artificial lakes. Artificial neural network was used for developing the solution and research results of 55% accuracy should be observed with reserve since small data set was used. However, accuracy for species that should not have been present was around 80% and it could be of significant use to ecologists.

Keywords: Data mining, Artificial neural network, restoration, vegetation, Drina.

1. UVOD

Jezera su često pod uticajem mnoštva različitih faktora, među kojima su hidro-morfologija, eutrofikacija, antropogeni uticaji, način upotrebe zemljišta u riparijalnoj zoni i drugi. Očuvanje akvatičnog bio-diverziteta sa jedne strane zavisi od očuvanja prirodnih vodenih ekosistema, a sa druge strane od formiranja novih alternativnih staništa, kao što su kopovi šljunka u planom području reka. Vodene makrofite predstavljaju važnu komponentu stanišnog diverziteta u takvim ekosistemima. Od kvaliteta i brzine uspostavljanja vegetacijskog pokrivača u novonastalim jezerima, zavisi uspostavljanje i drugih bioloških zajednica. Stoga, mogućnost predviđanja akvatične vegetacije u datim staništima ima veliki aplikativni značaj. Rezultati takvih modela se mogu direktno koristiti prilikom budućih iskopavanja šljunka, u cilju stvaranja jezera što pogodnijih za razvoj bioloških zajednica.

Tokom preliminarog istraživanja koje je sprovedeno u periodu od 2015. do 2017. godine na 14 šljunkara u plavnom području donjeg toka reke Drine, selektovano je nekoliko dominantnih faktora koji utiču na formiranje vodene vegetacije. Tom prilikom su prikupljeni podaci za vodenu vegetaciju, fizičko-hemijske parametre vode i hidro-morfologiju. Potencijal ovog istraživanja jeste u tome da se može doprineti obnovi vodene vegetacije reka nakon iskopavanja veštačkih jezera. Radnicima koji vrše iskopavanje šljunka bi mogli biti dostupni nacrti ili aplikacije, na tabletima, mobilnim uređajima ili nekoj drugoj tehničkoj napravi, koje modeluju morfometriju i rlativan položaj šljunkare u odnosu na glavni rečni tok i druge kopove.

Rad sadrži 5 sekcija. U narednoj sekciji opisana su neka od pretnodnih rešenja koja su korišćena kao smernice prilikom razvoja ovog istraživanja. Treća sekcija posvećena je definiciji problema i metodologiji za njegovo rešavanje. Četvrta sekcija se bavi rezultatima koji su dobijeni u istraživanju sa rekapitulacijom metodologije kao i diskusijom rešenja i poređenja sa prethodnim rešenjima, a peta sumarizacijom rada i pravcima daljeg razvoja.

2. PRETHODNA REŠENJA

Prilikom traženja drugih rešenja na ovu temu, pronađeno je svega nekoliko sličnih studija, što ukazuje na nedovoljnu istraženost ove problematike, te stoga povećava značaj ovog rada. U radu [1] je predložena i opisana veštačka neuronska mreža koja za cilj ima pronalaženje relacije između načina upotrebe zemljišta u

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bi dr Aleksandar Kovačević'.

priobalju i karakteristika vodene vegetacije duž glavnog toka reke Dunav. Problem predstavljen u pomenutom master radu se zasniva na reprezentaciji međuzavisnosti različitih faktora u predikciji vodene vegetacije. U pomenutom master radu izvršena je analiza glavnih komponenti nad ulaznim podacima uz pretpostavku da će se smanjenjem redundantnih podataka poboljšati preciznost neuronske mreže. Rad [2] je fokusiran na modelovanje indeksa makrofitske vegetacije pomoću fizičko-hemijskih parametara kvaliteta staništa primenom veštačkih neuronskih mreža.

Cilj rada je bio da se verifikuje znanje o potencijalnoj primeni makrofita u monitoringu životne sredine. U analizama je korišćena višeslojna neuronska mreža. Uticaj pojedinačnih parametara na svaki model evaluiran je pomoću globalne analize osetljivosti. neuronska mreža primenjena na indeks ekološkog statusa, takozvani MIR indeks je pokazala najbolji rezultat. Rad je pokazao da je modelovanje kvaliteta makrofitskih indeksa na osnovu fizičko-hemijskih parametara moguće uz korišćenje veštačkih neuronskih mreža. U radu [3] ključni element u zahtevima za modelovanje staništa bilo je određivanje i predikcija rasprostranjenja invazivne vrste vodene paprati na osnovu sredinskih atributa.

Korišćene su mašine potpornih vektora (*Support vector machine*, SVM) u kombinaciji sa dva algoritma pretrage [4], koji se koriste radi predikcije rasprostranjenosti invazivne vrste. Prvi algoritam jeste genetski algoritam (*Genetic Algorithm*, GA) [5], a drugi je pohlepni stepen (*Greedy Stepwise*, GS) [6] kako bi odabrali najvažnije eksplanatorne varijable za datu vrstu. Nažalost GA i GS nisu pomogli pri selekciji atributa i doprineli poboljšanju performanse SVM-a. Ipak, GA je prevazišao GS čime je doprineo poboljšanju rezultata. Sve primenjene metode pokazale su da fizičke i fizičko-hemijske karakteristike staništa mogu igrati ključnu ulogu pri uspostavljanju značajne stabilnosti populacije ove invazivne vrste.

3. POSTUPAK RAZVOJA REŠENJA

Podaci su prikupljeni u sklopu izrade doktorske disertacije kolege Bojana Damjanovića, na Prirodno-matematičkom Fakultetu, Departmana za biologiju i ekologiju, a čiji je Doc. dr Dušanka Cvijanović savetnik na doktorskim studijama. Oni su prikupljeni na 14 šljunkara u donjem toku reke Drine u periodu letnjih meseci od 2015. do 2017. godine i tiču se vodene vegetacije, vegetativnog indeksa, fizičko-hemijskih, hidro-morfoloških, hidrološki i bioklimatskih varijabli. Pre obučavanja i testiranja neuronske mreže potrebno je izvršiti pripremu podataka za neuronsku mrežu [7], odnosno, sprovesti određenu obradu kako bi se podaci mogli koristiti kao ulazi u neuronsku mrežu i izlazi iz neuronske mreže.

3.1. Priprema podataka

Nakon konsultacija sa kolegama sa Prirodno-matematičkog Fakulteta, dogovoreno je da će se primeniti neke od metoda poput GLM [8] koji je bio korišćen sa njihove strane radi provere korelacija nakon CCA analize i veštačka neuronska mreža, VNM (*Artificial Neural Networ*, ANN) [7] je uzet za obradu podataka po uzoru na prethodne radove [1,2]. Nakon obrade podataka i smanjenja inicijalnog skupa, napravljen je ekvivalentan skup podataka koji predstavlja normalizovan inicijalni skup. Skup

podataka je bio podeljen na razvojni skup, koji je činio 92% početnog skupa podataka, development skup koji je bio iskorišćen za optimizaciju modela i činio je svega 4% skupa i finalni testni skup na kojem su proveravani dobijeni modeli i činio je 2% početnog skupa podataka.

3.2. Formiranje početnog dvoslojnog modela

Na početku je neuronska mreža bila primenjena na vodenu vegetaciju (procenat vodene flore) kao izlaz, a ulaz je predstavljao čitav skup svih ulaznih vrednosti koji predstavljaju hidrološki, fizičko-hemijski i hidro-morfološki atributi. Kreirana su dva sloja, oba sa osobinama gustog (*Dense*) [9] sloja što znači da svaki ulazni atribut ima istu težinu za izlazni atribut. Svaki sloj pored broja ulaza koji ima, u ovom slučaju 64 obzirom da u njega ulaze sve vrednosti iz skupa podataka, mora da poseduje aktivacionu funkciju koja će da definiše njegovo ponašanje. Odlučili smo se za Rectifier Linear Unit [10] funkciju kao generealno najzastupljeniju funkciju u neuronskim mrežama, a takođe je korišćena u radu [1]. Izlazni sloj je koristio *soft_max* aktivacionu funkciju. Rezultati nisu bili zadovoljavajući, model sa podrazumevanim vrednostima koji je napravljen i primenjen na vodenu vegetaciju kao izlazne parametre i čitav skup ulaznih parametara je imao ogromne gubitke (*loss*) od oko 80% i grešku od oko 90%.

3.3. Redukcija skupa ulaznih vrednosti

Obzirom na loše rezultate koji su se dobili primenom čitavog skupa ulaznih atributa posmatrano na procenat pojavljivanja vodene vegetacije, odlučeno je da se isti redukuje. Pre svega je isprobana hidrologija u rasponu prethodnih 5 godina. Model koji je iskorišćen jeste sa podrazumevanim vrednostima, sa time da je smanjen početni skup ulaznih vrednosti sa 64 na 40. Dobijen je manji pomak u rezultatima, sa 10% tačnosti popelo se na 19%. Model je dalje isproban na fizičko-hemijski sastav i hidro-morfološki sastav koji su dali respektivno 22% tačnosti uz 78% gubitaka i 21% tačnosti uz 80% gubitaka.

3.4. Normalizovanje vrednosti skupa podataka

Nakon istraživanja i poređenja drugih rešenja počevši od rada [1], kao potencijalno poboljšanje je proizašlo to da funkcija *soft_max* bolje funkcioniše ukoliko su joj svi parametri normalizovani, kako ulazni, tako i izlazni. Zbog toga su sada ulazne i izlazne vrednosti bile uzete iz normalizovanog skupa podataka [12]. Za ulazne attribute su iskorišćeni prvo hidrologija tokom svih 5 godina, pri tome je uzet normalizovani skup vrednosti koji se pokazao značajnim nakon CCA analize. Ovoga puta je model sa podrazumevanim vrednostima dao tačnost od 31%, sa oko i dalje visokih 70% gubitaka. Model primenjen na fizičko-hemijski sastav imao je tačnost od 31% uz 70% gubitaka, na hidro-morfološki 34% tačnosti i 65% gubitaka.

3.5. Optimizacija modela

Ubačena su još 2 dodatna sloja, gde je prvi imao identičan broj neurona kao što je broj ulaza, sledeći sloj duplo više neurona, a poslednja dva sloja onaj broj neurona koliko ima izlaznih parametara [13,14]. Takođe nakon empirijskog određivanja vrednosti parametara sa različitim epohama, došlo se do zaključka da se povećanjem broja epoha sa 10 na raspon od 150 do 180, model daje najbolje rezultate. Ispod toga bi modeli veoma

često bili slabo obučeni da bi mogli da postignu očekivane rezultate [15]. Ovakav način strukturiranja neuronske mreže odrađen je u skladu sa preporukama iz radova koji su se bavili sličnom tematikom. Najbolji rezultat je dao fizičko-hemijski sastav sa čak 48% tačnosti.

3.6. Finalni model

Finalni model se sastojao od 4 sloja, pri čemu je prvi sloj imao duplo veći broj neurona u odnosu na ulaz, drugi sloj duplo više u odnosu na prvi, a poslednja dva sloja 3 puta više neurona nego što je broj ulaznih parametara, a ovakva struktura slojeva VNM je uzeta po uzoru na rad Sanskruti Patel [16]. Kao ulazni atribut primenjen je fizičko-hemijski sastav normalizovanog skupa podataka, na četvoroslojni model sa 150 epoha i optimizovanim slojevima, koji se opet ukazao najboljim sa do sada najvećih 55% tačnosti, uz 9% gubitaka. Hidro-morfološki sastav normalizovanog skupa podataka kao ulazni atribut dao je tačnost od 52% uz 8% gubitaka. Najslabiji rezultat je dala redukovana hidrologija normalizovanog skupa podataka tokom svih 5 godina kao ulazni skup atributa od 49% i 10% gubitaka.

4. REZULTATI I DISKUSIJA

Rezultati su dobijeni nakon eksperimentisanja sa modelima uz oslonac na neke od prethodnih radova radi poboljšanja modela VNM. U tabeli 1 je prikazana sumarna tabela svih rezultata koji su postignuti tokom istraživanja, sa time da je četvoroslojni model sa 150 epoha sa optimizovanim slojevima primenjen na normalizovani skup podataka vodene vegetacije kao izlazne attribute i fizičko-hemijski sastav kao ulazne attribute dao najbolji rezultat od 55%.

Tabela 1. Rezultati primene četvoroslojnog modela sa 150 epoha na normalizovani skup podataka sa vodenom vegetacijom kao izlaznim atributima

	Gubitak modela	Procenat tačnosti modela
Hidrologija tokom 5 godina	10%	49%
Fzičko-hemijski	9%	55%
Hidro-morfološki	8%	53%

Svi rezultati su izraženi procentualno

Tabela 2 predstavlja procenat tačnosti predikcije modela na vrednosti koje se neće pojaviti u odnosu na različite ulazne vrednosti. Model koji je bio korišćen jeste četvoroslojni sa 160 epoha i optimizovanim slojevima na normalizovani skup podataka uz vodenu vegetaciju kao izlazni skup atributa.

Tabela 2. Procenat tačnosti predikcije vrsta koje se neće pojaviti korišćenjem četvoroslojni model sa 160 epoha sa optimizovanim slojevima primenjen na normalizovani skup podataka sa vodenom vegetacijom kao izlaznih atributa

	Gubitak modela	Procenat tačnosti modela
Hidrologija tokom 5 godina	10%	70%
Fzičko-hemijski	9%	80%
Hidro-morfološki	8%	73%

Nakon analize podataka, empirijske upotrebe raznih metoda, algoritama i drugačijih kako izlaza tako i ulaza atributa za model, najveći efekat je dobijen normalizacijom ulaza i izlaza, nakon toga povećanjem broja epoha i finalno upotrebom fizičko-hemijskog sastava kao ulaznog atributa i vodene vegetacije kao izlaznog atributa u četvoroslojnom modelu. To govori da sama hidrologija, plavljenja i padavine ne utiču značajno na formiranje vegetacije u jezerima, već veći uticaj imaju fizičko-hemijska svojstva vode. Slične vrednosti su dobijene korišćenjem hidro-morfološkog skupa varijabli što dodatno ukazuje na to da sam oblik i morfologija jezera imaju ogroman uticaj na formiranje akvatične zajednice.

Ono što jeste značajan rezultat ovog istraživanja, jeste da se prilikom testiranja rezultata dobijala tačnost od preko 80% za 0 vrednosti. One su se nalazile u izlaznom skupu atributa vodene vegetacije, pri čemu predstavljaju vrste koje nisu prisutne. To znači da se sa relativnom tačnošću mogla odrediti predikcija biljne vrste koja se neće pojaviti na osnovu ulaznih atributa. To ukazuje da se uz još malo kvalitetnih uzoraka može postići veoma dobra predikcija i samih biljnih vrsta vodene vegetacije.

Pre svega treba sagledati neke od problema na koje se naišlo tokom istraživanja. Sam set podataka koji je dobijen nije imao mnogo korelacija između ulaza, te se moralo pribеći empirijskim rešenjima. Vrlo brzo je ustanovljeno da će glavni limitirajući faktor dolaska do bilo kakvih signifikantnih rešenja biti pomenuti mali broj podataka. Uzimajući u obzir ovo, dobijeno rešenje je na kraju uspelo da ispuni neka od početnih očekivanja. Takođe, ogromnu prepreku predstavljaju i uzorci, koji su vrlo skupi za dobavljanje, nije moguće tražiti neki drugi sličan set podataka radi poređenja, obzirom da nisu javno dostupni niti su vršena slična istraživanja. Pored svega toga, svaki izlazak na teren po nove podatke je veoma skup, te nije lako ni dopuniti trenutni set podataka.

Mogući izvor greške u finalnim rezultatima modela može biti i sam način uzorkovanja vegetacijskih snimaka. Naime, snimci (uzorci) se uzimaju tako što se na već predefinisanoj površini popisuju sve prisutne vrste i svakoj od njih se pridružuje odgovarajuća ocena za pokrovnost. Skala za procenu pokrovnosti vrste koja je korišćena u istraživanjima sprovedenim na šljunkarama je trostepena, gde se daju sledeće ocene: 1 - za vrstu koja je prisutna do pokrovnosti 0-25%; zatim ocena 2 - za vrste sa pokrovnosću od 25 do 75%; i potom vrste sa > 75% pokrovnosti dobijaju ocenu 3. Ove vrednosti vrsta su iskorišćene za računanje relativne pokrovnosti svake vrste po sektoru (1 sektor = do 100 vegetacijskih snimaka/uzoraka), što je predstavljalo zapravo izlaze u analizama.

5. ZAKLJUČAK

Tema i problem koji su obrađeni u ovom radu predstavljaju pokušaj predviđanja pojavljivanja vodene vegetacije u jezerima donjeg toka reke Drine, koji su nastali kao posledica iskopavanja šljunka za potrebe eksploatacije istog. Uzorci koji su bili korišćeni u eksperimentu su podaci prikupljeni od 2015. do 2017. godine, na jezerima koja su se nasumično pojavljivala i nestajala tokom prikupljanja podataka. Pripremljen skup je dalje normalizovan i podeljen na tri dela, razvojni

(92%), optimizujući (5%) i testirajući (5%). Isprobane su razne vrste modela i njihovih metoda uz pomoć Keras biblioteke, pri čemu je postignut najbolji rezultat primenom čitavog skupa fizičko-hemijskih karakteristika vode kao ulaza na čitav skup procenata pojavljivanja vodene vegetacije. Tačnost koja je dobijena je 55%, što ako se uzme u obzir da se radi o skupu podataka od 62 torke, i da šum koju entropija ekoloških procesa može da proizvede zbog svoje nepredvidivosti, i veoma visoke cene uzoraka, može da se smatra dobrim rezultatom. Rad sam po sebi ima veliki potencijal da značajno doprinese poboljšanju rečnog biodiverziteta svuda u Evropi. S obzirom na to da je koncept neuronskih mreža relativno nov pojam, njihova primena za određene grane nauke još uvek nije u potpunosti realizovana.

Na osnovu prethodno navedenog, postoji potreba za unapređenjem samog skupa podataka, dodatna istraživanja i pretrage na samoj oblasti i terenu bi značajno doprinela poboljšanju rezultata. Postoji šansa da bi se otkrile nove veze unutar skupa podataka, te bi se možda dobila nova obeležja koja bi bila od presudnog značaja.

Ako se uzme neka od stvari koje su one otkrile poput zavisnosti vrsta i šta najviše utiče na njihov procenat pojavljivanja, moguće je primeniti drugi pristup rešenju kao što su korišćenje programa R ili RapidMiner-a, koja bi mogla doprineti poboljšanju rezultata ili pak otkrivanju nekih novih zavisnosti i poboljšanja modela.

ZAHVALNICA

Posebnu zahvalnost zbog pripreme skupa podataka dugujem Bojanu Damnjanoviću, doktorantu sa Prirodno-matematičkog fakulteta, Departman za biologiju i ekologiju.

LITERATURA

- [1] Kaplar Aleksandar (2015). "Primena veštačkih neuronskih mreža u analizi visoko dimenzionalnih skupova podataka". Master rad.
- [2] Daniel Gebler, Dariusz Kayzer, Krzysztof Szoszkiewicz Anna Budka. "Artificial neural network modelling of macrophyte indices based on physico-chemical characteristics of water" <https://link.springer.com/article/10.1007/s10750-013-1585-7>
- [3] Roghayeh Sadeghi, RahmatZarkamii Patrick Van Damme. "Modelling habitat preference of an alien aquatic fern, *Azolla filiculoides* (Lam.), in Anzali wetland (Iran) using data-driven methods". <http://www.sciencedirect.com/science/article/pii/S030438014001847>
- [4] Algoritam pretrage https://en.wikipedia.org/wiki/Search_algorithm
- [5] Genetski algoritam https://www.researchgate.net/publication/234136753_The_variability_of_macrophyte_metrics_used_in_river_monitoring
- [6] Pohlepni algoritam https://www.researchgate.net/publication/234136753_The_variability_of_macrophyte_metrics_used_in_river_monitoring
- [7] Kriesel David. A Brief Introduction to Neural Networks. http://www.dkriesel.com/_media/science/neuronale-netze-en-zeta2-2col-dkrieselcom.pdf
- [8] Generalized linear model https://en.wikipedia.org/wiki/Generalized_linear_model
- [9] Keras <https://keras.io/>
- [10] Makrofitni indeks https://www.researchgate.net/publication/234136753_The_variability_of_macrophyte_metrics_used_in_river_monitoring
- [11] Why is normalization necessary in Artificial Neural Networks (ANN) https://www.researchgate.net/post/Why_normalization_is_necessary_in_ANN
- [12] Jelena Jovanović. Mašinsko učenje http://ai.fon.bg.ac.rs/wp-content/uploads/2016/10/ML_intro_2016.pdf
- [13] Nachimuthu Karunanithi, William J. Grenney, Member, ASCE, Darrell Whitley and Ken Bovee. Neural Networks for River Flow Prediction. (1994): 1943-5487
- [14] Gail A. Carpenter, Sucharita Gopal, Scott Macomber, Siegfried Martens, Curtis E. Woodcock, Janet Franklin. A Neural Network Method for Efficient Vegetation Mapping. Remote Sensing of Environment vol 70. (1999)
- [15] Over-fitting i under-fitting <https://chemicalstatistician.wordpress.com/2014/03/19/machine-learning-lesson-of-the-day-overfitting-and-underfitting/>
- [16] Sanskruti Patel. "An Empirical Study of Applying Artificial Neural Network for Classification of Dermatology Disease". Article (2017) https://www.researchgate.net/publication/317613284_An_Empirical_Study_of_Applying_Artificial_Neural_Network_for_Classification_of_Dermatology_Disease

Kratka biografija:



Aleksandar Radošević rođen je 29.04. 1994. godine u Novom Sadu. Osnovnu školu „Đorđe Natošević“ u Novom Sadu završio je 2009. godine. Gimnaziju „Svetozar Marković“ u Novom Sadu završio je 2013. godine. Iste godine upisao se na Fakultet tehničkih nauka, odsek Računarstvo i automatika. Osnovne studije je završio 2017. godine, nakon čega upisuje master akademske studije na Fakultetu tehničkih nauka, smer Softversko inženjerstvo i informacione tehnologije. Položio je sve ispite propisane planom i programom. Kontakt : aleksandarradošević1994@gmail.com



Bojan Damnjanović rođen je u Šapcu 1987. godine. Master rad pod nazivom "Ekološka studija reke Štira primenom RHS metode", odbranio je 2013. godine na Prirodno-matematičkom fakultetu, Univerziteta u Novom Sadu. Doktorand je na Prirodno-matematičkom fakultetu, Univerziteta u Novom Sadu od 2013. godine, studentski program doktor nauka - ekološke nauke. Od 2014. je u zvanju asistenta u nastavi strukovnih studija na Visokoj medicinskoj i poslovno - tehnološkoj školi strukovnih studija u Šapcu. Kontakt: bdamnjanovic@live.com

POREĐENJE PERFORMANSI TEHNIKA INDEKSIRANJA U RELACIONIM BAZAMA PODATAKA**COMPARISON OF INDEXING TECHNIQUES PERFORMANCES IN RELATIONAL DATABASES**

Marko Vještica, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu predstavljene su različite tehnike indeksiranja u relacionim bazama podataka. Nad kreiranom šemom baze podataka vršena su testiranja performansi izvršenja različitih upita i analizirani su dobijeni rezultati. Šema baze podataka projektovana je uz pomoć Oracle SQL Developer Data Modeler softverskog alata, a za njenu implementaciju korišćen je Oracle Database 11g Enterprise Edition.

Ključne reči: Tehnike indeksiranja, optimizator upita, poređenje performansi izvršenja upita

Abstract – Different indexing techniques in relational databases are presented in this paper. On a created database schema, performances of different queries are tested and the results analyzed. The database schema is developed using Oracle SQL Developer Data Modeler case tool, and is implemented on Oracle Database 11g Enterprise Edition.

Keywords: Indexing techniques, query optimizer, comparison of query execution performances

1. UVOD

Performanse izvršavanja upita i ažuriranja podataka oduvek su bili predmet diskusije u sistemima baza podataka koji obuhvataju veliku količinu podataka. Potrebno je balansirati različitim parametrima, jer često poboljšanjem performansi izvršavanja upita, dolazi do gubitka performansi ažuriranja podataka i obrnuto.

Uvođenjem dodatnih struktura koje pomažu u brzem traženju i pretraživanju podataka ili grupisanju i izvršavanju određenih funkcija, povećava se i kapacitet potrebne memorije na disku.

Iz tog razloga važno je odabrati odgovarajuću strukturu koja će doprineti poboljšanju performansi neke od operacija, a pri tome neće veoma opteretiti memoriju sistema.

Data Warehouse sistemi, koji su sve zastupljeniji, obuhvataju velike količine podataka koji mogu i ne moraju biti agregirani. Pojedini upiti u takvim sistemima ne izvršavaju se u željenom vremenu. Zbog toga je potrebno obezbediti način na koji bi se takvi upiti izvršavali u okviru zadovoljavajućeg vremena.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila dr Slavica Kordić, docent.

Jedan način kako je moguće poboljšati performanse pojedinih upita je da se obezbedi pristupni put za brže traženje ili pretragu podataka, što je moguće postići kreiranjem odgovarajućih indeksnih struktura. Indeksi su pomoćne pristupne strukture čija je namena da ubrzaju dobijanje podataka u zavisnosti od uslova pretrage [1].

Moguće je koristiti različite tehnike indeksiranja za određeni upit. Potrebno je odrediti koji indeks pokazuje dobre performanse prilikom izvršavanja upita. Cilj ovog rada je da izvršenjem niza testova bude ustanovljeno koji indeksi pružaju odgovarajuće performanse za različite upite. Korišćen je Oracle sistem za upravljanje bazom podataka (SUBP) za potrebe testiranja, kao jedan od najčešće upotrebljivanih SUBP-ova u praksi.

Osim Uvoda i Zaključka, ovaj rad sadrži još tri poglavlja. U poglavlju *Tehnike indeksiranja* predstavljene su različite tehnike indeksiranja u relacionim bazama podataka, kao i kratak pregled tehnika indeksiranja u Oracle SUBP-ovima. U poglavlju *Postupak testiranja performansi izvršenja upita* opisana je računarska konfiguracija nad kojom je vršeno testiranje i način izvršenja testova. U poglavlju *Testiranje i poređenje performansi izvršenja upita* predstavljeni su rezultati izvršenih testiranja. Ovaj rad obuhvata deo rezultata koji su prezentovani u master radu.

2. TEHNIKE INDEKSIRANJA

Organizacija datoteke je način uređivanja slogova u datoteci kada je ona smeštena na disku. Postoji više načina na koje je moguće pristupiti slogovima u datoteci, a usled različitih mogućih načina uređivanja datoteke, dobija se različita efikasnost određenih operacija nad tom datotekom [2]. Ukoliko su slogovi u datoteci uređeni po određenom kriterijumu i postavljeni upit sadrži upravo taj kriterijum, tada u većini slučajeva dolazi do smanjenja vremena odziva na upit.

Indeksne strukture su dodatne datoteke na disku koje obezbeđuju druge načine pristupa slogovima, ali tako da ne utiču na fizičku poziciju slogova u datoteci na disku. Oni omogućavaju efikasan pristup slogovima zasnovan na indeksiranim poljima koja su korišćena za kreiranje indeksa. U okviru jedne datoteke može postojati više indeksa koji su zasnovani nad različitim poljima, a takođe moguće je da pojedini indeksi budu zasnovani nad više polja.

U nastavku opisan je deo tehnika indeksiranja za potrebe ovog rada. Među njih spadaju: jednonivoovski uređeni indeksi, višenivoovski indeksi, bitmap indeksi, spojni indeksi i indeksi zasnovani na funkciji.

2.1. Jednonivoovski uređeni indeksi

Vrednosti jednonivoovskih indeksa uređeni su kako bi bilo moguće vršiti binarnu pretragu indeksa i time dodatno smanjiti vreme potrebno kako bi slogovi bili pronađeni. Tu spadaju primarni, klasterovani i sekundarni indeksi. Primarni indeks je uređena datoteka kreirana nad primarnim ključem sa jednim poljem uređene datoteke sa podacima. Klasterovani indeks je takođe uređena datoteka, ali za razliku od primarnog indeksa, moguće je imati iste vrednosti indeksiranog polja. Sekundarni indeks omogućava sekundarni pristup datoteci sa podacima za koju već postoji primarni pristup, stoga ona ne mora biti uređena.

2.2. Višenivoovski indeksi

Ideja višenivoovskih indeksa je da bude ubrzana binarna pretraga indeksa prilikom izvršavanja odgovarajućeg upita. To je moguće postići time što su višenivoovski indeksi organizovani kao stabla. Time je smanjen broj pristupa blokovima na disku prilikom pretrage slogova. Stablo pretrage može biti korišćeno kao mehanizam za pretragu slogova koji su smešteni u datoteci na disku. Svaka indeksirana vrednost polja ima pokazivač ka slogu u datoteci sa podacima ili ka bloku na disku. Ideja je da stablo ima što je moguće manju visinu, da je balansirano, da ne zahteva velike promene i rekonstrukciju prilikom dodavanja ili brisanja slogova iz datoteke podataka, da su čvorovi što je moguće više popunjeni kako bi bio smanjen broj nivoa stabla i kako bi bila ostvarena ušteda na prostoru. Sve te probleme pokušava da reši B-stablo. Da bi sve to bilo obezbeđeno, algoritmi za dodavanje i brisanje slogova postaju kompleksniji. Svaki unutrašnji čvor B-stabla ima q pokazivača ka podređenim čvorovima i $q-1$ pokazivača ka slogovima koji sadrže date indeksirane vrednosti. Svi listovi su na istom nivou u stablu, odnosno B-stablo je balansirano. B^+ -stablo je varijacija B-stabla sa dodatnim ograničenjima koja doprinose poboljšanju performansi izvršavanja upita i ažuriranja podataka. Osnovna razlika je u tome što listovi sadrže slogove za svaku vrednost indeksiranog obeležja, zajedno sa pokazivačem ka odgovarajućem slogu u datoteci sa podacima. Listovi su često povezani pomoću dodatnog pokazivača, koji pokazuje ka narednom ili prethodnom listu, kako bi bila omogućena brža redosledna obrada.

2.3. Bitmap indeksi

Bitmap indeksi često se koriste u relacijama koje imaju veliki broj slogova. Indeksirana je svaka vrednost ili opseg vrednosti jednog ili više obeležja. Za svaku vrednost indeksiranog obeležja kreiran je niz bitova dužine n , gde n predstavlja broj slogova u relaciji. Ukoliko neko obeležje ima m različitih vrednosti, onda je kreirano m bitmap nizova za to obeležje. Svaki niz indeksiranog obeležja ima vrednost 1 na i -toj poziciji ukoliko i -ti slog ima vrednost obeležja koja odgovara tom nizu. U suprotnom, na i -toj poziciji nalazi se vrednost 0.

2.4. Spojni indeksi

Spojni indeksi sadrže n vrednosti, gde n predstavlja broj spojenih relacija, a svaki slog u indeksu predstavlja jednu spoenu kombinaciju identifikatora slogova iz različitih relacija. Korišćenjem ovih indeksa, nema potrebe za

spajanjem relacija, nego je dovoljno preuzeti identifikatore slogova koji se nalaze u indeksu.

2.5. Indeksi zasnovani na funkcijama

Indeksi zasnovani na funkciji koriste se kada su često postavljani upiti koji zahtevaju izvršavanje određene funkcije nad velikom količinom podataka, što može dugo da traje. Tada je efikasniji način kreirati indeks nad rezultatima te funkcije, što znači da će za svaki upit biti pretražen indeks, umesto što će biti izvršena funkcija koja je zahtevana u upitu. Ideja je da bude iskorišćen indeks, a ne da uvek mora biti izvršeno pretraživanje cele tabele i računanje rezultata funkcije.

2.6. Oracle tehnike indeksiranja

Današnji Oracle SUBP-ovi podržavaju širok spektar različitih tehnika indeksiranja. Oracle server može da iskoristi indekse kako bi brže pristupio podacima ili kako bi izvršio proveru važenja ograničenja podataka. Oracle SUBP-ovi podržavaju razne tehnike indeksiranja, a za potrebe ovog rada izdvojene su sledeće: indeksi sa B-stablom, bitmap indeksi, bitmap spojni indeksi i indeksi zasnovani na funkciji [3, 4].

3. POSTUPAK TESTIRANJA PERFORMANSI IZVRŠENJA UPITA

Prilikom testiranja tehnika indeksiranja korišćena je Oracle Database 11g Enterprise Edition baza podataka. Korišćeni operativni sistem je Microsoft Windows 10 64-bit. U nastavku predstavljena je hardverska konfiguracija računara na kojem je vršeno testiranje: procesor - Intel(R) Core(TM) i7-6700HQ CPU @ 2.60 GHz, 4 Cores, 6 MB Cache; RAM memorija - 16 GB DDR4, 2133 MHz; grafička kartica - NVIDIA GeForce GTX 960M, 4 GB DDR5; disk - 1 TB HDD, 7200 rpm.

Pomoću SQL*Plus korisničkog interfejsa vršena su sva testiranja u Oracle bazi podataka, iz kojeg su pozivane kreirane skripte sa test slučajevima. U svakoj skripti je pre Select naredbe, koja predstavlja test slučaj, navedena *alter system flush buffer_cache* komanda, pomoću koje su pre svakog poziva Select naredbe ispražnjeni podaci iz bafera u operativnoj memoriji. Time je postignuto da je svaka Select naredba izvršena nad podacima koji se nalaze na disku. Prilikom izvršavanja Select naredbe, potrebno je pratiti vreme trajanja izvršavanja upita i plan izvršenja upita. U SQL*Plus-u moguće je pratiti vreme trajanja izvršenja pojedinačne naredbe uz pomoć *set timing on* komande. Moguće je pratiti vreme izvršavanja upita do nivoa 10ms. Upotrebom *set autotrace traceonly* komande omogućen je prikaz plana izvršenja upita i statistika, a zabranjuje se ispis rezultata izvršenja upita. Time je rešen i problem ispisa velike količine podataka na terminal, jer to dodatno usporava vreme potrebno za izvršenje upita. U komentaru Select naredbe moguće je napisati koji indeks je potrebno iskoristiti prilikom izvršenja upita, što će optimizator upita u većini slučajeva i učiniti.

4. TESTIRANJE I POREĐENJE PERFORMANSI IZVRŠENJA UPITA

Na osnovu prethodno opisanog postupka, izvršena su testiranja performansi prilikom izvršavanja upita. Na taj

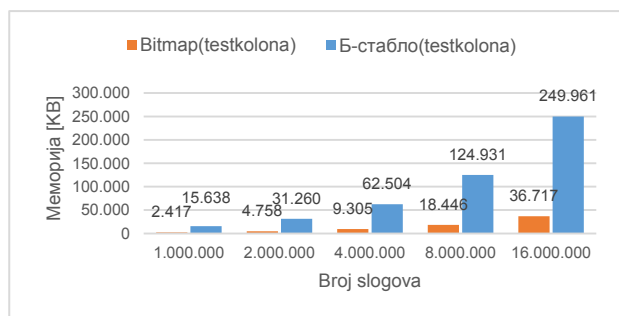
način moguće je pokazati kako se određene tehnike indeksiranja zaista pokazuju u praksi.

Testovi su podeljeni u tri grupe na osnovu indeksa koji su korišćeni prilikom testiranja performansi izvršenja upita. Vršene su sledeće grupe testova: poređenje performansi indeksa sa B-stablom i bitmap indeksa, poređenje performansi indeksa zasnovanih na funkciji i poređenje performansi bitmap spojnih indeksa.

4.1. Poređenje performansi indeksa sa B-stablom i bitmap indeksa

Testiranja su vršena nad tabelom koja nema implementirano ni jedno ograničenje. Ona sadrži kolonu predviđenu za primarni ključ tabele, jednu test kolonu i 10 pomoćnih kolona kako bi bila obezbeđena velika količina podataka za potrebe testiranja.

Grafikon 5.1 predstavlja memoriju na disku koju indeksi zauzimaju u odnosu na povećanje broja slogova u tabeli.



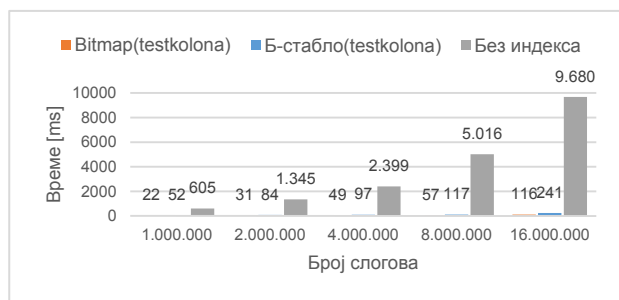
Grafikon 5.1 – Memorija na disku koju zauzimaju indeksi

Količina memorije koju zauzme indeks sa B-stablom značajno je veća u odnosu na bitmap indeks. To je posledica što bitmap indeks čuva nizove bitova za svaku jedinstvenu vrednost obeležja. Indeks sa B-stablom u proseku zauzima 6,67 puta više memorije u odnosu na bitmap indeks.

Grafikon 5.2 prikazuje vreme potrebno za izvršenje upita sa COUNT funkcijom i tri OR operatora u uslovu upita:

`SELECT COUNT(*) FROM tabela WHERE testkolona = 40 OR testkolona = 60 OR testkolona = 80;`

Upoređena su vremena prilikom korišćenja obe vrste indeksa i u slučaju kada nije korišćen nijedan indeks.



Grafikon 5.2 – Vreme potrebno za izvršenje upita sa COUNT funkcijom i OR operatorima

Oba indeksa pokazuju bolje performanse u odnosu na slučaj bez korišćenja indeksa. Bitmap indeks je u proseku bio brži 2,24 puta od indeksa sa B-stablom. To je posledica prednosti bitmap indeksa prilikom korišćenja COUNT funkcije u kombinaciji sa bitwise operatorima. Dovoljno je iskoristiti bitwise operatore nad nizovima bitmap indeksa i prebrojati jedinice u rezultujućem nizu.

Kako raste broj slogova u tabeli, tako su obe vrste indeksa pokazivale sve bolje performanse u odnosu na slučaj bez korišćenja indeksa. Slične performanse pokazane i sa SUM funkcijom, kada je sumirano indeksirano obeležje. Upiti sa funkcijama MAX i MIN imali su nešto kraće vreme izvršavanja prilikom korišćenja indeksa sa B-stablom u odnosu na bitmap indekse.

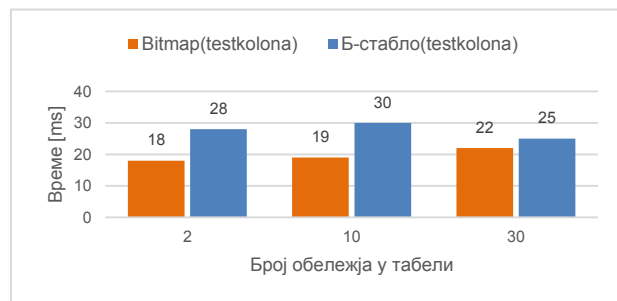
Na osnovu izvršenih testova, pokazano je da povećanje broja obeležja u tabeli ima skoro linearan uticaj na povećanje vremena izvršenja upita ukoliko nije kreiran nijedan indeks (grafikon 5.3). Testovi su izvršeni pomoću upita sa COUNT funkcijom i operatorom jednakosti u uslovu Select naredbe:

`SELECT COUNT(*) FROM tabela WHERE testkolona = 40;`



Grafikon 5.3 – Odnos vremena izvršenja upita prilikom povećanja broja obeležja u tabeli bez korišćenja indeksa

Prilikom izvršavanja testova sa istim upitom, ukoliko su kreirani indeksi nad tabelom, pokazano je da nema uticaja povećanje broja obeležja u tabeli. To je slučaj pod uslovom da je Select naredba izvršena samo nad indeksom, bez potrebe prolaska kroz tabelu (grafikon 5.4).



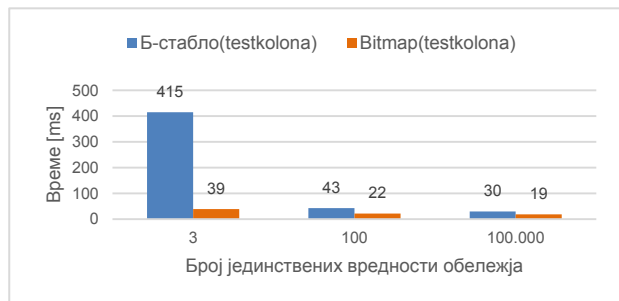
Grafikon 5.4 – Vreme potrebno za izvršenje upita u zavisnosti od promene broja obeležja u tabeli

Takođe je na osnovu testova pokazano da povećanjem broja slogova u tabeli postoji gotovo linearan rast vremena izvršenja upita bez korišćenja indeksa. Rezultati testova sa SUM funkcijom i operatorom jednakosti u uslovu Select naredbe pokazali su da prilikom povećanja broja slogova 2 puta, vreme izvršenja upita bude povećano u proseku 2,01 puta (grafikon 5.5).



Grafikon 5.5 – Odnos vremena izvršenja upita prilikom povećanja broja slogova u tabeli bez korišćenja indeksa

Testiran je i slučaj kada je povećavan broj jedinstvenih vrednosti indeksiranog obeležja, čime je povećavana selektivnost podataka. To svakako doprinosi indeksima sa B-stablom, dok bitmap indeksima više odgovara niska selektivnost podataka jer je potrebno napraviti manje nizova jedinstvenih vrednosti. Međutim, bitmap indeksi takođe pokazuju bolje performanse, jer *Oracle* SUBP smešta bitmap indeks u strukturu B-stabla (grafikon 5.6).



Grafikon 5.6 – Vreme potrebno za izvršenje upita prilikom promene broja jedinstvenih vrednosti obeležja

4.2. Poređenje performansi indeksa zasnovanih na funkciji

Test slučaj korišćen za ovu vrstu indeksa je upit sa *COUNT* funkcijom u *Select* naredbi i uslovom koji sadrži funkciju *POWER* i unutar nje funkciju *MOD*:

```
SELECT COUNT(*) FROM tabela WHERE POWER (MOD (testkolona, 11), 2) = 49;
```

Rezultati testiranja pokazuju da kreiranjem indeksa nad ugnježdenom funkcijom *MOD* značajno doprinosi izvršenju upita jer je SUBP u mogućnosti da iskoristi taj indeks prilikom izračunavanja spoljne funkcije *POWER*. Kada su indeksirane obe funkcije zajedno, pokazane su još bolje performanse jer nije bilo potrebe za izračunavanjem funkcije *POWER*, već je dovoljan prolazak kroz kreirani indeks. Prilikom testiranja kreirani su bitmap indeksi zasnovani na funkciji (grafikon 5.7).

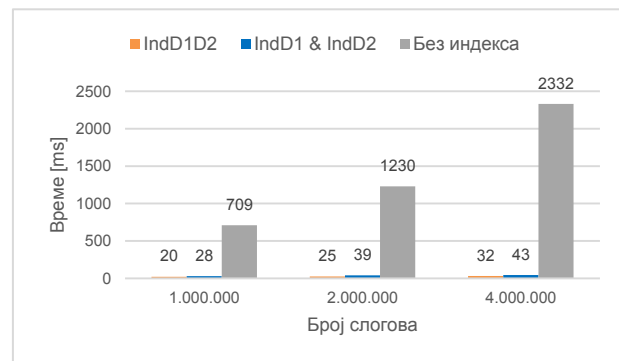


Grafikon 5.7 – Vreme potrebno za izvršenje upita sa ugnježdenom funkcijom

4.3. Poređenje performansi bitmap spojnih indeksa

Za potrebe testiranja kreirane su tri tabele kako bi bilo moguće simulirati zvezdastu šemu u *Data Warehouse* sistemima. Tabela nad kojom su kreirani indeksi referencira na preostale dve tabele. Kreiran je jedan kompozitni bitmap spojni indeks nad dva obeležja gde se svaki nalazi u jednoj referenciranoj tabeli, kao i dva odvojena bitmap spojna indeksa za svako obeležje ponaosob. Upit sadrži u *Select* naredbi funkciju *COUNT* i uslove spajanja navedene tri tabele. Dve tabele koje simuliraju dimenzione tabele sadrže u sebi fiksni broj slogova od 100.000, dok

je broj slogova u trećoj tabeli manjan. Testiran je slučaj bez korišćenja indeksa, zatim sa korišćenjem kompozitnog spojnog indeksa i sa korišćenjem oba pojedinačna spojna indeksa. Rezultati pokazuju slične performanse prilikom korišćenja indeksa (grafikon 5.8).



Grafikon 5.8 – Vreme potrebno za izvršenje upita sa bitmap spojnim indeksima

5. ZAKLJUČAK

Rezultati testiranja pokazali su prednost bitmap indeksa u odnosu na indekse sa B-stablom u *Oracle* SUBP-ovima. U većini agregacionih funkcija, bitmap indeksi pokazali su bolje performanse, posebno prilikom korišćenja *bitwise* operatora. Takođe, prednost bitmap indeksa je i u malom zauzeću memorije na disku, o čemu treba brinuti, iako je u današnje vreme relativno niska cena fizičke memorije.

Korišćenjem indeksa zasnovanih na funkciji, moguće je bitno doprineti performansama izvršenja upita, ukoliko su često korišćene određene funkcije. Svakako je pozitivno to što je moguće iskoristiti deo funkcija ukoliko postoje ugnježdene funkcije u upitu.

Dalje istraživanje prvenstveno bi bilo usmereno na spojne indekse. Rezultati pokazuju slične performanse izvršenja upita prilikom korišćenja kompozitnog indeksa i pojedinačnih indeksa. Potrebno je ustanoviti kakav uticaj na performanse navedenih indeksa ima povećanje broja referenciranih tabela ili povećanje broja slogova u referenciranim tabelama, kao i povećanje broja indeksiranih obeležja u okviru indeksa.

LITERATURA

- [1] Ramez Elmasri, Shamkant B. Navathe: Database Systems: Models, Languages, Design, and Application Programming, Sixth Edition, 2011.
- [2] Raghu Ramakrishnan, Johannes Gehrke: Database Management Systems, Second Edition, 2000.
- [3] Priya Vennapusa: Oracle Database 10g: SQL Tuning, Volume I, Student Guide, Edition 2.0, decembar 2006.
- [4] Oracle Database Concepts, Indexes and Index-Organized Tables, dostupno na: <https://docs.oracle.com/cloud/latest/db112/CNCPT/indexiot.htm#CNCPT721> (31. januar 2018.)

Kratka biografija:



Marko Vještica rođen je 9. jula 1994. godine u Novom Sadu. Školske 2013/2014. godine upisuje Fakultet tehničkih nauka u Novom Sadu, studentski program Računarstvo i automatika. Osnovne akademske studije završio je 2017. godine.
kontakt: marko.vjestica@uns.ac.rs

PRIMENA PAYPAL SISTEMA ZA INTERNET KUPOVINU POLISA OSIGURANJA APPLICATION OF PAYPAL FOR ONLINE PURCHASE OF INSURANCE POLICIES

Teodora Mitrović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – U radu je analiziran PayPal sistem za Internet plaćanje. Opisan je sistema za Internet prodaju polisa osiguranja. U okviru ovog sistema projektovan je i implementiran modul za plaćanje uz oslonac na PayPal.

Ključne reči: PayPal, elektronsko plaćanje, polise osiguranja

Abstract – The paper gives overview of the PayPal system for Internet payments. The system for online sale of insurance policies is described. Within this system, a payment module based on PayPal has been designed and implemented.

Keywords: PayPal, electronic payment, insurance policy

1. UVOD

Elektronska trgovina se odnosi na online poslovne aktivnosti. Poslovna transakcija može obuhvatati naručivanje, odabir dobara, fakturisanje kao i plaćanje. Transakcije ovog tipa izvršavaju se bez potrebe za čekovima ili gotovinom. Sa razvojem elektronske trgovine, razvoj sistema elektronskog plaćanja je bio neminovan. PayPal je jedan od vodećih predstavnika sistema elektronskog plaćanja. Preko PayPal sistema se novac može slati, primiti ili stavljati na račun. Korisnik postaje deo PayPal sistema [1] otvaranjem svog naloga. Nalozi mogu biti lični (*personal*), napredni (*premier*) i poslovni (*business*). Nalozi obezbeđuju različite funkcionalnosti, odnosno koriste se za različite namene [1].

Sistem ima politiku [2] koja štiti kupce i prodavce. Kupac se štiti u smislu da ima pravo na prigovor na transakciju ukoliko ne primi ono što je platio. Ukoliko se prigovor prihvati, kupcu se novac refundira. Prodavac se najviše štiti od lažnih prigovora. PayPal naplaćuje proviziju za transakcije koje obrađuje. Provizija se uzima za primanje novca, dok je slanje novca besplatno. Procenat zavisi od tipa naloga, valute transakcije, iznosa transakcije, zemlje pošiljaoca, zemlje primaoca, načina plaćanja, itd. [3].

2. APLIKACIJA ZA PRODAJU OSIGURANJA

Informacioni sistem za podršku rada osiguravajuće kuće (ISOK) se bavi prodajom putnog osiguranja. Jedan njegov deo predstavlja aplikacija za internet prodaju osiguranja (AIPO). Takođe, AIPO sadrži podsistem koji predstavlja koncentrator plaćanja. Poslovanje osiguravajuće kuće podrazumeva prodavanje polisa osiguranja.

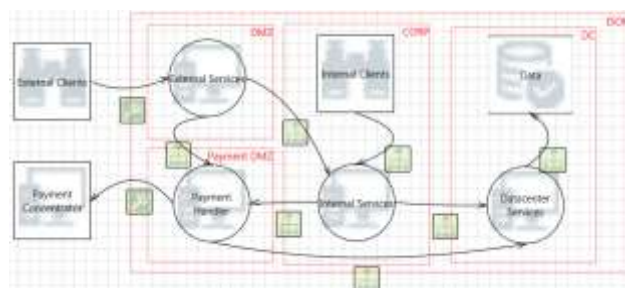
NAPOMENA:

Ovaj rad proistekao je iz master rada čiji je mentor dr Goran Sladić, vanr. prof.

Polisa osiguranja [4] je ugovor između mušterije koja kupuje polisu i lica koje prodaje polisu. U zamenu za isplatu polise osiguranja, lice koje izdaje polisu obećava da će snositi posledice realizacije rizika, u smislu isplaćivanja novčane naknade koja je dogovorena prilikom sklapanja ugovora. Koncentrator plaćanja predstavlja odvojen sistem koji posreduje između servisa za plaćanje (poput PayPal servisa i servisa banke) i preduzeća koja žele da omoguće različite vidove elektronskog plaćanja prilikom prodaje svojih proizvoda.

AIPO se sastoji od nekoliko podsistema koji omogućavaju poslovanje osiguravajuće kuće (slika 1). To su: sajt za prodaju polisa osiguranja uz podrškom online plaćanja, aplikacija za upravljanje cenama i stavkama cenovnika, aplikacija za prodaju polisa u poslovnica, i slično. Izdvojene su tri bezbednosne zone:

- **DMZ zona**, koja nudi servise za klijente koji dolaze sa interneta. Ova zona je veza spoljašnje mreže sa unutrašnjim delom sistema;
- **Payment DMZ zona** (zona plaćanja), gde se nalazi komponenta koja posreduje u komunikaciju između eksternog servisa koncentratora plaćanja i AIPO servisa;
- **CORP zona** predstavlja korporativnu mrežu, odnosno deo sistema koji je instaliran na računarima zaposlenih u poslovnica osiguravajuće kuće;
- **DC zona** je data centar i obuhvata baze podataka kao i komponente sistema koje obezbeđuju njihovu bezbednu upotrebu.



Slika 1. Dijagram toka podataka AIPO sistema

3. MODUL ZA PLAĆANJE

Opisani sistem je realizovan uz praćenje MVC šablona [5]. Za implementaciju opisanog sistema se koristi programski jezik Java (verzija 1.8) [6], pri čemu se realizacija oslanja na Spring okruženje (verzija 1.5.8.) [7]. Kao razvojno okruženje korišćen je Spring Tool Suite [8] i IntelliJ [9]. Svaka od komponenti predstavlja zaseban projekat. Sve komponente sistema se pokreću na Apache Tomket [10] serveru sa izuzetkom aplikacije za podršku razmene email-ova.

Za email server je korišćen besplatni hMailServer [11], koji ima podršku za IMAP, POP3 i SMTP email protokole. Kao klijent email servera je korišćen php paket SquirrelMail [12]. Komunikacija između komponenti je realizovana preko REST protokola [13]. Svi zahtevi i odgovori u ovoj komunikaciji imaju format JSON fajlova, te se za mapiranje JSON [14] objekata koristi Jackson biblioteka [15].

Autentifikacija i Single Sign On [16] funkcionalnost su prepušteni KeyCloak [17] sistemu. Kontrola pristupa obezbeđena od strane KeyCloak-a je zasnovana na ulogama. Stoga je KeyCloak u implementiranom sistemu zadužen za Single Sign On i deo funkcionalnosti koje se tiču upravljanja ulogama, dok je upravljanje permisijama implementirano u zasebnoj administratorskoj aplikaciji. Autorizacija uz proveravanje permisija prijavljenog korisnika je realizovana pomoću Spring interceptor-a.

Za skladištenje podataka je odabrana MySQL baza (verzija 5.7) [18], dok je objektno-relaciono mapiranje prepušteno Hibernate [19] sistemu. U cilju pojednostavljenja razvoja klasa u sloju modela sistema, korišćena je Javina biblioteka Lombok [20].

Kada je u pitanju korisnički interfejs, za implementaciju veb sajtova je korišćen JavaScript okvir AngularJS [21]. Sve veb stranice su implementirane uz praćenje SPA (Single Page Application) arhitekture.

Uzevši u obzir kompleksnost sistema, kao i činjenicu da podrazumeva novčane transakcije, posebna pažnja posvećena je i logovanju relevantnih događaja u sistemu. Logovanje je realizovano uz pomoć Logback okvira [22].

Sa obzirom na veliku potrebu za upotrebom PayPal-a u informacionim sistemima, razvijeni su načini za povezivanje sa PayPal sistemom [23]. Neki od načina za integraciju su:

- PayPal Core API
- PayPal Mobile API
- Website Payments Standard

Korisnici sa ličnim računima nemaju podršku za rad sa PayPal API-jem, dok je za poslovne naloge obezbeđena. Za pristup API-ju neophodni su API kredencijali, koji se sastoje od: API korisničkog imena i API lozinke.

PayPal REST API, ima dva režima rada: produkcion (live) režim i režim testiranja (sandbox). Režim testiranja omogućava testiranje rada aplikacije u toku razvoja. Da bi aplikacija mogla da komunicira sa API-jem, mora biti registrovana, što znači da mora je neophodan poslovni nalog. Alternativa je koristiti tesne naloge koji se kreiraju za korisnika onog trenutka kada on napravi svoj lični nalog na PayPal-u.

Za potrebe razvoja i testiranja aplikacija koje se oslanjaju na PayPal sistem obezbeđeno je okruženje za razvoj i testiranje, tzv. sandbox okruženje [24]. Prelazak sa sandox okruženja na produkciono okruženje podrazumeva samo promenu konfiguracionih parametara.

U okviru konfigurisanja korisničkog naloga moguće je registrovati aplikaciju, pod određenim imenom, koja će koristiti usluge PayPal-a. PO registraciji aplikacije, ona dobija svoje kredencijale na osnovu kojih se autentifikuje prilikom komunikacije sa API-jem (slika 2).

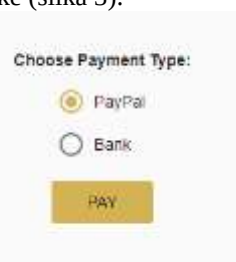


Slika 2. Kredencijali registrovane aplikacije

Da bi sistem mogao da komunicira sa PayPal REST API-jem, neophodno je najpre uključiti PayPal SDK [25] kao zavisnost. U slučaju AIPO sistema, Koncentrator je aplikacija koja komunicira sa PayPal sistemom, te se SDK uključuje u zavisnosti Koncentratora. Da bi bilo moguće autentifikovanje aplikacije, u konfiguraciju Koncentratora se moraju uključiti i kredencijali koje je aplikacija dobija po registraciji.

Proces kupovine polise osiguranja preko javnog portala informacionog sistema osiguravajuće kuće podrazumeva 4 koraka. U prvom koraku se unose datumi početka i završetka putovanja, broj putnika za svaku od starosnih kategorija, region destinacije kao i iznos na koji se polisa izdaje. Takođe, u prvom koraku se naglašava da li će se putnici baviti nekim od sportova. Ukoliko je to slučaj, unosi se i sport. U drugom koraku korisnik unosi podatke nosioca polise kao i ostalih putnika. Za nosioca polise je obavezno uneti email adresu, zbog toga što će se njemu, po završetku plaćanja, poslati polisa u pdf-formatu. Treći i četvrti korak podrazumevaju unos podataka vezanih za dodatke polisi kao što su osiguranje kuće/stana ili osiguranje vozila. Ovi koraci su opcioni.

U petom koraku se bira način plaćanja polise. Korisnik može da odabere između plaćanja preko PayPal-a i plaćanja preko banke (slika 3).



Slika 3. Odabir načina plaćanja.

Klikom na dugme Pay, objekat polise osiguranja, sa sračunatom finalnom cenom posle popusta, šalje se u data centar (Datacenter) gde je onda pohranjen. Po uspešnom skladištenju polise, šalje se zahtev za izvršenje transakcije pozivom *sendPaymentInquiry* metode *PayPalService* klase (lisitng 1).

Metodi *sendPaymentInquiry* se prosleđuju podaci kao što je iznos koji se plaća kao i podaci o kupcu osiguranja. Ova metoda pravi objekat *payment* klase *com.paypal.api.payments.payment.Payment* u koji se postavljaju informacije o tome ko je kupac, u kojoj se valuti plaća, transakcije, kao i opis. Takođe, postavlja se i *intent* plaćanja, što je u ovom konkretnom slučaju prodaja te se intent setuje na *sale*. *Payment*-u se, u sistemu kada je to

potrebno, mogu postaviti vrednosti poreza, naziv države, čitava elektronska korpa sa podacima o proizvodima, plan naplaćivanja itd. *Payment* objektu se postavljaju i redirekcionirajući linkovi, odnosno putanje do metoda koje će API pozvati onda kada se transakcija izvrši uspešno ili kada izvršavanje transakcije bude neuspešno.

```
@Service
public class PayPalServiceImpl implements PayPalService {
    ...

    @Override
    public PaymentInquiryInfoDTO
    sendPaymentInquiry(PaymentInquiryDTO piDTO,
        HttpServletRequest request) {

        String cancelUrl = createCancelURL(request);
        String successUrl = createSuccessURL(request);
        Amount amount = new Amount();
        amount.setCurrency("USD");
        amount.setTotal(piDTO.getAmount());
        Transaction transaction = new Transaction();
        transaction.setAmount(amount);
        List<Transaction> transactions = new
        ArrayList<>();
        transactions.add(transaction);

        Payee payee = new Payee();
        payee.setEmail(piDTO.getCustomer());

        Payer payer = new Payer();
        payer.setPaymentMethod(
            PayPalPaymentMethod.paypal.toString());

        Payment payment = new Payment();
        payment.setIntent(PaypalPaymentIntent.sale.
            toString());
        payment.setPayer(payer);
        payment.setTransactions(transactions);
        RedirectUrls redirectUrls = new
        RedirectUrls();
        redirectUrls.setCancelUrl(cancelUrl);
        redirectUrls.setReturnUrl(successUrl);
        payment.setRedirectUrls(redirectUrls);

        Payment p = payment.create(apiContext);
        String approval_url = createApprovalURL(p);

        String[] parts = approval_url.split("token=");
        String token = parts[1];
        PaymentInquiryInfoDTO piInfoDTO = new
        PaymentInquiryInfoDTO();
        piInfoDTO.setPaymentId(token);
        piInfoDTO.setPaymentUrl(approval_url);
    }
}
```

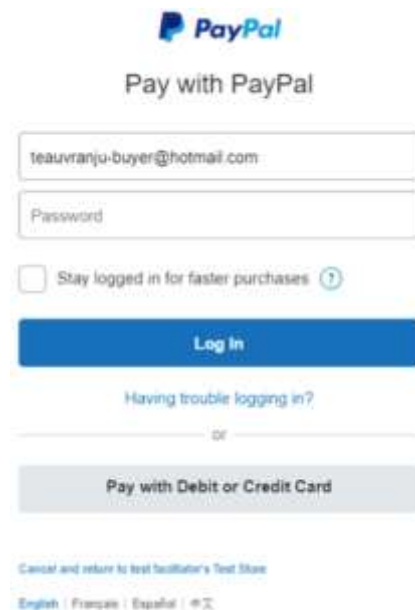
Listing 1. Metoda *sendPaymentInquiry*

Sa pozivom metode *create* nad objektom *payment*, šalje se zahtev PayPal REST API-ju za kreiranje plaćanja u ime kupca čija je email adresa prosleđena kroz objekat *payment*.

Metodi *create* se ovom prilikom prosleđuje objekat klase, koji sadrži kredencijale aplikacije na osnovu kojih će se sistem autentifikovati API-ju.

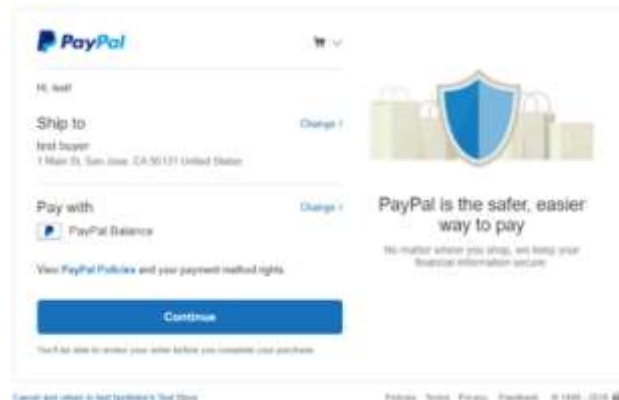
Odgovor na poslato zahtev sadrži informacije o kreiranoj transakciji (ukoliko je sve prošlo bez problema) kao i link na kom korisnik mora da potvrdi plaćanje da bi se transakcija obavila.

Sistem potom redirektuje korisnika koji kupuje polisu osiguranja na link za odobravanje transakcije. Ukoliko korisnik nije ulogovan na PayPal sistem, PayPal-ov API redirektuje korisnika na *login* (slika 4). Od tog trenutka, pa do uspešnog (ili neuspešnog) izvršenja transakcije, korisnik komunicira sa PayPal sistemom.



Slika 4. Autentifikacija na PayPal sajtu

Po uspešnom logovanju, korisniku se prikaže njegov račun i omogući mu se da potvrdi transakciju (slika 5).



Slika 5. Potvrda transakcije

Korisnik potvrđuje izvršenje transakcije i kada PayPal-ov API obradi transakciju, poziva jedan od linkova koji su mu prosleđeni kroz *payment* objekat. Odnosno, ukoliko je transakcija uspešno izvršena, API šalje HTTP zahtev na *success_url*, dok se u suprotnom zahtev šalje na *cancel_url*.

Od tog trenutka, kontrola je predata informacionom sistemu osiguravajuće kuće, i podaci o uspešnoj ili neuspešnoj transakciji se skladište u bazu posmatranog sistema.

Po završenoj obradi transakcije od strane AIPO-a, sistem redirektuje korisnika na stranicu sa obaveštenjem o uspešnosti plaćanja.

4. ZAKLJUČAK

Sa razvojem elektronske trgovine, razvoj sistema elektronskog plaćanja je bio neminovan. Elektronski platni sistemi su zasnovani na informaciono-komunikacionim tehnologijama. Koriste se za transfer novca, pri čemu se njihova primena ne ograničava samo na elektronsku trgovinu, već prožima sfere međubankarskog plaćanja, homebanking-a, plaćanja debitnim, trgovinskim, kreditnim karticama, itd.

PayPal je online sistem koji okuplja mušterije i trgovce, i pojedincima i kompanijama omogućava da transfere sredstava obavljaju elektronski. Danas je PayPal zastupljen u preko 200 zemalja sveta i podržava preko 20 različitih valuta. Ovaj sistem daje kredibilitet malim prodavcima, sa obzirom na to da kupci imaju više poverenja u prodavca iza kog stoji pouzdan sistem za naplatu.

U ovom radu dat je predlog upotrebe PayPal sistema za online kupovinu polisa osiguranja. Predloženo rešenje je demonstrirano na procesu kupovine polise korišćenjem veb aplikacije javnog portala AIPO-a. U datom rešenju korišćen PayPal REST API za pristup PayPal servisima.

U trenutnoj implementaciji sistema, nedostaje podsistem za podršku rada mobilnih korisnika. Unapređenja sistema bi mogla biti implementacija mobilne aplikacije, pri čemu bi se za plaćanje koristio MobileSDK. Drugo unapređenje se može bazirati na različitim planovima plaćanja, izdavanju faktura i slično.

5. LITERATURA

- [1] "PayPal" Internet: <https://www.paypal.com/rs/home>
- [2] "PayPal – Safety and Security" Internet: <https://www.paypal.com/rs/webapps/mpp/paypal-safety-and-security>
- [3] "PayPal Fees" Internet: https://www.paypal.com/rs/webapps/mpp/ua/useragreement-full#exhibit_A
- [4] "Insurance policy" Internet: <https://thelawdictionary.org/insurance/>
- [5] "MVC design pattern" Internet : https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm
- [6] "Java" Internet : <https://www.java.com/en/>
- [7] "Spring Boot" Internet : <https://spring.io/projects/spring-boot>
- [8] "Spring Tool Suite" Internet : <https://spring.io/tools/sts>
- [9] "IntelliJ" Internet : <https://www.jetbrains.com/idea/>
- [10] "Apache Tomcat" Internet : <http://tomcat.apache.org/>

- [11] "hMail Server" Internet : <https://www.hmailserver.com/>
- [12] "Squirrel Mail" Internet : <https://squirrelmail.org/about/>
- [13] "REST Protocol" Internet : <https://www.w3.org/2001/sw/wiki/REST>
- [14] "JSON" Internet: <http://www.json.org/>
- [15] "Jackson Java JSON-processor" Internet: <http://jackson.codehaus.org/>
- [16] "Single Sign On" Internet : <http://www.opengroup.org/security/sso/>
- [17] "KeyCloak" Internet : <https://www.keycloak.org/>
- [18] "MySQL" Internet : <https://www.mysql.com/>
- [19] "Lombok" Internet : <https://projectlombok.org/>
- [20] "Hibernate" Internet : <http://hibernate.org/>
- [21] "AngularJS" Intertnet : <https://angularjs.org/>
- [22] "Logback" Internet : <https://logback.qos.ch/>
- [23] "PayPal Developer API Overview" Internet: <https://developer.paypal.com/docs/api/overview/>
- [24] "PayPal Sandbox" Internet: <https://www.sandbox.paypal.com/rs/home>
- [25] "PayPal Developer – REST API SDKs" Internet: <https://developer.paypal.com/docs/api/rest-sdks/>

Kratka biografija:

Teodora Mitrović je rođena 04.8.1994. godine u Vranju. Osnovnu školu „Vuk Karadžić“ završila je 2009. godine. Gimnaziju „Bora Stanković“ u Novom Sadu završila je 2013. godine. Osnovne akademske studije na Fakultetu tehničkih nauka završila je 2017. godine na smeru Računarstvo i automatika, studentski program Primenjene računarske nauke i informatika. 2017. godine upisuje master akademske studije na Fakultetu tehničkih nauka u Novom Sadu. Iste godine se zapošljava kao saradnik u nastavi na Fakultetu tehničkih nauka, Katedri za informatiku.

IZRAČUNAVANJE CENA POLISA OSIGURANJA POMOĆU SISTEMA BAZIRANIH NA PRAVILIMA**CALCULATION OF INSURANCE POLICY PRICES USING RULES-BASED SYSTEMS**

Mina Medić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu su analizirani sistemi bazirani na pravilima sa osvrtom na Drools sistem, kao i sistemi elektronskog poslovanja. Specificirana je i implementirana aplikacija za podršku rada osiguravajuće kuće, sa akcentom na modul za kalkulaciju cena polisa osiguranja.

Ključne reči: Ekspertni sistemi, Sistemi bazirani na pravilima, Drools, Elektronsko poslovanje

Abstract – The paper analyzes the rule-based systems with focus at the Drools system. Also, an overview of electronic business systems is given. An application for supporting the operations of the insurance company has been designed and implemented, with emphasis on the module for calculating the insurance policy prices.

Keywords: Expert systems, Rule based systems, Drools, Electronic Business

1. UVOD

Elektronsko poslovanje je opšti koncept koji obuhvata sve oblike poslovnih transakcija ili razmene informacija koje se izvode korišćenjem informacione i komunikacione tehnologije i to: između organizacija, između organizacija i njihovih klijenata i između organizacija i javne administracije [1]. Ubrzana razmena informacija dovela je do kvalitetnih promena u upravljanju poslovanjem i odnosu prema konkurenciji. Takođe, dolazi i do povećanje tražnje za uslugama kao i ekspanzije elektronske trgovine putem Interneta [2].

Oblast elektronskog poslovanja, zbog svoje dinamičnosti i stalnog razvoja, zahteva primenu brojnih alata. Ovi alati moraju biti u mogućnosti da u kratkom vremenskom periodu obrade velike količine podataka koje generišu ovakvi sistemi i omoguće da se dobijeni rezultati pretoče u kvalitetne poslovne odluke.

Ekspertni sistemi zasnovani na pravilima (eng. *RBS – rule based systems*) mogu da pruže značajan doprinos brzini i kvalitetu donošenja poslovnih odluka u elektronskom poslovanju. Povezivanjem ove dve oblasti otvaraju se nove mogućnosti za poboljšanjem kvaliteta ovih sistema. Mnogi poslovni sistemi danas koriste sisteme bazirane na znanju kako bi izvršili automatizaciju procesa.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Goran Sladić, vanr. prof.

Sistemi zasnovani na pravilima koriste pravila, kao najopštiju paradigmu za definisanje zakona, kako bi opisali ponašanje sistema, specifične algoritme upravljanja i odlučivanja. Pisana u prosto formi *uslov/akcija* pravila omogućuju izvršavanje akcija u onom obliku u kom bi to radio i čovek. Kao softverski sistem koji je logički odvojen od aplikativnog koda, ekspertni sistemi zasnovani na pravilima omogućuju korisnicima lako rukovanje, izmenu i dodavanje pravila vezanih za samo poslovanje.

U okviru rada razvijena je web aplikacija za Internet prodaju polisa osiguranja. S obzirom na veliki broj faktora koji mogu uticati na cenu polise, razvijen je poseban modul koji na osnovu sistema baziranih na pravilima definiše krajnju cenu iste.

2. OBLASTI ISTRAŽIVANJA

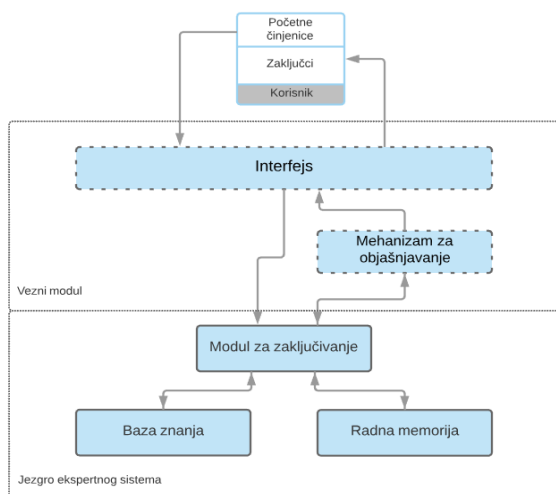
Ključne oblasti razmatrane za potrebe specifikacije i implementacije sistema za kalkulaciju cena polisa osiguranja su ekspertni sistemi i elektronsko poslovanje.

2.1. Ekspertni sistemi

Ekspertni sistemi (ES) su nastali u okviru veštačke inteligencije i dobro su istraženi kao oblast, što se vidi i iz definicija – relativno su konzistentne i konvergiraju ka istom pojmu. Neke od definicija ES su:

- “Vrsta programa koja donosi odluke ili rešava probleme iz određene oblasti korišćenjem znanja i analitičkih pravila koja su definisali eksperti za datu oblast” [3].
- “Ekspertni sistem je računarski program koji je projektovan tako da oponaša sposobnost rešavanja problema eksperta (čoveka) za neku oblast” [4].
- “Ekspertni sistem je računarski program koji koristi ugrađeno znanje iz neke specifične oblasti i donosi zaključke na osnovu njega, a u cilju rešavanja problema ili pružanja saveta” [5, 6].

Prethodne definicije sugerišu da ekspertni sistemi predstavljaju programe koji koriste proces zaključivanja (rezonovanja) sličan ljudskom kod rešavanja problema iz raznovrsnih domena. Ovo zaključivanje zasniva se na ekspertskom ljudskom znanju, koje je kodirano u program u vidu posebne strukture koja se naziva baza znanja. Poseban mehanizam koristi raspoloživo znanje u procesu zaključivanja, sa ciljem da dođe do rešenja problema. Na taj način rešavaju se problemi koji su izvan domašaja konvencionalnih računarskih programa. Osnovna struktura ekspertnog sistema može se predstaviti slikom 1.



Slika 1. Osnovna struktura ekspertnog sistema

Baza znanja (knowledge base) ES služi za skladištenje domenskog znanja, pri čemu je ono izraženo korišćenjem neke metode za predstavljanje znanja [6]. *Radna memorija* (working memory) ES simulira kratkoročnu memoriju eksperta. Tu se skladište početne činjenice o problemu i svi zaključci do kojih ES dođe u radu [6]. *Mehanizam za zaključivanje* (inference engine) uparuje činjenice iz radne memorije sa znanjem iz baze znanja korišćenjem neke metode za zaključivanje i izvodi zaključke. Izvedeni zaključci se skladište u radnoj memoriji [6]. *Interfejs* (korisnički) i *mehanizam za objašnjavanje* (explanation facility) su opcion i služe za povezivanje eksperta i inženjera znanja sa bazom znanja.

S obzirom da eksperti najčešće svoje znanje predstavljaju u obliku pravila, ona su jedna od najčešće korišćenih metoda za predstavljanje znanja u okviru ES. U skladu sa tim razvijena je posebna vrsta ekspertnih sistema – sistemi bazirani na pravilima. Teorijska zasnovanost ove metode počinje na *iskaznoj logici* i na njenom proširenju – *predikatskom računu*. Pravila su, u stvari, pojednostavljena računarska implementacija predikatskog računa, a jedna od osnovnih tehnika zaključivanja je *modus ponens* [6]. Osnovna forma pravila je sledeća:

$$\text{pravilo: (premisa)} \rightarrow (\text{zaključak}) \quad (1)$$

gde (*premisa*) predstavlja formulu koja mora biti zadovoljena kako bi se pravilo aktiviralo, a (*zaključak*) čini akciju koja će se izvršiti na uspešnu aktivaciju pravila. Pravilo može da se sastoji iz više permisa i tada se one povezuju odgovarajućim logičkim operatorima I (AND), ILI (OR) i NE (NOT).

2.2. Elektronsko poslovanje

Danas e-poslovanje ostvaruje ekspanzivan rast. U dosadašnjem periodu e-poslovanje je doživelo posebnu ekspanziju u maloprodaji, izdavaštvu i u finansijskim uslugama. Generalno posmatrano, prednosti elektronskog u odnosu na tradicionalno poslovanje su vezane za povećanje kvaliteta, agilnosti, za dodatne usluge sa jedne strane, i za sniženje prodajnih cena, smanjenje vremena izlaska na tržište, odnosno realizovanje transakcija sa druge strane. Takođe se intenzivira zajednički nastup na

elektronskom tržištu vezan za razvoj elektronskih partnerstva.

Plaćanje, kao neizostavni deo elektronskog poslovanja, se može definisati kao proces prenosa novčanih vrednosti. Pravna definicija platnog sistema mogla bi se formulisati na sledeći način: platni sistem je organizacija koja podržava prenos vrednosti u smislu ispunjenja novčanih obaveza [7]. Platni sistem, u najširem smislu, predstavlja skup sistema za transfer novčanih sredstava koji olakšavaju cirkulaciju novca. Da bi platni sistem na zadovoljavajući način obavljao svoju ulogu, potrebno je da se:

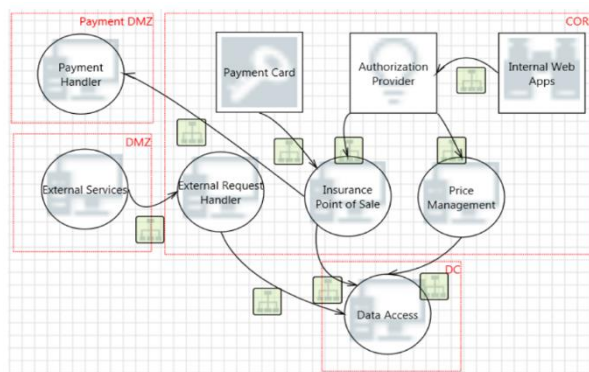
- Novčana sredstva što kraće zadržavaju u kanalima platnog prometa;
- Omogući bezbedno izvršavanje transakcija;
- Transakcije izvršavaju po ekonomski prihvatljivim cenama [8].

3. APLIKACIJA ZA INTERNET PRODAJU POLISA OSIGURANJA

Poslovanje osiguravajuće kuće se zasniva na prodaji polisa osiguranja. Prilikom definisanja osiguranja potrebno je identifikovati i izmeriti relevantne rizike radi definisanja iznosa kapitala koji osiguravajuća kompanija mora posedovati kako bi, u svakom trenutku, bila sposobna da izmiri dospele obaveze prema osiguranicima. U okviru ovoga rada razvijena je aplikacija koja omogućuje prodaju polisa osiguranja posredstvom Interneta (AIPO). Realizovani sistem se sastoji od tri bezbednosne zone:

- *DMZ zona*, koja nudi servise za klijente koji dolaze sa Interneta i predstavlja sponu između spoljašnje mreže (Interneta) i unutrašnje (intraneta);
- *Payment DMZ zona*, gde se nalazi koordinator plaćanja koji posreduje u komunikaciji između eksternog servisa koncentratara plaćanja i AIPO servisa vezanih za prodaju osiguranja;
- *CORP zona*, koja predstavlja korporativnu mrežu gde se nalaze radne stanice internih klijenata i servisi za podršku njihovog rada;

S obzirom na veličinu implementiranog sistema, fokus rada biće usmeren na prikazivanje rada servisa CORP zone i njihove komunikacije sa bazama podataka i sistemom znanja koji su prikazani na slici 2.



Slika 2. Dijagram toka podataka CORP zone

Unutar CORP zone nalaze se tri servisa čiji je zadatak da odgovaraju na zahteve koji dolaze iz DMZ zone, ali i na zahteve koji dolaze od strane internih korisnika. Ovom zonom obuhvaćene su sledeće aplikacije:

- *External Request Handler* - prihvata zahteve iz DMZ zone, vrši njihovu autentifikaciju i validaciju, te zahtev dalje prosleđuje data centru.
- *Price Management* - skup administrativnih servisa za ažuriranje podataka vezanih za ponudu osiguravajuće kuće, kao i za formiranje cene osiguranja.
- *Insurance Point of Sale* - aplikacija služi kao simulator plaćanja putem platne kartice, odnosno upotrebom Point-of-Sale (POS) terminal.

Internal Web App predstavlja aplikaciju za podršku rada zaposlenih. Zadatak ove aplikacije jeste da komunicira sa odgovarajućim servisima od kojih pribavlja podatke i kojima šalje zahteve.

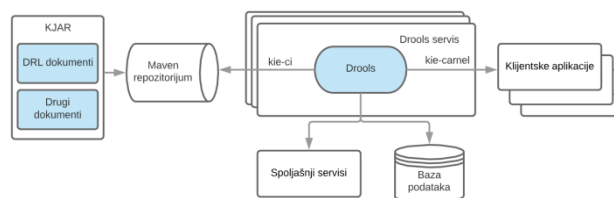
Aplikacija poznaje tri vrste korisnika. *Anonimnog korisnika* kojem je putem javnog portala AIPO aplikacije omogućena kupovina osiguranja, *prodavca* koji putem internih aplikacija ima mogućnost ažuriranja cenovnika, kao i direktne prodaje polisa osiguranja klijentima i *finansijskog analitičara*, koji predstavlja inženjera znanja i omogućeno mu je kreiranje pravila koja utiču na izračunavanje cena kupljenih polisa.

4. IMPLEMENTACIJA SISTEM ZA KALKULACIJU CENA POLISE

Svi klijentski segmenti AIPO aplikacije implementirani su korišćenjem *Angular* [9] radnog okvira u verziji 5, dok je serverski deo pisan na programskom jeziku Java [10] i to korišćenjem više *opensource* softverskih alata napisanih na Javi, kao što je *Spring* [11] radni okvir. Java je izabrana kao osnovni programski jezik, jer prevazilazi problem interoperabilnosti, odnosno Java programi mogu da se izvršavaju na bilo kom poznatijem operativnom sistemu. Aplikacija je razvijana korišćenjem *Apache Maven* [12] alata za upravljanje projektima. Za skladištenje podataka iskorišćena je *MySQL* [13] relaciona baza podataka, a pristup istoj realizovan je posredstvom *Hibernate* [14] alata za objektno-relaciono mapiranje. *Keycloak* [15], zajedno sa *Spring Security* [16] paketom je iskorišćen za implementaciju kontrole pristupa, autentifikacije i kompletne bezbednosti aplikacije.

Prvi korak u razvoju sistema za kalkulaciju cena polisa bio je da se pronađe odgovarajući razvojni alat za bazu znanja (BRE ili BRMS). Nakon analiziranja nekih od najpoznatijih alata za predstavljanje baze znanje, odlučeno je da u tu svrhu bude iskorišćen Drools. Drools [17] je sistem za upravljanje poslovnim pravilima (Business Rule Management System – BRMS) napisan na programskom jeziku Java. Osnovna premisa čitavog koncepta Drools-a leži u rečenici – *znanje je sve* (Knowledge is everything, KIE).

Drools poseduje specijalnu sintaksu za pisanje pravila koja je veoma slična Javi – Drools Rule Language. Drools kao baza znanja integrisana je u aplikaciju na način koji je opisan slikom 3.



Slika 3. Drools sistem kao servis

Nakon pronalazjenja odgovarajućeg alata i cele integracije sistema bilo je neophodno kreirati bazu znanja. U tu svrhu kreirani su odgovarajući *.drl* fajlovi čija je namena definisanje pravila. Deo definisane baze znanja prikazan je na listingu 1.

```
rule "All 3 packages include, give 2% discount"
agenda-group "travelPrice"
when
    $p: PolicyLiteDTO(vehicleInsurance == true,
                      propertyInsurance == true)
then
    $p.setFinalPrice($p.getFinalPrice() -
                    $p.getFinalPrice() * 0.02);
end

rule "Final price with coefficients"
agenda-group "sumCoefficients"

salience -1
when
    $p: PolicyLiteDTO($risks : riskItems)
    $sumCoeff : Number()
    from accumulate( RiskItemWithCoeffDTO($coeff
    : coefficient) from $risks, sum( $coeff ) )
then
    $p.setFinalPrice($p.getBaseFinalPrice() +
                    $p.getBaseFinalPrice() *
                    $sumCoeff.intValue() / 100);
end
```

Listing 1. Deo baze znanja

Drools sintaksa uslovljava da svako pravilo mora da ima jedinstveni identifikator koji se piše posle rezervisane reči *rule*. Pravila koja su predstavljena imaju različit prioritet (*salience*). Prioritet drugog pravila je -1, što znači da će ono biti izvršeno nakon svih pravila sa većim prioritetom. Takođe, podešeno je i da svako od ova dva pravila pripadaju različitim grupama (*agenda-group*). Početak premise pravila se uvek označava rezervisanom reči *when*, dok se posledica pravila piše u vidu običnih Java komandi i to posle rezervisane reči *then*. Kraj svakog pravila se označava rezervisanom reči *end* [17].

Sledeći korak implementacije uključivao je pronalazjenje odgovarajućeg web editora, putem kojeg će finansijski analitičar vršiti manipulaciju nad bazom znanja. U tu svrhu iskorišćen je TinyMCE editor [18].

Na samom kraju implementirana je forma za kupovinu osiguranja koja se sastoji iz četiri koraka. U prvom koraku kupac/prodavac popunjava podatke o putnom osiguranju kao što su dužina trajanja polise, region u koji se putuje, osigurana suma, kao i svrha putovanja. Drugi korak prikazuje kupcu prethodno popunjene podatke, kao i trenutnu cenu osiguranja, sa napomenom da u cenu još uvek nisu uključeni rizici i popusti koji se obračunavaju korišćenjem prethodno definisanih pravila. U trećem koraku se popunjavaju podaci o osiguranim licima. Četvrti i peti korak kupovine osiguranja je opcion i zavisi od toga da li se korisnik odlučio za neki od dva dodatka osnovnom osiguranju, a to su: osiguranje vozila i imovine. Poslednji korak podrazumeva obračunavanje AIPO sistema, sistemu baze znanja, koja na osnovu unesenih

podataka, a primenom definisanih pravila izračunava krajnju cenu. Deo forme prikazan je na slici 4.

Slika 4. Deo forme za kupovinu osiguranja

5. ZAKLJUČAK

Elektronsko poslovanje je poslednjih godina u nagloj ekspanziji i danas je prisutno u gotovo svim granama industrije i većini tržišta razvijenih zemalja. Stalni razvoj i sve veća prihvaćenost Interneta ujedno povećava i obim elektronskog poslovanja, dok stalni razvoj tehnologije i telekomunikacionih standarda obezbeđuju infrastrukturu za ovaj, relativno nov, model poslovanja. Danas za uspeh ovakvog vida poslovanja kompanije moraju da ulože dosta truda i pruže dodatne servise koji potrošačima daju bolje uslove kupovine, mogućnost prilagođavanja servisa i sl.

Modeli razvijani za potrebe ovog rada imaju za cilj da naprave pomak u korišćenju praktičnih alata, kao što su sistemi bazirani na pravilima, za izračunavanje cena polisa osiguranja. Pravila, susrećemo svakodnevno u našim životima, nauci, tehnologiji i kao takva predstavljaju odličan način predstavljanja znanja.

Predstavljeno rešenje bi se moglo poboljšati pre svega definisanjem složenije strukture sistema koja bi pokrila više različitih faktora koji mogu uticati na cene.

Trenutna baza znanja počiva na iskustvenim, heurističkim pravilima koja bi trebala biti zamenjena pravilima koja su nastala u dogovoru sa domenskim ekspertima. Takođe, modul za kreiranje novih pravila ne poseduje nikakav način provere ispravnosti unetih iskaza u odnosu na Drools jezik.

6. LITERATURA

- [1] Stankić R., "Elektronsko poslovanje", Centar za izdavačku delatnost ekonomskog fakulteta u Beogradu, 2008.
- [2] Čavoški S., "Simulacioni modeli zasnovani na agentima kao podrška odlučivanju u elektronskom poslovanju", Fakultet organizacionih nauka, 2016.
- [3] http://www.pcai.com/web/glossary/pcai_d_f_glossary.html (pristupljeno u junu 2018.)
- [4] Durkin J., "Expert Systems - Design and Development", Macmillan publishing company, 1994.
- [5] Jackson P., "Introduction to Expert Systems", Addison Wesley, 1999.
- [6] Tomić B., "Ekspertni sistemi i sistemi za izveštavanje", Univerzitet u Beogradu, Fakultet organizacionih nauka, 2012.
- [7] Babić S., "Model interoperabilnog elektronskog poslovanja platnih sistema zasnovanih na ontologijama", Fakultet organizacionih nauka, 2013.
- [8] <https://www.nbs.rs/internet/latinica/35/index.html> (pristupljeno u julu 2018.)
- [9] <https://angular.io/>. (pristupljeno u julu 2018.)
- [10] <https://www.java.com/en/> (pristupljeno u julu 2018.)
- [11] <https://spring.io> (pristupljeno u julu 2018.)
- [12] <https://maven.apache.org/> (pristupljeno u julu 2018.)
- [13] <https://www.mysql.com/> (pristupljeno u julu 2018.)
- [14] <http://hibernate.org/> (pristupljeno u julu 2018.)
- [15] <https://www.keycloak.org/>. (pristupljeno u julu 2018.)
- [16] <https://projects.spring.io/spring-security/> (pristupljeno u julu 2018.)
- [17] Salatino M., De Maio M., Aliverti E., "Mastering JBoss Drools 6", Packt Publishing, 2016.
- [18] <https://www.tiny.cloud/> (pristupljeno u julu 2018.)

Kratka biografija:



Mina Medić rođena je u Somboru 1994. god Osnovnu školu „Žarko Zrenjanin“ u Apatinu završila je 2009. godine. Gimnaziju „Nikola Tesla“ u Apatinu, završila je 2013. godine kao nosilac Vukove diplome. Fakultet tehničkih nauka upisala je 2013. godine - smer softversko inženjerstvo i informacione tehnologije. Osnovne studije završila je u julu 2017. godine sa prosečnom ocenom od 9,58 U oktobru 2017. godine upisala je master studije na Fakultetu tehničkih nauka - smer elektronsko poslovanje. Sve ispite predviđen planom i programom položila je u junu 2018. godine.

kontakt: alekmina94@gmail.com

KEYCLOAK SISTEM ZA UPRAVLJANJE IDENTITETOM I KONTROLOM PRISTUPA

KEYCLOAK IDENTITY AND ACCESS MANAGEMENT SYSTEM

Katarina Preradov, *Fakultet tehničkih nauka, Novi Sad*

Oblast – SOFTVERSKO INŽENJERSTVO I INFORMACIONE TEHNOLOGIJE

Kratak sadržaj – U radu je opisan Keycloak sistem za upravljanje identitetom i kontrolom pristupa. Predstavljen je sistem za internet prodaju osiguranja u okviru kojeg je projektovano i implementirano upravljanje identitetom i kontrolom pristupa uz oslonac na Keycloak.

Ključne reči: Keycloak, autentifikacija, autorizacija, kontrola pristupa

Abstract – This paper describes Keycloak system for identity and access management. Also, an overview of the system for online insurance sales is given. Within this system, identity and access management is implemented using Keycloak.

Keywords: Keycloak, authentication, authorization, access control

1. UVOD

Razvoj interneta doprineo je popularnosti veb-a i sve manjoj upotrebi klasičnih, desktop aplikacija. Korisnici veruju da će informacije koje im povere ostati bezbedne. Upravo te informacije predstavljaju osnovni resurs i moć, i njihovo posedovanje u pravom trenutku je od velikog značaja. Iz tih razloga velika pažnja se ulaže u očuvanje bezbednosti kritičnih resursa.

Kada se govori o poverljivosti, bezbednosti i kontrolisanom pristupu podacima, misli se na to da sistem i ljudi koji su zaduženi za njegovu administraciju u svakom trenutku znaju koji su to korisnici zaduženi za koji deo sistema i koja je njihova uloga u istom. To znači da svaki korisnik ima jasno definisan skup dozvoljenih operacija koje sme da izvršava nad tačno određenim skupom resursa.

Svako odstupanje od navedenog smatra se narušavanjem bezbednosti. Da bi se ovo sprečilo, neophodno je na ispravan način implementirati mehanizme za identifikaciju, autentifikaciju i autorizaciju korisnika. Identifikacija predstavlja proces utvrđivanja da li je korisnik poznat sistemu, autentifikacija je utvrđivanje identiteta korisnika, dok autorizacija predstavlja utvrđivanje prava koja korisnik ima nad resursima u sistemu. Autorizacija zahteva uspešnu autentifikaciju, dok je identifikacija sastavni deo postupka autentifikacije.

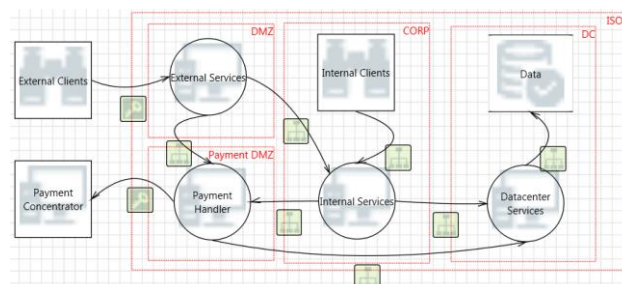
2. APLIKACIJA ZA PRODAJU OSIGURANJA

Aplikacija za internet prodaju osiguranja (AIPO) je deo informacionog sistema za podršku rada osiguravajuće kuće (ISOK) i bavi se prodajom putnog osiguranja. Poslovanje osiguravajuće kuće se zasniva na prodavanju polisa osiguranja. Polisa predstavlja ugovor između osiguranika (lica koje kupuje polisu) i osiguravača (lica koje prodaje polisu). Polisa je sredstvo kojim osiguravač garantuje da će snositi posledice ostvarenja rizika u vidu isplate dogovorene novčane naknade osiguraniku. Prodajom polise, osiguravač dobija novac, a osiguranik dobija pomenutu garanciju. Razvijen je i poseban podsistem koji predstavlja koncentrator plaćanja. Koncentrator posreduje između servisa za plaćanja (PayPal servis ili servis banke) i osiguravajuće kuće koja želi da omoguće različite vidove elektronskog plaćanja prilikom prodaje polisa.

Sa aspekta bezbednosti, AIPO je podeljena na četiri zone, pri čemu je svaka zona predstavljena rizicima kojima se poverljivi podaci u sistemu izlažu. Te zone su:

1. **DMZ - demilitarizovana zona** nudi servise za spoljne klijente. Predstavlja vezu između spoljne (internet) i unutrašnje mreže.
2. **Payment DMZ – zona plaćanja** je koordinator koji posreduje u komunikaciju između eksternog servisa koncentratora plaćanja i unutrašnjih servisa u vezi sa prodajom polisa.
3. **CORP zona** predstavlja korporativnu mrežu namenjenu internim klijentima kao i servise za podršku njihovom radu.
4. **DC zona** predstavlja *data centar*, sadrži sve baze podataka i servise za njihovu bezbednu upotrebu.

Tokovi podataka između ovih zona su složeni i prikazani su na slici 1.



Slika 1. Dijagram toka podataka AIPO sistema

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Goran Sladić, vanr. prof.

2.1. Korisnici sistema

Kako bi se implementirala autentifikacija i kontrola pristupa, neophodno je identifikovati tipove korisnika koji će koristiti sistem. U AIPO neophodno je koordinisati komunikaciju između sistema i internih korisnika (operišu u CORP zoni), anonimnih korisnika (operišu u DMZ zoni) i eksternih servisa. U skladu sa navedenim, identifikovani su sledeći tipovi korisnika definisani ulogama:

- **Anonimni korisnik** je bilo koji korisnik koji pristupa određenim servisima DMZ i Payment zone, a ne prolazi kroz proces autentifikacije.
- **Zaposleni** je korisnik koji obavezno mora proći kroz proces autentifikacije da bi mogao koristiti aplikaciju. Svaki zaposleni, nakon uspešne autentifikacije, ima pristup servisima iz CORP zone kao i svom repozitorijumu elektronske pošte.
- **Prodavac** je zaposleni koji komunicira sa klijentima koji lično dolaze u kancelariju osiguravajuće kuće da bi izvršili kupovinu polisa osiguranja.
- **Finansijski analitičar** je zaposleni koji manipuliše ponudama osiguranja i cenovnikom, kao i pravilima za izračunavanje finalne cene. Osnovica se može uvećati ili smanjiti za određeni procenat u zavisnosti od ovih pravila.

2.2. Poslovni procesi

Osiguravajuća kuća se sastoji od nekoliko poslovnih procesa koji su direktno podržani u AIPO. Ovo uključuje:

1. Kupovinu polise osiguranja
2. Izdavanje novog cenovnika
3. Ažuriranje pravila za kalkulaciju cene osiguranja

3. KEYCLOAK

Iako je bezbednost ključni aspekt bilo koje aplikacije, njena implementacija može biti komplikovana. Štaviše, često je zanemarena, slabo implementirana i nametnuta u kodu. U poslednje vreme pojavljuju se bezbednosni serveri kojima se realizuju svi aspekti u vezi sa autentifikacijom i autorizacijom. *Keycloak* predstavlja jednu implementaciju navedenih servera [1]. Reč je o sistemu otvorenog koda (eng. *open source*) razvijenom od *RedHat* [2] zajednice, namenjenom, pre svega, za veb aplikacije i RESTful servise [3]. Fleksibilan je i nezavisan od tehnologije, lako se može podesiti i prilagoditi sopstvenoj infrastrukturi kompanije. Omogućava implementaciju bezbednosti čitave aplikacije i njenih servisa sa relativno malo programskog koda.

Cilj *Keycloak*-a jeste da omogući da bezbednosni aspekti aplikacije budu jednostavni za implementaciju, tako da programerima olakša i ubrza razvoj aplikacija u okviru organizacije. Bezbednosne funkcije koje programeri obično moraju da sami pišu su implementirane u okviru *Keycloak*-a i lako se prilagođavaju individualnim zahtevima organizacije. *Keycloak* nudi korisnički interfejs za prijavljivanje, registraciju, administraciju i upravljanje korisničkim nalogom. Takođe, može se integrisati i sa drugim provajderima (eng. *third party providers*) kao što su Facebook ili Google+.

Keycloak je poseban server koji je potrebno prilagoditi za specifične potrebe sistema. Aplikacije koje koriste usluge

Keycloak-a moraju da imaju referencu ka ovom serveru. Koristi standarde otvorenog protokola kao što su *OpenID Connect* [4] ili *SAML 2.0* [5] kako bi autentifikovao aplikacije koje opslužuje. Aplikacije preusmeravaju korisnički pretraživač iz aplikacije na *Keycloak* server za proveru identiteta gde korisnici unose svoje kredencijale. Ovo je važno jer su korisnici potpuno izolovani od aplikacija i aplikacija nikada ne vidi korisničke kredencijale. Umesto toga, aplikacije imaju token koji je kriptografski potpisan. Ovi tokeni mogu imati informacije o identitetu korisnika kao što su korisničko ime, adresa, e-pošta i ostali podaci. Takođe mogu da sadrže podatke o dozvolama tako da aplikacije mogu donositi odluke o tome da li korisnik sme da pristupa traženom resursu ili ne.

3.1. Ključni koncepti i termini

Keycloak uvodi nove termine koje je potrebno razumeti da bi se ovaj sistem mogao koristiti za osiguravanje veb aplikacija i REST servisa. Neki od važnijih termina i njihovo značenje u kontekstu *Keycloak*-a su:

- **Users** (korisnici) – entiteti koji se prijavljuju na sistem. Mogu da imaju atribut kao što su e-mail, korisničko ime, lozinku, adresu, broj telefona i datum rođenja. Mogu da se dodaju u grupe korisnika i imaju uloge koje im se dodeljuju.
- **Authentication** (autentifikacija) – proces identifikacije i validacije korisnika.
- **Authorization** (autorizacija) – proces određivanja prava korisnika.
- **Credentials** (kredencijali) – podaci koje *Keycloak* koristi za autentifikaciju korisnika. To može biti lozinka, jednokratna lozinka, digitalni sertifikat ili otisak prsta.
- **Roles** (uloge) – identifikuju tip korisnika. Predefinisane uloge su *admin*, *user*, *manager* i *employee*. To su uloge koje najčešće svaka kompanija ima. Omogućeno je i dodavanje novih uloga.
- **User role mappings** (mapiranje korisničkih uloga) - definiše mapiranje između uloge i korisnika. Korisnik može biti povezan sa nula ili više uloga. Informacije o mapiranju ovih uloga mogu biti enkapsulirane u tokene.
- **Groups** (grupe korisnika) - za grupu se mogu definisati atributi i uloge. Korisnici koji pripadaju određenoj grupi nasleđuju i atribut i uloge koje ta grupa definiše.
- **Realms** (oblast) - upravlja skupom korisnika, kredencijala, uloga i grupa. Korisnik pripada i prijavljuje se u određenu oblast. Oblasti su izolovane jedne od drugih i mogu upravljati i autentifikovati samo one korisnike koji se nalaze u njoj.
- **Clients** (klijenti) - aplikacije koje mogu zatražiti od *Keycloak*-a uslugu autentifikacije korisnika.
- **Client adapters** (klijentski adapteri) - omogućavaju integraciju *Keycloak*-a sa različitim platformama.
- **Client roles** (klijentske uloge) - koje uloge imaju klijentske aplikacije.
- **Identity token** - token koji daje informacije o identitetu korisnika.
- **Access token** - token koji se može poslati u zaglavlju HTTP zahteva i pomoću kojeg se odobrava pristup servisu koji se poziva.

- **Session** (sesija) - kada se korisnik prijavi na sistem, kreira se sesija. Sesija sadrži informacije kao što su kada se korisnik prijavio ili koje aplikacije su učestvovala u okviru *Single Sign-On* mehanizma te sesije.
- **Themes** (teme) - izgled koji obezbeđuje *Keycloak* je stilizovan. Svaka tema ima definisan HTML i CSS koji se može prilagoditi različitim potrebama aplikacije

3.2. Autentifikacija

Ključni pojam je oblast (eng. *realm*) gde se definišu klijenti koji u *Keycloak* terminologiji predstavljaju aplikacije koje je potrebno osigurati. Takođe se unutar oblasti mogu definisati korisnici koji postoje u sistemu kao i njihove uloge. *Keycloak* vodi računa o autentifikaciji (koja se bazira na principu korisničkog imena i lozinke) i autorizaciji (koja se zasniva na principu korisničkih uloga). Postoji veliki broj načina za autentifikaciju korisnika i oni se mogu podeliti u tri grupe:

- na osnovu nečega što zna (npr. korisničko ime i lozinka)
- na osnovu nečega što poseduje (npr. razne vrste kartica)
- na osnovu nečega što je njegova karakteristika (npr. otisak prsta ili zenica oka)

Iako biometrijski način utvrđivanja identiteta korisnika postaje sve zastupljeniji, kombinacija korisničkog imena i lozinke je još uvek najčešće korišten pristup. Iz tog razloga, *Keycloak* uvodi pojam politike. Politika predstavlja grupu pravila koja se primenjuju u cilju formiranja što sigurnije lozinke. Predefinisane politike u *Keycloak*-u su:

- **HashAlgorithm** – algoritam za heširanje prilikom skladištenja
- **Hashing Iterations** – broj iteracija heširanja
- **Minimum Length** – najmanja dozvoljena dužina
- **Maximum Length** – najveća dozvoljena dužina
- **Digits** – broj decimalnih cifara
- **Lowecase Characters** – broj malih slova
- **Upper Characters** – broj velikih slova
- **Special Characters** – broj specijalnih karaktera
- **Not Username** – lozinka nije ista kao korisničko ime
- **Regular Expression** – mora zadovoljavati dati regularni izraz (eng. *regex*)
- **Expire Password** – broj dana koliko je lozinka ispravna
- **Not Recently Used** – nova lozinka se mora razlikovati od onih koje su korištene nedavno
- **Password Blacklist** – lozinka ne sme da se nalazi u datoj listi.

3.3. Single Sign-On

Single Sign-On (SSO) je način identifikacije korisnika na nivou sesije koja dozvoljava korisniku da koristi jedan skup kredencijala za prijavljivanje (npr. korisničko ime i lozinku) da bi pristupio različitim aplikacijama. SSO autentifikuje krajnjeg korisnika za sve aplikacije kojima je korisnik dobio prava i eliminiše dalje zahteve za autentifikaciju kada korisnik koristi različite aplikacije tokom iste sesije.

Kako AIPO predstavlja distribuirani sistem sa centralizovanim autorizacionim servisom i nekoliko korisničkih aplikacija, *Keycloak* je upotrebljen u svrhu SSO-a. Prilikom uspešne prijave na bilo koju od klijentskih aplikacija, korisniku se dodeljuje odgovarajući pristupni token. Korisnik pomoću ovog tokena može da pristupa i svakoj drugoj klijentskoj aplikaciji, iako se nije eksplicitno prijavio na istu. U ovu svrhu se koristi *OpenID Connect* protokol, koji se zasniva na *Oauth 2.0* radnom okviru [6].

4. IMPLEMENTACIJA BEZBEDNOSNOG SEGMENTA U APLIKACIJA ZA PRODAJU OSIGURANJA

Aplikacija za internet prodaju osiguranja je razvijena u programskom jeziku *Java*. Veb aplikacija je rađena u *Spring Boot* okruženju, a podaci su skladišteni u *MySQL* relacionoj bazi podataka. Kao alat za mapiranje relacionog modela na objektni korišćen je *Hibernate* radni okvir. Klijentski deo aplikacije je razvijen koristeći *Angular 4* radni okvir. Za komunikaciju između učesnika sistema korišćena je REST arhitektura zasnovana na razmeni JSON objekata. Sigurnost komunikacije realizovana je primenom *HTTPS* protokola.

4.1. Integracija Keycloak-a

Činjenica da je aplikativno rešenje pisano na *Java* programskom jeziku u *Spring Boot* radnom okviru bitno olakšava integraciju sa *Keycloak*-om. Razlog za to jeste što je sam *Keycloak* server pisan na istom jeziku. Integracija *Keycloak*-a sa *Spring*-om omogućena je dodavanjem odgovarajuće zavisnosti u *Spring Boot* projekat.

Potrebno je izvršiti određena podešavanja kako bi data komponenta na ispravan način komunicirala sa *Keycloak* serverom. Za navedena podešavanja se koristi *application.properties* fajl. U ovom fajlu vrši se podešavanje ID-a klijenta, URL pomoću koga se dobavlja autorizacioni token, tajni klijentski ključ koji je generisan od strane *Keycloak* servera, kao i naziv oblasti kojoj klijent pripada. Na slici 2 je dat primer kako se *Data Center* (DC) aplikacija povezuje sa *Keycloak* serverom.

```
# Keycloak Json properties
keycloak.auth-server-url=http://localhost:8080/auth
keycloak.realm=InternalApp
keycloak.public-client=false
keycloak.resource=datacentar
keycloak.bearer-only=true
keycloak.ssl-required = external
keycloak.credentials.secret = 6c7f38d5-93dc-427f-b67b-264a29cbdc9a
keycloak.principal-attribute=preferred_username
```

Slika 2. Integracija DC aplikacije sa *Keycloak* serverom

Nakon što se aplikacije integrišu, moguće je prepustiti sprovođenje kontrole pristupa resursima *Keycloak*-u. Ova podešavanja je takođe potrebno navesti u *application.properties* fajlu. Sva bezbednosna ograničenja su smeštena u promenljivu koja se zove *securityConstraints* i koja ograničava pristup određenim URL-ovima za korisnike koji imaju određenu ulogu ili uloge. Za URL se može koristiti tzv. *wildcard*, koji omogućava označavanje svih URL-ova koji zadovoljavaju neko pravilo. Upotreba *wildcard*-ova olakšava pisanje ovih pravila, jer se ne moraju duplirati određena ograničenja za svaki URL. Na slici 3 je prikazan isečak koda iz *Data Center* aplikacije

koji ograničava pristup svim URL-ovima aplikacije na specificirane uloge.

```
# Keycloak security
keycloak.securityConstraints[0].authRoles[0] = seller
keycloak.securityConstraints[0].authRoles[1] = financialAnalyst
keycloak.securityConstraints[0].authRoles[2] = requestHandler
keycloak.securityConstraints[0].authRoles[3] = paymentHandler
keycloak.securityConstraints[0].securityCollections[0].name = protected
keycloak.securityConstraints[0].securityCollections[0].patterns[0] = /*
```

Slika 3. Određivanje prava pristupa

Deo konfiguracije koji je prikazan na slici 3 ima sledeće značenje: samo korisnici koji imaju ulogu *seller*, *financialAnalyst*, *requestHandler* ili *paymentHandler* imaju prava da pristupe bilo kojem URL-u, zato što je „/*“ *wildcard* koji obuhvata bilo koji URL. Drugo tumačenje bi bilo da resursima *Data Center* aplikacije mogu pristupiti samo korisnici koji su prošli kroz proces autentifikacije pri čemu im je dodeljena neka od navedenih uloga.

5. ZAKLJUČAK

Keycloak predstavlja sistem za autorizaciju i autentifikaciju koji je bezbedan, robustan, stabilan, testiran i spreman za komercijalnu upotrebu. Pored toga što olakšava implementaciju, doprinosi kvalitetu konačnog softverskog proizvoda na nekoliko drugih načina. Pre svega, *Keycloak* je *open source* rešenje napisano na *Java* programskom jeziku, a pod pokroviteljstvom *RedHat* zajednice. Bezbednost *open source* rešenja počiva na velikoj zajednici ljudi koji imaju pristup kodu, pa samim tim mogu lakše uočiti propuste. Sa druge strane, *open source* priroda ne podstiče kompromitovanje bezbednosti jer se koriste poznati, javni algoritmi. *Java* programski jezik je, takođe, tehnologija koja je već dugo u komercijalnoj upotrebi i ima veliku zajednicu korisnika. Činjenica da je *RedHat* jedan od najvećih distributera *open source* projekata takođe govori o ozbiljnosti *Keycloak* alata. Takođe, aktivnost *Keycloak* zajednice se može javno pratiti i ukazuje na sve veću upotrebu i popularnost ovog alata. Kao i svaki drugi softver, *Keycloak* ima i neke nedostatke. U zavisnosti od tehnologije i arhitekture aplikacije, moguće je naići na niz poteškoća prilikom integracije *Keycloak*-a. Bez obzira na to, vreme potrebno za rešavanje ovih problema je manje od vremena koje je potrebno za implementaciju delova funkcionalnosti koje *Keycloak* pruža.

U ovom radu dat je predlog upotrebe *Keycloak* sistema u aplikaciji za *online* kupovinu polisa osiguranja. Predloženo rešenje je implementirano upotrebom servisa raspoređenih po različitim zonama koji međusobno komuniciraju putem REST API-ja. Međutim, nisu ispoštovana neki od osnovnih principa mikroservisne arhitekture. U tom smislu, budući pravci razvoja bi mogli uključivati reorganizaciju arhitekture sistema, kako bi servisi bili što više nezavisni i modularni.

Komunikacija između servisa bi se mogla unaprediti upotrebom RPC-a [7]. Drugo unapređenje se može bazirati na implementaciji dodatnih načina plaćanja, na primer izdavanja faktura, plaćanje preko računa firme i slično. Jedan od zahteva koji nije zadovoljen je upotreba *Java Card* tehnologije [8] za plaćanje putem POS (*Point Of Sale*) terminala.

Uvođenje mikroservisne arhitekture kao i novih funkcionalnosti nameće nove izazove za bezbednost sistema, pa bi u tom kontekstu bilo potrebno verifikovati u kojoj meri *Keycloak* podržava mikroservisnu arhitekturu kako i da li su potrebin neki dodatni bezbednosni mehanizmi koji bi podržali bezbednost prilikom korišćenja *Java Card* tehnologije.

6. LITERATURA

- [1] Keycloak: <http://www.keycloak.org/> (pristupljeno u julu 2018.)
- [2] RadHat: <https://www.redhat.com/en> (pristupljeno u julu 2018.)
- [3] REST: <https://docs.oracle.com/javaee/6/tutorial/doc/gijqy.html> (pristupljeno u julu 2018.)
- [4] OpenID Connect: <http://openid.net/connect/> (pristupljeno u julu 2018.)
- [5] SAML: <https://auth0.com/blog/how-saml-authentication-works/> (pristupljeno u julu 2018.)
- [6] OAuth2.0: <https://tools.ietf.org/html/rfc6749> (pristupljeno u julu 2018.)
- [7] BLOOMER, John. *Power programming with RPC*. "O'Reilly Media, Inc.", 1992
- [8] Java Card API: <http://www.oracle.com/technetwork/java/embedded/javacard/overview/index.html> (pristupljeno u julu 2018.)

Kratka biografija:

Katarina Preradov je rođena 16.09.1994. godine u Novom Sadu. Osnovnu školu „Vuk Karadžić“ u Bačkoj Palanci završila je 2009. godine. Gimnaziju „20. oktobar“ u Bačkoj Palanci završila je 2013. godine. Osnovne akademske studije na Fakultetu tehničkih nauka u Novom Sadu završila je 2017. godine na smeru Softversko inženjerstvo i informacione tehnologije. 2017. godine upisuje master akademske studije na Fakultetu tehničkih nauka u Novom Sadu, smer Softversko inženjerstvo i informacione tehnologije, modul Elektronsko poslovanje. Iste godine se zapošljava kao saradnik u nastavi na Fakultetu tehničkih nauka, Katedri za informatiku.

**PREGLED SOFTVERSKIH ALATA ZA SIMULACIJU RADA FOTONAPONSKIH SISTEMA
OVERVIEW OF PHOTOVOLTAIC SYSTEM SIMULATION SOFTWARES**

Tijana Savčić, Vladimir Katić, *Fakultet tehničkih nauka, Novi Sad*

Oblast: ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu opisano je nekoliko odabranih softverskih alata za simulaciju fotonaponskih (FN) sistema. Svaki od alata je ocjenjen prema određenim kriterijumima, te je na kraju odabran softver koji je optimalan za simulaciju rada sistema. On je primjenjen za simulaciju rada FN elektrane izgrađene na krovu FTN-a u Novom Sadu. Dobijeni rezultati su upoređeni sa stvarnom proizvodnjom ostvarenom u prve tri godine rada elektrane i utvrđeno je malo odstupanje (<4%).

Ključne reči: Obnovljivi izvori energije, Fotonaponski sistemi, softverski alati za projektovanje i simulaciju rada fotonaponskih sistema

Abstract –The paper describes several selected software tools for simulation of a photovoltaic (PV) systems. Each of the tools is evaluated according to certain criteria, and at the end, the optimal software is selected. It is used to simulate the operation of the roof-top PV plant built of the Faculty of Technical Sciences. The obtained results are compared with the actual generation in the first three years of operation and low discrepancy is stated (<4%).

Keywords: Renewable energy sources, photovoltaic systems, software tools for design and simulation of photovoltaic systems

1. UVOD

Energija Sunčevog zračenja je više nego dovoljna da zadovolji sve potrebe za energijom na svijetu. Sa njom raspolaže manje više svaka zemlja na svijetu, pa je to jedan od razloga zbog čega je interesovanje za ovim oblikom energije sve veće [1]. Pri tome polazna, ključna informacija jeste potencijal solarne energije na određenoj lokaciji, koji se može dobiti merenjem.

Ova merenja se više desetina godina obavljaju u sklopu standardnih aktivnosti hidro-meteoroloških stanica širom sveta, a takođe (na indirektan način) i upotrebom geostacionarnih satelita. Rezultati se nalaze u obimnim bazama, tako da je primena softverskih alata neminovna. Takođe, za potrebe korišćenja solarne energije u vidu fotonaponskih (FN) sistema potrebno je izvršiti određene proračune što je dovelo do razvoja i potražnje softverskih alata koji služe za projektovanje, simulaciju i analizu rada ovih sistema.

U postojećoj literaturi ima više različitih pregleda ovih softvera zasnovanih na različitim ciljevima i kriterijumima, koji mogu poslužiti za lakši izbor najpogodnijeg za određenu aplikaciju [2-5].

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Katić, red.prof.

U pregledu Nemačkog energetskog udruženja iz 2008. godine, data je podela različitih programa po ciljevima njihovog sastavljanja [2]. Definirano je šest ciljeva, a sistematizovano je čak 49 različitih softvera, koji su zatim detaljnije obrađeni. Međutim, nije urađeno njihovo poređenje i rangiranje.

U [3] su predstavljeni softveri do 2012. god., koji su bili raspoloživi za pristup preko Interneta. Ovi programi su sistematizovani u šest kategorija, a detaljnije su analizirani najinteresantniji, po izboru autora: Sunny Design, Profotonapsis, PV Syst, Homer, PV Sol, i Solar Advisor Model (SAM). Primenom devet različitih kriterijuma (Cena, Izvor meteo podataka, Tracking, Senčenje, Izlazni podaci, Moduli, Invertori, Vreme učenja i Opšti utisak), kao najpovoljniji odabran je softver Sunny Design i primenjen je na primeru proračuna solarne FN elektrane snage 0,5 MW na lokaciji u blizini Beograda.

U [4] je razmatrano samošestlako dostupnih i aktuelnih softvera u 2016. god.: PV GIS, PV Syst, Sunny Design, PV Size, SAM i Homer. Poređenje je vršeno po devet različitih kriterijuma: biblioteka modula, biblioteka invertora, baza meteoroloških podataka, efekat senčenja, izlazni podaci, složenost strukture, vreme učenja, dostupnost i cena softvera, opšti utisak. Na osnovu ocenjivanja i njihovog rangiranja odabran je softver PV Syst, koji je zatim primenjen na FN elektranu „FTN-1“ u Novom Sadu. Utvrđeno je da projektovana i stvarna proizvodnja ove FN elektrane odstupaju samo za +3,7%. Međutim, uočeno je da ovaj softver ne daje elemente proračuna isplativosti FN elektrane.

U [5] su razmatrani softveri za modelovanje i simulaciju kućnih (krovnih) FN elektrana raspoloživih u 2018. god. Posmatrano je četiri različita softvera: Homer, PV Syst, Solarius PV i BlueSol. Pomoću njih je urađeno projektovanje i analiza jedne krovne FN elektrane snage 4,96 kWp. Dobijeni rezultati su analizirani i ocenjeni na bazi sedam kriterijuma: okruženje, baza podataka, jednostavnost, licenca, dodatne funkcije, dobijeni rezultati i tačnost. Najbolje ocenjen je bio softver Solarius PV. Za toga su još svi rezultati upoređeni sa proizvodnjom FN elektrane „FTN-1“, Novi Sad snage 9,6 kWp i utvrđeno je da najmanje odstupanje od samo -2,5% ima odabrani softver Solarius PV.

S obzirom da su u pregledanoj literaturi različiti autori došli do različitih zaključaka, odnosno da su predloženi različiti najpovoljniji softveri, cilj ovog rada jeste da se napravi pregled raspoloživih besplatnih softvera i da se razmotre njihove karakteristike. Biće razmatrani samo softveri za projektovanje i simulaciju rada FN sistema.

U ovom radu će biti prikazano šest različitih softverskih alata. To su PvGis, RetScreen, PvPlanner, Homer, PvSyst

i SAM, a odabrani softver će biti upoređen i evaluiran sa podacima od krovne FN elektrane „FTN-1“, Novi Sad.

2. SOFTVERSKI ALATI ZA PROJEKTOVANJE I SIMULACIJU RADA FN SISTEMA

2.1. PVGIS

PVGis je besplatan softver koji je dostupan na internetu. Služi za procjenu proizvodnje električne energije FN sistema. Cilj ovog softvera je da pokaže kakav uticaj ima izabrana lokacija na planiranje FN sistema, a prikazuju se i podaci o Sunčevom zračenju za izabranu lokaciju za duži vremenski period. Mana ovog softvera je ta što za analizirani sistem ne postoji finansijski model kao i ta što ne postoji biblioteka sa komponentama sistema.

2.2. RetScreen

RetScreen Expert je najnovija verzija softvera RetScreen, unutar koje je integrisano više opcija za analizu rada sistema. Na osnovu lokacije se proračunava proizvodnja električne energije. Pored toga, vrši se i analiza troškova i utvrđuje se izvodljivost projekta.

Moguće je formirati različite konfiguracije sistema i potom izvršiti poređenje istih kako bise došlo do optimalnog i najisplativijeg rješenja. Mana je što je broj komponenti sistema sadržan u biblioteci ograničen i nije moguće dodavanje ili formiranje modula ili invertora od strane korisnika, te ne posjeduje opciju za analizu sjenčenja.

2.3. PvPlanner

PvPlanner je softver koji je dostupan samo na internetu i za njegovo korišćenje je neophodna internet konekcija. Precizni satelitski podaci omogućuju korisniku da napravi procjenu o Sunčevom zračenju za određenu lokaciju. Automatski se izračunava sjenčenje sistema, ali ne postoji opcija postavljanja objekata koji se nalaze u blizini i bacaju sjenku na sistem. Korisnički interfejs je prilično intuitivan. Mana je što ne postoji opcija za unosom meteoroloških podataka od strane korisnika, kao ni biblioteka sa komponentama, već se koriste generički moduli i korisnici biraju samo tehnologiju izrade modula. Invertori su takođe generički i jedino što korisnik bira je efikasnost invertora. Probna verzija je besplatna, dok puna verzija predstavlja najskuplju u odnosu na analizirane softvere.

2.4 Homer

Homer je softver koji za cilj ima projektovanje i analizu rada mikro-mreža. Ovaj softver služi za simulaciju rada sistema sa više izvora energije i više različitog opterećenja. Nije posebno dizajniran za fotonaponske sisteme, te je izvještaj o proizvodnji električne energije uz pomoć FN sistema manje detaljan, prvenstveno zbog ograničenih meteoroloških podataka i podataka o modulima. Nije naročito intuitivan, jer se dosta parametara unosi ručno od strane korisnika. Takođe, ne postoji biblioteka komponenti sistema, što predstavlja nedostatak.

2.5. PVSyst

PvSyst je softver čije su glavne odlike brza procjena proizvodnje u fazi projektovanja, detaljna studija, dimenzionisanje sistema i izvještaj u vremenskim intervalima od jednog sata. Uz pomoć ovog softvera se vrši simulacija

većine parametara koji su bitni za FN sistem i generiše se seobuhvatni izvještaj simulacije. Baza meteoroloških podataka je praktično neograničena s obzirom na to da je moguće uvesti podatke iz 15 različitih izvora. Posjeduje izuzetno veliku bazu podataka o komponentama.

2.6. System advisor model (SAM)

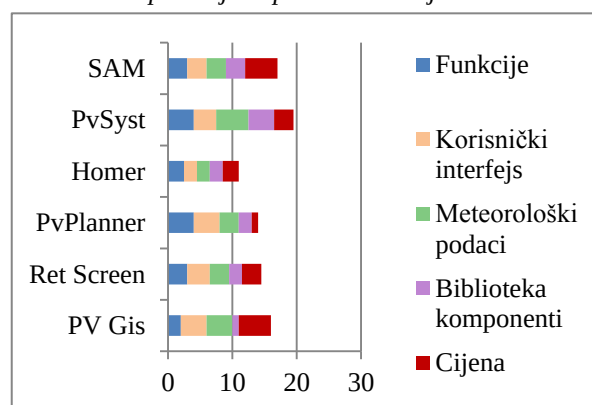
System advisor model (SAM) je besplatan softver. Cilj ovog softvera je da izvrši predviđanje rada sistema i troškove energije sistema koji je povezan na mrežu na osnovu troškova instalacije i troškova tokom rada koji su određeni parametrima unijetim od strane korisnika. Unutar softvera su sadržani različiti tipovi finansijskih modela koji se koriste u SAD, a korisnik može da odabere odgovarajući model kako bi obezbjedio ulazne podatke za pokretanje simulacije i za generisanje izvještaja o proizvodnji energije, gubicima i finansijskim troškovima. Veliki broj parametara se unosi od strane korisnika. Posjeduje veliku bazu podataka o modulima i invertorima.

3. POREĐENJE I ODABIR NAJPOVOLJNIJEG SOFTVERA

Svaki od predhodno navedenih softvera je ocjenjen prema različitim kriterijumima i izvršeno je međusobno poređenje. Poređenje između programa je urađeno kako bi se utvrdilo koji od ovih softvera je najbolji za projektovanje i simulaciju solarnih FN sistema. Ovom analizom su uzeti u obzir alati koji služe prije svega za projektovanje i simulaciju FN solarnih sistema, samim tim softveri koji imaju opcije za projektovanje i simulaciju rada drugih načina proizvodnje električne energije, nisu bolje ocjenjeni zbog te činjenice. Kriterijumi po kojima su ocjenjeni softverski alati su sledeći: funkcije, korisnički interfejs, baza podataka o vremenu, biblioteka modula i invertora i dostupnost i cijena softvera.

Rezultati međusobnog poređenja predstavljeni su u tabeli 1. Može se videti da je najbolje ocenjen softver PV Syst (kao i u [4]), koji je ukupno ocenjen sa 19 od 25 mogućih poena. U nastavku rada će ovaj softver biti iskorišćen za simulaciju rada FN elektrane izgrađene na krovu Fakulteta tehničkih nauka u Novom Sadu „FTN-1“.

Tabela 1: Ukupna ocjena po svim kriterijumima



4. PRIMJENA ODABRANOG SOFTVERA

Nakon pregleda softverskih alata, odabran je optimalan alat za simulaciju FN sistema.

Kao najbolji softver odabran je PvSyst. Uz pomoć ovog softverskog alata je odrađena simulacija rada FN elektrane izgrađene na krovu Fakulteta tehničkih nauka.

4.1. Fotonaponska elektrana

FN elektrana na krovu zgrade Fakulteta tehničkih nauka je snage 9,6 kWp, odnosno izlazne 8 kW. U sklopu ovog sistema je trofazni inverter Sunny Tripower 8000 TL, proizvođača SMA Solar Technology, nominalne izlazne snage 8 kW, na koji su priključena dva niza FN panela od polikristalnog silicijuma (40 kom.), tipa JKM- 240P proizvođača Jinko Solar maksimalne snage 240 Wp [6].

4.2. Rezultati simulacije

Rezultati koji su dobijeni nakon odrađene simulacije, a koji predstavljaju procenjenju proizvodnju električne energije dati su na slici 1. Ovi rezultati su upoređeni sa stvarnom proizvodnjom električne energije za period prve tri godine rada elektrane od 2012. do 2014. god. [6].

U tabeli 2 prikazane su vrijednosti stvarne i procenjene proizvodnje električne energije. Može se uočiti da postoje određena odstupanja, kako stvarne u odnosu na procenjenju, tako i među stvarnim vrijednostima.

To se objašnjava različitim meteorološkim prilikama koje za pojedine godine odstupaju od dugogodišnjih prosjeka. Vidi se da ukupna procenjena proizvodnja dobijena simulacijom u PvSyst-u iznosi 11.584 kWh, dok je stvarna za prve tri godine rada FN elektrane „FTN-1“ Novi Sad bila 11.128,67 kWh. Odstupanje stvarne u odnosu na procenjenju godišnju proizvodnju je oko 4%.

FTN PV SIM
Balances and main results

	GlobHor kWh/m ²	DiffHor kWh/m ²	T Amb °C	GlobInc kWh/m ²	GlobEff kWh/m ²	EArray MWh	E_Grid MWh	PR
January	34.9	19.20	0.24	54.2	50.8	0.476	0.451	0.867
February	55.1	30.20	2.39	75.1	70.5	0.654	0.628	0.871
March	100.3	49.10	7.44	122.2	115.1	1.030	0.996	0.849
April	137.1	67.00	12.49	150.0	141.2	1.227	1.189	0.826
May	178.5	82.60	18.06	178.2	167.0	1.416	1.373	0.802
June	183.1	82.10	20.57	177.4	166.4	1.390	1.344	0.789
July	200.5	71.20	22.52	200.0	188.0	1.537	1.490	0.776
August	165.7	73.70	22.29	175.4	165.2	1.365	1.322	0.785
September	114.5	54.60	16.59	132.6	124.8	1.064	1.031	0.810
October	79.6	41.50	12.57	105.6	99.3	0.874	0.844	0.833
November	46.2	25.00	6.95	70.0	65.5	0.597	0.572	0.851
December	28.2	18.00	1.71	42.2	39.5	0.368	0.344	0.848
Year	1323.7	614.20	12.04	1482.9	1393.2	11.998	11.584	0.814

Legends: GlobHor Horizontal global irradiation
DiffHor Horizontal diffuse irradiation
T Amb Ambient Temperature
GlobInc Global incident in coll. plane
GlobEff Effective Global, corr. for IAM and shadings
EArray Effective energy at the output of the array
E_Grid Energy injected into grid
PR Performance Ratio

Slika1. Rezultati simulacije

5. ZAKLJUČAK

Prilikom planiranja FN sistema koriste se softveri za projektovanje i optimizaciju rada elektrane. Programi koji za cilj imaju dimenzionisanje i simulacije omogućavaju da se provjere radna stanja sistema i da se izvrši simulacija rada sistema pri različitim uslovima.

Ovdje je opisano šest različitih softvera za simulaciju rada elektrane, a kao najbolji odabran je PvSyst. U PvSyst-u je simuliran rad FN elektrane na krovu Fakulteta tehničkih nauka.

Dobijeni rezultati su upoređeni sa stvarnom proizvodnjom elektrane.

Uočena su određena odstupanja stvarne proizvodnje u odnosu na predviđenu, kako na mjesečnom nivou tako i na godišnjem nivou. Međutim, ona su mala i iznose ispod 4%.

Tabela 2. Prikaz stvarne i procenjene proizvodnje električne energije

	PVSys	2012.	2013.	2014.
Januar	451	499,25	400,21	371,15
Februar	628	552,68	427,02	656,1
Mart	996	1300,39	794,76	1026,9
April	1189	1118,2	1227,57	1035,3
Maj	1373	1307,1	1262,39	1246,7
Jun	1344	1542,8	1182,58	1315,5
Jul	1490	1485,17	1595,89	1353,5
Avgust	1322	1522,98	1385,3	1319,6
Septembar	1031	1113,56	944,76	874
Oktobar	844	824,29	916,55	822,7
Novembar	572	658,65	552,64	542,3
Decembar	344	229,23	486,8	491,5
Ukupna godišnja proizvodnja	11584	11154,3	11176,47	11055,25

6. LITERATURA

- [1] S. Marković, S. Mijailović "Renewable energy and energy management's influence on job creation", Visoka poslovna škola "Prof dr Radomir Bojković", 2015.
- [2] Planning and Installing Photovoltaic Systems, A guide for installers, architects and engineers, second edition, The German Energy Society, Berlin, 2008.
- [3] Mladen Petronijević, Vladimir Katić, „Projektovanje fotonaponske elektrane – pregled softverskih alata“, *Zbornik radova Fakulteta tehničkih nauka*, God.27, Br.5, 2012, pp.1025-1028
- [4] Milica Repac, „Pregledsoftvera za modelovanje FNelektreana“, Diplomski rad, Fakultet tehničkih nauka, Novi Sad,2016.
- [5] Miroslav Mega, "Komparativnaanalizasoftvera za projektovanje FN krovnihsistema", Diplomski rad, Fakultet tehničkih nauka, Novi Sad,2018.
- [6] V. Katić, Z. Čorba, B. Dumnić, D. Milićević, B. Popadić, "Realizacija krovne fotonaponske elektrane na Fakultetu tehničkih nauka u Novom Sadu", *Tehnika*, Vol.70, Br.4, 2015, str. 655-662.

Kratka biografija:



Tijana Savčić rođena je u Loznici 1991. god. Diplomirala je na Fakultetu tehničkih nauka 2015. godine sa temom "Revitalizacija hidroelektrane Zvornik". Iste godine upisuje Master akademske studije na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine. Master rad je odbranila 2018. godine.



Vladimir Katić je rođen 1954. godine u Novom Sadu. Doktorirao je na Univerzitetu u Beogradu 1991. godine. Od 2002. godine je redovni profesor Univerziteta u Novom Sadu. Trenutno je i prodekan Fakulteta tehničkih nauka i šef katedre za energetska elektroniku i pretvarače. Oblasti interesovanja su energetska elektronika, obnovljivi izvori energije, električna vozila i kvalitet električne energije

**ПРОНАЛАЖЕЊЕ РЕФЕРЕНТНИХ ТАЧАКА НА РЕНДГЕНСКИМ СНИМЦИМА
КАРЛИЦЕ ПРИМЕНОМ КОНВОЛУЦИОНИХ НЕУРОНСКИХ МРЕЖА
DETECTING LANDMARKS ON PELVIS X-RAYS USING CONVOLUTIONAL NEURAL
NETWORKS**

Радован Туровић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – Главни допринос овог рада је нови метод за детекцију три кључне тачке на карлици: расис, ласис и симфисис на некалибрисаном рендгенском снимку карлице, базиран на конволутивним неуронским мрежама, робустан на транслацију и ротацију карлице. У раду је предложено једно решење за проблем недовољно великог обучавајућег скупа засновано на генерисању рендгенских снимака на основу стварних снимака компјутерске томографије. Обучена мрежа је способна да довољно прецизно одреди позиције кључних тачака у случајевима када на рендгенском снимку нема заштитника од радијације и карлица је читава обухваћена снимком.

Кључне речи: карлица, рендгенски снимак, машинско учење, конволутивне неуронске мреже, генерисање скупа података

Abstract – Main contribution of this paper is new method for landmark detection on pelvis X-rays: Rasis, Lasis and Simphysis from uncalibrated X-ray, based on convolutional neural networks and robust to slight translations and rotations. In this paper a solution for the lack of instances in X-rays training dataset is proposed by generating X-rays from Computed Tomography scans. Trained network is able to precisely enough predict landmarks' positions on X-rays without radiation protector and with full pelvis in view.

Keywords: pelvis, X-ray, machine learning, convolutional neural networks, data augmentation, data generation

1. УВОД

Реконструкција 3Д модела различитих делова људског тела игра важну улогу у унапређењу савремене медицине у области планирања индивидуалне терапије. Иако, у основи, карлица сваког човека има исту анатомију, карлице се могу међусобно разликовати по величини, пропорцијама и низу других параметара.

Због тога је битно да лекар, нарочито у случају планирања хируршког захвата има прецизне информације о делу тела пацијента над којим треба извршити захват.

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је био др Драган Дину, доцент.

Пример једне овакве хируршке интервенције, која се изводи на карлици пацијента, је замена природног кука вештачким (артропластика кука). Планирање овог оперативног захвата за конкретног пацијента подразумева одређивање (1) фактора заснованих на медицинској анализи пацијента која обухвата утврђивање типа и обима обољења, старост пацијента, избор метода оперативног захвата и (2) фактора који утичу на облик и геометрију импланата, где су најзначајније димензије и морфолошке карактеристике карлице и бутне кости пацијента.

Током времена је развијена типизација импланата према геометријским и оперативним критеријумима, па се тако импланти могу поделити према типу, димензијама (обично до 10 по типу) као и начину фиксирања у бутну кост. Користећи установљену типизацију, за конкретног пацијента се пре оперативног захвата бира један или више унапред специфицираних импланата, за које је медицинским анализама установљено да потенцијално одговарају том пацијенту.

Уколико има више импланата за одабир, он се не може одабрати пре него што се над пацијентом започне захват и стигне до фазе уградње импланата. Овакав начин планирања замене кука за последицу може да има уграђивање импланата који не одговара пацијенту у потпуности. Овај проблем би могао успешно бити решен применом индивидуалне терапије, што подразумева израду 3Д модела карлице датог пацијента и израду импланата који у потпуности одговара пацијенту.

Овакав приступ би позитивно утицао на квалитет живота пацијента након хируршког захвата, као и на смањење трошкова израде импланата, јер би било довољно направити један имплант по мери пацијента уместо више сличних и тражити онај који највише одговара пацијенту у току оперативног захвата.

Тренутно најраспрострањенији метод реконструкције 3Д модела карлице је метод заснован на ЦТ снимку (енг. *Computed Tomography*). Предност овог метода је што сам ЦТ снимак садржи податке о тродимензионалности карлице, због чега је могуће направити прецизан 3Д модел карлице уз ослонац на ЦТ снимак. Међутим, ЦТ снимак се прави док је пацијент у лежећем, односно растеретном положају, због чега кости карлице нису у истом положају у којем би биле приликом уобичајених свакодневних активности, за разлику од рендгенског снимка који се прави док је пацијент у стојећем положају. Ово може довести до реконструкције неадекватног 3Д модела

који служи као основа при индивидуалној терапији пацијента и може узроковати озбиљне грешке у лечењу пацијента. Уз наведено, ЦТ снимање излаже пацијента већим дозама зрачења у односу на рендген, што у одређеним условима лакше може проузроковати појаву малигних хелија код пацијента.

Овај рад је корак ка реконструкцији 3Д модела карлице на основу некалибрисаног рендгенског снимка. Представљен је поступак базиран на конволуционим неуронским мрежама за аутоматско одређивање позиције три референтне тачке на карлици: Расис, Ласис и Симфисис (лат. *Rasis, Lasis* и *Symphisis*) које одређују антериорно-постериорну раван. На основу реконструисаних тачака је применом већ резвијених метода могуће реконструисати 3Д модел карлице.

2. ПРЕГЛЕД СТАЊА У ОБЛАСТИ

Рани радови у области детекције кључних тачака на рендгенским снимцима заснивају се на идентификацији карактеристика ниског нивоа, као што су ивице органа и костију на рендгенском снимку [1]–[3] како је наведено у [4]. Овакви методи детекције су се у општем случају показали осетљивим на варијабилни контраст на рендгенским снимцима, као и шум и оклузије (заклањање дела ткива).

Развој машинског учења је резултовао новим методама које се користе за учење и предвиђање позиција кључних тачака на рендгенском снимку. Међу техникама машинског учења, најбоље су се показали регресиони модели тренирани над великим бројем исечака рендгенских снимака [4]–[6]. Напредак дубоког машинског учења и појава медицинских сетова података претходили су развоју метода за детектовање кључних тачака уз помоћ дубоких неуронских мрежа. Различите методе детекције локације кључних тачака су представљене у [7]–[11]. Од ових радова издвајају се два рада према начину на који је резултујућа дубока неуронска мрежа истренирана са становишта улазних података. У [7] аутори у недостатку великог броја тренинг рендгенских снимака мрежу тренирају помоћу великог броја исечака рендгенских снимака, док је у [12] приказан другачији приступ руковању малим тренажним сетом података, где се конволуциона мрежа претходно истренирана на немедицинским снимцима доучава обучавајућим скупом који садржи медицинске снимке да препознаје локације кључних тачака на снимку. У овом раду је примењен другачији приступ од оних објашњених у [7], [13] заснован на генерисању великог обучавајућег скупа на основу реалних података.

3. МЕТОДОЛОГИЈА

Да би конволутивна неуронска мрежа научила да прецизно одређује позиције кључних тачака на рендгенском снимку, потребно ју је обучити добро осимшљеним и довољно великим обучавајућим скупом. У овом поглављу је описано како је формиран тренинг скуп, који су алати коришћени за припремање скупа, тренирање неуронске мреже и евалуацију резултата, као архитектура модела и поступак тренирања неуронске мреже.

3.1 Алати

Обучавање мреже се изводило на ХЛРН-3 комплексу (ориг. HLRN-III), односно суперрачунару Креј ИксЦе30 (ориг. Cray XC30). Суперрачунар се састоји од многобројних чворова од којих сваки садржи 24 Интелових Ајвибриџ језгара (ориг. Intel IvyBridge) на 2.4 GHz и 64 GB радне меморије.

За формирање обучавајућег скупа, визуализација модела и генерисаних слика је била од кључног значаја. У ту сврху је коришћен Амира софтверски пакет за визуализацију и графичку обраду. У алату је коришћен модул који је у стању да симулира простирање рендгенских зрака кроз 3Д запремину сачувану у ЦТ снимку и направи 2Д пројекцију истих на раван која симулира детекторску плочу рендгенског уређаја. Алат има могућност скриптовања и оно је реализовано у Тикл скриптинг и Пајтон програмском језику. Осим самог генерисања снимака, скриптоване су трансформације ЦТ снимака, односно транслација и ротација, као и измена параметара у генерисању снимака попут интензитета зрачења и контраста између различитих ткива, и угао фростума простирања зрака. Моделовање нервне мреже као и њено тренирање и коришћење је писано у Пајтон програмском језику са ослоном на Гуглову Тензорфлу (енг. *Tensorflow*) библиотеку за машинско учење кроз Керас (енг. *Keras*) библиотеку.

3.2 Подаци

Обучавајући скуп коришћен у овом раду је настао пројектовањем ЦТ снимака пацијената на дводимензиону раван како би се добили рендгенски снимци. Оправдање за овај корак се може наћи у 2 битне чињенице. Пре свега ЦТ снимци настају прављењем низа рендгенских снимака што природу ЦТ снимка чини идентичном рендгенском снимку. Такође, ЦТ снимци дају верну и прецизну тродимензиону репрезентацију модела на којој се веома прецизно могу означити референтне тачке неопходне за обучавање.

Симулирањем простирања рендгенских зрака кроз модел изгенерисани су рендгенски снимци у Амири. Процес припреме симулације садржи пар основних корака:

- иницијално постављање 3Д модела карлице у простор,
- постављање детекторске плоче са једне стране модела и извора зрачења са супротне стране у односу на модел,
- варирање положаја и ротације модела,
- варирање величине детекторске плоче и раздаљине извора од плоче,
- варирање таласне дужине зрачења(контраст) и интензитета.

Осим очигледних промене попут боље/лошије видљивости ткива на рендгену, контраста, ротације и помераја, приликом промене положаја и ротације дешавају се и суптилније промене. Због природе пројекције, пошто се радијација шири купасто из тачкастог извора ка детекторској плочи и притом

настаје перспективна пројекција, долази до дисторзије простора.

Делови ткива који су ближе извору радијације него детекторској плочи делују већи на крајњем снимку. Стога, због деформације пројекције услед промена приликом ротација и транслагација, ове трансформације су уведене да би добили скуп података над којим ћемо обучити мрежу која је отпорна на овакав тип деформација.

Референтне тачке су добијене уз ослонац на тродимензионе моделе карлица. Експерт из области медицине је на моделима обележио референтне тачке. Од обележених тачака, за овај пројекат су одабране 3 тачке, расис, ласис и симфисис (лат. Rasis, Lasis и Simphysis). Оне су специфичне по томе што заједно чине раван под називом антериорно-постериорна раван (раван која дели људско тело на предњу и задњу страну).

Пошто као излаз неуронске мреже очекујемо координате ових референтних тачака на рендгенском снимку, њихова припрема се ослања на исти принцип као и генерисање рендгенских снимака.

За сваку инстанцу изгенерисаног рендгенског снимка извршена је и пројекција кључних тачака на детекторску плочу и у околици су израчунате координате релативно у односу на димензије детекторске плоче. Координате референтних тачака су представљене као проценти одступања тачке од почетка детекторске плоче, њеног горњег левог угла, у односу на дужину, тј. ширину детекторске плоче.

За овај пројекат на располагању је било 50 ЦТ снимака. По сваком ЦТ снимку, изгенерисано је 500 различитих рендгенских снимака што резултује обучавајућим скупом од 25 хиљада јединствених инстанци рендгенских снимака. Визуелним поређењем насумично одабраних изгенерисаних рендгенских снимака са стварним рендгенским снимцима, закључено је да су изгенерисани снимци веродостојни.

3.3 Поступак

Обучавање је подељено у фазе, при чему је само први модел конволутивне нервне мреже трениран од почетка, док је свако наредно проширење обучавано са претходно обученим тежинама једноставније мреже. Приликом сваког проширења, није прекинут процес обучавања једноставнијих модела, већ су фино дообучавани да се са већим бројем итерација потенцијално добије већа прецизност.

Сходно наведеном, у првој фази је обучен једноставни модел до одређене прецизности. У другој фази су од те тачке паралелно тренирани једноставан модел (први модел) и модел са проширењем (други модел) конволутивних слојева нервне мреже. Након тога, додана је мрежа са проширењем и у потпуно повезаном слоју (трећи модел), тренирана са већ обученим тежинама другог модела. Други модел је дао најбоље резултате а његов модел је следећи:

- КККАККАРККАКАРПППА

где К представља конволуциони слој, А представља посебно спецификован активациони слој, Р за редукциони слој и П за потпуно повезан слој. Сви конволуциони слојеви имају језгра димензија 3x3.

Само обучавање трајало је по 25 епоха у серијама од по 25 слика (енг. batch). Од алгоритама за ажурирање тежина, коришћен је Адам оптимизатор са фактором пропадања стопе учења (енг. learning rate decay). Ради боље генерализационе способности обучаване конволутивне неуронске мреже уведена је и стопа отказа (енг. dropout) од 25%.

Функција губитка (енг. loss function) је веома битна за обучавање јер дефинише удаљеност излаза неуронске мреже у односу на очекивани излаз. Неадекватно дефинисана функција губитка може довести до неспособности мреже да учи. С друге стране, адекватно дефинисана функција губитка може донети значајне бенефите попут бржег обучавања као и могућности да мрежа боље научи да израчуна излазе. За овај рад је оформљена специфична функција губитка. Наиме, у случају простог еуклидског растојања се дешавало да две израчунате референтне тачке буду веома близу очекиваним положајима док трећа значајно одступа. Да би се овај проблем санирао, осим поменуте метрике у функцију губитка је додано доменско знање. Оно што желимо јесте да се тачке "држе заједно", што је омогућено израчунавањем разлике између растојања парова очекиваних положаја референтних тачака и растојања парова израчунатих положаја тачака. Доменска компонента функције губитка ће приликом оцењивања рада мреже "казниће" самостално екстремно одступање јер се повећањем разлике у обиму погоршава и оцена перформанси модела. Резултат је ограничено одступање самосталних тачака, односно, тачке ће се груписати у жељени троугаони облик из чега ће проистећи тачнија предикција.

4. РЕЗУЛТАТИ

Извршена је валидација и анализа података о тренирању и перформансама развијаних мрежа. Најбоље резултате је дала друга мрежа (прво проширење иницијалне мреже).

Приликом обучавања, функција губитка се понашала очекивано. У последњој итерацији је примећено да губитак приликом валидације неочекивано има нижу вредност у односу на губитак приликом тренирања. У општем случају ова појава говори да се обучавана мрежа превише специјализова за обучавајући скуп. Међутим, анализирањем понашања обе компоненте функције губитка понаособ, дошло се до закључка да је мрежа веома добро научила троугаону структуру коју праве кључне тачке.

На основу урађене статистичке анализе над ~1000 добијено је да у случаја све 3 кључне тачке постоји одступање мање од 20 mm у преко 90% инстанци. Просеци релативних одступања предикованих тачака Ласис, Симфисис и Расис су 1,84%, 1,86% и 1,59%, респективно.

Процена перформанси неуронске мреже над 13 инстанци реалних рендгенских снимака је вршена визуелно услед недостатка времена. За свих 13 инстанци извршена је предикција положаја кључних тачака. Потом је експерт из домена медицине анализирао резултате мреже и дао оцену да ли је мрежа добро означила кључне тачке или није. Закључено је да у случају када се само пацијент налази на снимку и када се кости карлице у потпуности виде на снимку, предикција је веома прецизна. Пример успешне реконструкције дат је на слици 1.

5. ЗАКЉУЧАК

Сходно временским ограничењима пројекта током којег је овај експеримент вршен, резултати су обећавајући. Губитак најбоље мреже је мањи од 10%. На крају обучавања није достигнут плато (што се могло видети приликом надгледања функције губитка у току обучавања) што је показатељ да има места даљем унапређењу.

На основу резултата описаног експеримента над валидационим скупом, може се закључити да је имплементирана неуронска мрежа способна да научи положаје референтних тачака.

Неуронска мрежа даје добре резултате над генерисаним као и стварним рендгенским снимцима који су сличних карактеристика инстанцама генерисаног скупа. То пре свега значи да се на стварном рендгенском снимку види карлица у целости.

Проблематични стварни рендгенски снимци су они на којима се појављује заштитник од радијације и/или се не види део карлице (уобичајено горњи део када се снима врх фемура превасходно).

У наставку пројекта препоручени путеви даљег рада би били унапређење модела, унапређење процеса обучавања као и додатна аугментација скупа за обучавање. Од предложених модуса даљег рада очекује се да би највећи допринос дало генерисање рендгенских снимака који би имали карактеристике стварних рендгенских снимака на којима предикција није била задовољавајућа. То су пре свега инстанце са заштитником од радијације као и инстанце на којима недостаје део карлице на снимку.



Слика 1: Пример успешне предикције над инстанцом реалног рендгенског снимка

6. ЛИТЕРАТУРА

- [1] Y. Chen, X. Ee, W. K. Leow, и T. S. Howe, „Automatic extraction of femur contours from hip x-ray images“, *Comput. Vis. Biomed. Image*, стр. 1–10, Окт. 2005.
- [2] D. Cristinacce и T. Cootes, „Automatic feature localisation with constrained local models“, *Pattern Recognit.*, том 41, изд. 10, стр. 3054–3067, Окт. 2008.
- [3] R. Smith, K. Najarian, и K. Ward, „A hierarchical method based on active shape models and directed Hough transform for segmentation of noisy biomedical images; Application in segmentation of pelvic X-ray images“, *BMC Med. Inform. Decis. Mak.*, том 9, изд. SUPPL. 1, стр. S2, Нов. 2009.
- [4] C. Chen, W. Xie, J. Franke, P. A. Grützner, L.-P. Nolte, и G. Zheng, „Fully automatic X-ray image segmentation via joint estimation of image displacements“, *Med. Image Comput. Comput. Assist. Interv.*, том 16, изд. Clm, стр. 227–34, 2013.
- [5] C. Lindner, S. Thiagarajah, J. M. Wilkinson, G. A. Wallis, и T. F. Cootes, „Accurate Fully Automatic Femur Segmentation in Pelvic Radiographs Using Regression Voting“, у *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, том 15, изд. Pt 3, Springer, Berlin, Heidelberg, 2012, стр. 353–360.
- [6] C. Lindner, S. Thiagarajah, J. M. Wilkinson, G. A. Wallis, и T. F. Cootes, „Fully automatic segmentation of the proximal femur using random forest regression voting“, *IEEE Trans. Med. Imaging*, том 32, изд. 8, стр. 1462–1472, Авг. 2013.
- [7] B. Aubert, C. Vazquez, T. Cresson, S. Parent, и J. De Guise, „Automatic spine and pelvis detection in frontal X-rays using deep neural networks for patch displacement learning“, у *Proceedings - International Symposium on Biomedical Imaging*, 2016, том 2016–June, стр. 1426–1429.
- [8] F. C. Ghesu, B. Georgescu, T. Mansi, D. Neumann, J. Hornegger, и D. Comaniciu, „An artificial agent for anatomical landmark detection in medical images“, у *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016, том 9902 LNCS, стр. 229–237.
- [9] D. Yang, S. Zhang, Z. Yan, C. Tan, K. Li, и D. Metaxas, „Automated anatomical landmark detection on distal femur surface using convolutional neural network“, у *Proceedings - International Symposium on Biomedical Imaging*, 2015, том 2015–July, стр. 17–21.
- [10] R. Sa u ocmanu, „Intervertebral disc detection in X-ray images using faster R-CNN“, у *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, 2017, том 2017, стр. 564–567.
- [11] J. Zhang, M. Liu, и D. Shen, „Detecting Anatomical Landmarks from Limited Medical Imaging Data Using Two-Stage Task-Oriented Deep Neural Networks“, *IEEE Trans. Image Process.*, том 26, изд. 10, стр. 4753–4764, Окт. 2017.
- [12] N. Tajbakhsh и ocmanu, „Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?“, *IEEE Trans. Med. Imaging*, том 35, изд. 5, стр. 1299–1312, Мај 2016.
- [13] Y. Gao, Y. Shao, J. Lian, A. Z. Wang, R. C. Chen, и D. Shen, „Accurate Segmentation of CT Male Pelvic Organs via Regression-Based Deformable Models and Multi-Task Random Forests“, *IEEE Trans. Med. Imaging*, том 35, изд. 6, стр. 1532–1543, Јуни 2016.

Кратка биографија:



Радован Туровић рођен је у Врбасу, у Републици Србији 1992. године. Мастер рад на Факултету техничких наука из области Електротехнике и рачунарства – Рачунарства и аутоматике одбранио је у 2018. години.

**AUTOMATSKO TESTIRANJE CVC DMS FUNKCIJE IMPELEMENTIRANE U ADMS
SOFTVER****TESTS AUTOMATIZATION FOR CVC FUNCTION IMPLEMENTED IN ADMS
SOFTWARE**

Miloš Kajkut, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je izvršeno automatsko testiranje funkcije Centralizovane regulacije napona (CVC). Funkcija Centralizovane regulacije napona vrši regulaciju napona u distributivnim mrežama u kojima postoje distribuirani generatori. Funkcija je implemantirana u ADMS softver, koji je bio predmet automatskog testiranja. Testiranje je sprovedeno u nekoliko faza: prva faza obuhvata formiranje osnove koja će biti predmet testiranja, druga faza obuhvata odabir nivoa testiranja i tehnike testiranja, treća faza obuhvata dizajn i kreiranje manualnih testova, četvrta faza obuhvata automatizaciju manualnih testova i njihovo izvršavanje.

Abstract – In this paper, the automatization of manual test for the Centralized Voltage Control (CVC) function was performed. Main role of Centralized Voltage Control function is to regulate voltage in distribution power network with presence of distributed generators. Centralized Voltage Control function was implemented in ADMS software, which was tested with automated tests. Test process was implemented in four phases. In first phase, technical documents and design documentation were formed. In second phase. Level of testing and test design technique, which were used in process of testing, were defined. Design and creation of Test Cases were implemented through the third phase. In final, fourth phase, test automatization and execution of automated tested were done.

Cljučne reči: regulacija napona, centralizovana regulacija napona, testiranje softvera, sistemsko testiranje, automatsko testiranje.

1. UVOD

Iako je pojava softvera omogućila značajnu automatizaciju mnogih profesija, profesija softver inženjera je i dalje ostala dužnost čoveka. Tokom svog rada, čovek kao softverski inženjeri pravi greške. Posledice tih grešaka su greške u softveru [1].

Na primer, ako programer zaboravi da postoji mogućnost unosa neograničeno velikog niza slovnih znakova u proizvoljnom polju na ekranu računara, njegov softver ima nedostatak – grešku. Kada se takav softver pokrene i unese prevelik niz slovnih znakova, greška će se manifestovati kao neočekivano ponašanje softvera.

Greška softvera je u tome što umesto da odbije on prihvata prevelik niz slovnih znakova.

Testiranje softvera može da dovede do značajnog umanjenja rizika od pojave devijacije rezultat i usluge softvera koji se testira [1]. Ako je softver i sve njegove faze životnog ciklusa (definisane zahteva koje softver treba da ispuni, dizajn, pisanje tehničke dokumentacije, itd.) predmet rigoroznog i sistematskog testiranja test inženjera, tada test inženjer ima veću šansu za pronalaženja grešaka u određenim fazama. Nakon otkrivanja grešaka u softveru, može se preći na fazu pronalaska i uklanjanja uzroka koje su dovele do devijacije i pogrešnih rezultata. Testiranje softvera ne menja kvalitet softvera, koji se testira, direktno [1]. Kada se tokom procesa testiranja pronađe greška u softveru i uzrok greške se ispravi, kvalitet softvera se povećava. Zaključak je da proces pronalaženja i uklanjanje uzroka greške povećava kvaliteta softvera, proces testiranja direktno ne povećava kvalitet softvera [1].

Rad se sastoji od pet celine. U prvoj glavi dat je uvod u rad. U drugoj glavi predstavljen je V model [1,2,3], takođe su predstavljeni i nivoi testiranja, tipovi testiranja, test dizajn tehnike, način praćenja i kontrole procesa testiranja, model automatskog testiranja, tehnička i dizajn dokumentacija i specifikacija funkcije Centralizovane regulacije napona [4,5,6,7]. U trećoj glavi predstavljen je model izvoda distributivne mreže nad kojim je izvršeno automatsko testiranje, takođe su predstavljeni modeli automatskih testova i rezultati izvršavanja testova. U četvrtoj glavi dat je zaključak. Na kraju, u petom delu data je literatura koja je korišćena prilikom realizacije ovog rada.

2. AUTOMATSKO TESTIRANJE

U ovom delu predstavljeni su nivo testiranja i test dizajn tehnike koje su korišćene prilikom realizacije rada.

2.1 Testiranje tokom životnog ciklusa softvera

V model [1] je razvijen kako bi ispravio nedostatke tradicionalnog modela „vodopada“. Glavni problem modela „vodopada“ [1] je što se greške detektuju kasno, zbog toga što je testiranje primenjeno na samom kraju procesa razvoja softvera. Kroz V model, testiranje softvera počinje u najranijim fazama razvojnog procesa softvera, što je jedan od osnovnih principa struktuiranog testiranja. Kroz V model se može primetiti da postoje mnoge aktivnosti, vezane za testiranje, koje moraju da se izvrše pre završetka faze pisanja izvornog koda softvera. Aktivnosti treba da se sprovedu paralelno sa procesom

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji je mentor bio dr Goran Švenda, red. prof.

razvoja softvera, kako bi se aktivnosti uspešno izvršile neophodno je da test inženjeri sarađuju sa programerima i biznis analitičarima. Produkt rada programera i biznis analitičara je osnova za testiranje u narednom ili nekom od narednih nivoa V modela.

V model je model koji na najbolji način ilustruje kako aktivnosti vezane za testiranje (verifikacija i validacija) mogu da se integrišu u svaku fazu procesa razvijanja softvera. U okviru V modela, validacija se odvija u ranim nivoima procesa razvoja softvera (pregled klijentskih zahteva), dok se u poslednjim nivoima procesa razvoja vrši testiranje koje treba da da ocenu da li je softvera prihvatljiv u pogledu zahteva koje je trebao da ispuni.



Slika 2.1.1 – V Model [1]

Postoje različite varijacije V modela, uobičajeni V model sadrži četiri nivoa testiranja. Svaki nivo testiranja ima svoj cilj, ti ciljevi su [1]:

- Komponentno testiranje: pronalaženje grešaka i verifikacija funkcionalnosti softverskih komponenti (moduli, programi, objekti, klase itd.) koje se testiraju odvojeno;
- Integraciono testiranje: testiranje razmene informacija između različitih komponenti softvera, interakciju između različitih delova sistema kao što su operativni sistem, baze podataka i hardvera;
- Sistemsko testiranje: uzima u obzir ponašanje celokupnog softvera, koje je definisano kroz proces razvoja softvera. Fokus sistemskog testiranja je verifikacija ponašanja softvera u skladu sa specifičnim zahtevima klijenta.
- Prihvatljivo testiranje: validaciono testiranje u skladu sa potrebama i zahtevima klijenta, kao i biznis procesa. Cilj prihvatljivog testiranja je da pruži krajnju ocenu da li softver ispunjava sve zahteva i potrebe klijenta.

U praksi V model može da ima više ili manje različitih nivoa razvoja i testiranja softvera u zavisnosti od projekta. Nivoi testiranja mogu da se kombinuju i reorganizuju u zavisnosti od zahteva projekta i arhitekture softvera koji se realizuje.

2.3 Test dizajn tehnike

Tehnika crne kutije može da se podeli na pet kategorije [1]:

- Tehnika ekvivalentne podele;

- Tehnika analize graničnih vrednosti;
- Tehnika tabele odluka;
- Tehnika dijagrama stanja;
- Tehnika korisničkog slučaja.

Tehnika ekvivalentne podele – Može da se primeni na bilo kom nivou testiranja, i obično se koristi kao prva tehnika prilikom pisanja testova. Tehnika ekvivalentne podele pruža mogućnost da se testira samo jedan uslov iz svakog skupa osnova za pisanje testova. To je moguće zbog toga što će svi uslovi iz jednog skupa biti tretirani na isti način od strane softvera. Ukoliko se softver prema jednom uslovu iz nekog skupa ponaša očekivano, onda nema svrhe testirati i ostale uslove iz tog skupa i može da se zaključi da će se softver i prema ostalim uslovima, iz tog skupa, ponašati isto. Obrnuto, ako se softver prema jednom uslovu ne ponaša kako je predviđeno, onda se može zaključiti da se softver neće ponašati predviđeno ni prema ostalim uslovima iz posmatranog skupa.

Tehnika graničnih vrednosti – Ova tehnika se zasniva na testiranju graničnih vrednosti između definisanih ekvivalentnih grupa.

Tehnika tabele odluka – dobar način za rešavanje problema različite kombinacije ulaznih parametara. Ova tehnika se naziva i tehnika „uzrok – posledica“, kao posledica postojanja logičkog dijagrama ili „uzrok – posledica dijagram“ koji ponekad služi za pomoć prilikom primene tehnike tabele odluka. Tabela odluka omogućava sistematski način definisanja složenih poslovnih pravila, što je korisno i za programere i za test inženjere. Tabele odluka mogu biti korišćenje u procesu dizajna testova, bez obzira da li su one navedene u specifikacijama, jer pomažu test inženjerima da istraže efekte različitih kombinacija ulaznih parametara ili provere da li različite funkcionalnosti pravilno primenjuju pravila poslovanja

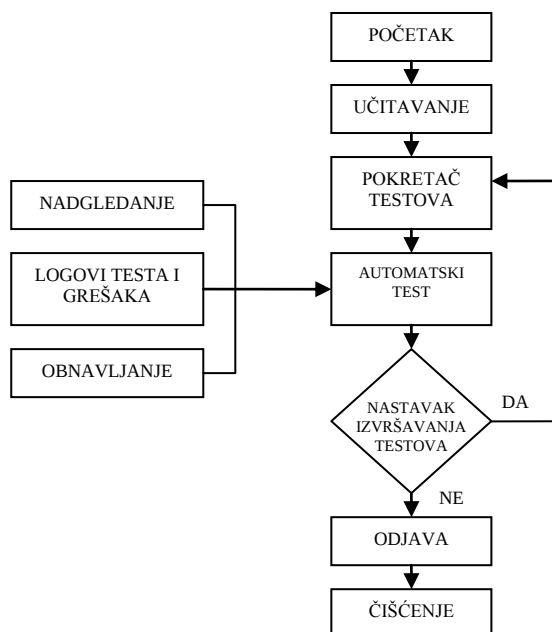
Tehnika dijagrama stanja – najčešće se koristi u softveru čiji rad može da se opiše kroz „mašinu stanja“. „Mašina stanja“ može da se definiše kao konačan broj različitih stanja u kojima softver može da se nađe, gde je svaka tranzicija iz jednog stanja u drugo jasno definisana kroz pravila „mašine“. Odnosno, softver kod koga se za iste ulazne parametre mogu očekivati različiti izlazni parametri, u zavisnosti od stanja pre prosleđivanja ulaznih parametara, se definiše kao jedno stanje softvera u „mašini stanja“. Dijagram stanja ili „mašina stanja“ je model na kome će se bazirati rad softvera, ali i testiranje softvera.

2.4 Zajedničke funkcije

Zajedničke funkcije su rutine koje svaki automatski test prati tokom njegovog izvršavanja, ali i prilikom njegovog razvijanja. Neke funkcije mogu biti sadržane u svim automatskim testovima, npr. funkcije koje služe za oporavak od neočekivanih događaja. Ispisivanje logova i rezultata izvršavanja. Ove funkcije su najčešće realizovane kroz podprograme, što omogućava njihovo pozivanje iz bilo kog dela automatskog testa.

Drugi tip zajedničkih funkcija su skripte, npr. osvežavanje baze podataka ili popunjavanje baze podataka sa poznatim vrednostima. brisanje privremenih datoteka koje se koriste prilikom izvršavanja automatskog testa.

Zajedničke funkcije bi trebale biti implemetirane tako da ih je moguće izvršiti pojedinačno (nezavisno od automatskog testa), sekvencijalno ili integrisano kroz automatski test [3].



Slika 2.4.1 – Algoritam zajedničkih funkcija [3]

POČETAK – Ova funkcija priprema okruženje u kojem će se izvršavati automatski test. Pokreće se pre izvršavanja svakog automatskog testa u svrhu utvrđivanja da li je odgovarajuća verzija softvera instalirana, da li je softver ispravno konfigurisan, sve datoteke neophodne za izvršavanje testa su dostupna. U radu, ovaj deo se radi manuelno. Zbog kompleksnosti problema provere verzije softvera i konfiguracija softvera, provera se izvršava manuelnom inspekcijom.

UČITAVANJE – Funkcija pokreće aplikaciju ADMS softvera i uverava se da li je aplikacija dostupna za izvršavanje automatskih testova nad njom. Funkcija obezbeđuje validno korisničko ime i šifru kako bi se aplikacija pokrenula.

POKRETAČ AUTOMATSKIH TESTOVA i TESTOVI – Sadrži niz podprograma koji izvršavaju testiranje funkcije, implementirane u ADMS softveru. Podprogrami koji se izvršavaju biće opisani u nastavku rada.

NADGLEDANJE – Funkcija vrši nadgledanje statusa softvera nakon izvršavanja svakog podprograma iz automatskog testa kako bi se dobila povratna informacija da li je izvršeni podprogram doveo softver u nekonzistentno stanje.

OBNAVLJANJE – U radu ova funkcija nije automatizovana i izvršava se manuelno zbog velike kompleksnosti same akcije obnavljanja. Indikator funkcije su logovi grešaka, koji pružaju informaciju da se desila neočekivana akcija ili događaj. Kada se tokom izvršavanja automatskog testa desi neočekivan događaj ili akcija, izvršavanje testa se završava uz poruku koja opisuje neočekivani događaj ili akciju. Obnavljanje ili vraćanje na prethodno stanje se izvršava manuelno, nakon što se utvrdi uzrok događaja ili akcije.

ODJAVA – Obrnuta funkcija od „Učitavanja“, funkcija obezbeđuje da se softver vrati u poslednje poznato stanje pre izvršavanja automatskog testa. Ova funkcija se izvršava na kraju procesa izvršavanja automatskog testa, pre „Čišćenja“.

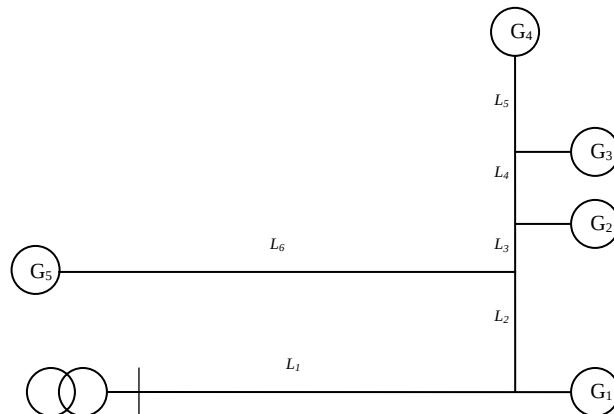
LOGOVI TESTA – Funkcija se poziva prilikom izvršavanja svakog podprograma tokom procesa izvršavanja automatskog testa. Funkcija u posebnu datoteku zapisuje rezultate izvršavanja automatskog testa, rezultate izvršavanja svakog podprograma, vreme izvršavanja svakog podprograma.

LOGOVI GREŠAKA – Funkcija se poziva prilikom svakog neuspešnog izvršavanja automatskog testa. Nakon pozivanja, funkcija u posebnu datoteku upisuje stanje u kojem je bio softver, vreme kad se softver neuspešno izvršio. Razlozi zbog kojih automatski test nije uspešno izvršen mogu biti različiti, među njima može biti i postojanje greške u softveru (automatski test treba da je otkrije). Zbog toga, potrebno je obezbediti neophodne informacije o tome koje akcije su izvršene neposredno pre manifestacije greške u softveru, kao i uslovi u kojima se greška manifestovala.

ČIŠĆENJE – Funkcija uklanja posledice izvršavanja podprograma unutar procesa izvršavanja automatskog testa.

3. VERIFIKACIJA MODELA ZA AUTOMATSKO TESTIRANJE

Automatski testovi izvršavani su nad srednje naponskim izviđanjem koji je prikazan na Slika 3.1.



Slika 3.1 – SN izvod nad kojim su izvršeni automatski testovi

U Tabela 3.1 prikazani se vrednosti reaktivni svih sekcija koje pripadaju posmatranom izvodu nad kojim će se izvršavati testovi.

Tabela 3.1 – Vrednosti reaktansi sekcija

Sekcija	$X[\Omega]$
X_{L1}	0.306
X_{L2}	0.303
X_{L3}	0.179
X_{L4}	0.161
X_{L5}	0.084
X_{L6}	0.215

U Tabela 3.2 prikazani se podaci o generatorima koji pripadaju posmatranom izvodu nad kojim će se izvršavati testovi

Tabela 3.2 – Podaci o generatorima na SN izvodu

	$Q_{ind}[kVar]$	$Q_{cap}[kVar]$	$\cos\varphi_{ind}$	$\cos\varphi_{cap}$
G1	400	300	0.8	0.9
G2	500	400	0.8	0.9
G3	450	320	0.9	0.9
G4	450	380	0.8	0.9
G5	400	300	0.8	0.9

Ocena ispravnosti i pouzdanosti CVC funkcije implementirane u AMDS softveru

U ciklusu automatskog testiranja izvršeni su automatski testovi koji imaju najveći prioritet, kako bi se dobila povratna informacija o pouzdanosti i ispravnosti CVC funkcije. Automatski testovi koji su se izvršavali u ciklusu automatskog testiranja testiraju najvažnije specifikacije i zahteve koje funkcija treba da ispuni. Zbog svega navedenog, nije usvojen kriterijum koji bi procentualno ili kvantitativno opisao potreban broj uspešno izvršenih automatskih testova na osnovu kojih bi bila data ocena da li funkcija zadovoljava i ispunjava sve zahteve. Uspešno izvršavanje svih automatskih testova, je jedini kriterijum koji treba da bude ispunjen kako bi CVC funkcija dobila pozitivnu ocenu o pouzdanosti i ispravnosti.

Konačna ocena je da CVC funkcija ispunjava sve zahteve i da je implementirana u softver prema specifikacijama i dizajnu. Nisu primećeni problemi i greške u radu funkcije prilikom izvršavanja ciklusa automatskog testiranja. Takođe nisu prijavljene greške u radu i implementaciji funkcije. Funkcija je pouzdana, ispravna i spremna je za dalju isporuku klijentu. Može se smatrati da će funkcija ispunjavati sve zahteve pri radu u produkciji.

4. ZAKLJUČAK

Realizacijom automatskog testiranja kroz rad ostvareni su benefiti koji nisu mogli biti ostvareni kroz manuelno testiranje. Ostvareni benefiti su:

- Ponovno pokretanje automatskih testova na novijim verzijama softvera. Ako nove verzije softvera obuhvataju ili ne obuhvataju izmene nad CVC funkcijom: Automatski testovi koji testiraju CVC funkciju mogu biti ponovo izvršeni na novijim verzijama softvera.
- Ukoliko se dizajn i automatski testovi nižih prioriteta, moguće je češće izvršavanje više testova (svih nivoa prioriteta).
- Bolje korišćenje resursa, koje podrazumeva manji broj test inženjera uključenih u proces automatskog testiranja funkcije, redukovanje vremena potrebnog za testiranje i smanjenje finansijskih zahteva za izvršavanje procesa testiranja.

- Nakon automatizacije manualnih testova postoji mogućnost za izvršavanjem automatizovanih testova u bilo kom vremenskom periodu i neograničen broj puta.

Kao jedan od glavnih problema, nakon automatizacije, izdvaja se mogućnost netačnih rezultat izvršavanja testova. Moguća su sledeća scenarija:

- Lažno neuspešno izvršavanje testa koje nije posledica greške u softveru ili greške prilikom automatizacije testa.
- Neuspešno izvršavanje kao posledica izmene u automatskom testu ili loša implementacija automatskog testa.
- Prijavljivanje iste greške nekoliko puta može dovesti do loših rezultata u konačnom izveštaju o ciklusu automatskog testiranja. Jedna greška u softveru može biti registrovan kroz nekoliko ili sve automatske testove.
- Lažno uspešno izvršavanje automatskog testa kao posledica u lošem dizajnu ili implementaciji automatskog testa.
- Lažno uspešno izvršavanje automatskog testa kao posledica nepravilnog proveravanja moguće greške u softveru.

5. LITERATURA

- [1] R.Black, Erik Van Veenendaal, Dorothy Graham, Foundations of Software Testing 3rd Edition, South – Western, 2009.
- [2] M.Ferwster, Dorothy Graham, Software Test Automation – Effective use of test execution tools, ACM Press New York
- [3] L.G.Hayes, The Automated Testing Handbook, Software Testing Institute, 2004.
- [4] B.Bakhshideh Zad, J.Lobry,F.Vallee, A centralized approach for voltage control of MV distribution systems using DGs power control and a direct sensitivity analysis method, University of Mons, Belgium, 2016.
- [5] M.Berenna, E.De Berardinis, F.Foidaelli, G.Sapienza, D.Zaninelli, Voltage Control in Smart Grids: An Approach Based on Sensitivity Theory, J. Electromagnetic Analysis & Applications, 2010, 2, 467-474
- [6] S.Conti, A.M.Greco, S.Raiti, Voltage Sensitivity Analysis in MV Distribution Network, University of Catania, Catania, 2006.
- [7] G.Sapienza, Voltage Control in MV networks whit high DG presence a real-time digital simulator, Politecnico di Milano, Milano, 2012.

Kratka biografija:



Miloš Kajkut rođen je u Apatinu 1993. godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Elektroenergetski sistemi odbranio je 2016. god. Iste godine upisao se na master studije na istom smeru.

HIBRIDNA PARALELIZACIJA FILTRIRANJA 2D SLIKA

HYBRID PARALLELIZATION OF 2D IMAGE PROCESSING

Jelena Janković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je prezentovano jedno rešenje hibridne paralelizacije filtriranja 2D slika. Filtriranje je odrađeno pomoću Gausovog i Sobelovog algoritma za filtriranje 2D slika. Implementirana je paralelna verzija ovih algoritama korišćenjem CUDA platforme za grafičke procesore. Obezbeđena je distribuirana arhitektura uz primenu MPI biblioteke. Ukazane su prednosti implementiranog hibridnog paralelnog rešenja u odnosu na sekvencijalno.

Abstract – In this paper we present a single solution of the hybrid parallelization of 2D image processing. Image processing was done using Gaussian and Sobel 2D image filtering algorithm. A parallel version of these algorithms has been implemented using the CUDA platform for graphic processors. Distributed architecture is provided with application of MPI library. The advantages of the implemented hybrid parallel solution versus sequential are shown.

Ključne reči: Filtriranje, MPI, CUDA, paralelizacija, hibridna arhitektura

1. UVOD

Kako su fizički limiti tehnologije izrade računarskog hardvera skoro dostignuti, paralelizacija dobija na sve većem značaju kada je u pitanju brzina i efikasnost obrade podataka. Grafički procesori imaju sve veću primenu kada su u pitanju izračunavanja opšte namene, pa su mnoge aplikacije prenete da koriste grafičke procesore za intenzivnija izračunavanja.

Time su postignute brzine nekoliko redova veličine veće u poređenju sa ekvivalentnim sekvencijalnim implementacijama napisanim za obične procesore. U ovom radu implementiran je algoritam 2D konvolucije primenom Gausovog i Sobelovog filtera.

Algoritam konvolucije je podložan za paralelizaciju i može se razložiti na podprobleme koji ne zavise jedni od drugih i koji se mogu istovremeno izvršavati na različitim procesorima.

Hibridni pristup predstavljen u ovom radu kombinuje CPU i GPU računanja koristeći MPI biblioteku i CUDA platformu. Naime, razvijeno hibridno rešenje koristi MPI biblioteku kako bi se omogućilo širenje računanja između računarskih čvorova u klasteru, dok CUDA platforma služi za obavljanje lokalnih izračunavanja na svakom čvoru klastera. U 2. glavi uvedeni su pojam digitalne slike i pojam obrada digitalne slike.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Bogdan Satarić, docent.

Objašnjeno je filtriranje, kao i algoritam 2D konvolucije primenom Gausovog i Sobelovog filtera. Predstavljena je reprezentacija ulazno-izlaznih podataka, kao i format slike koji se koristi u ovom radu.

U 3. glavi opisano je razvojno okruženje koje je korišćeno za izradu ovog rada. Takođe, ova glava ima za svrhu da objasni šta je paralelno programiranje, interfejs za prosleđivanje poruka, model prosleđivanja poruka, MPI biblioteka i njene prednosti, MPI podržane tipove podataka, kao i osnovne pojmove i koncepte CUDA platforme. Objasnjeno je šta predstavlja CUDA programski model, kako se organizuju niti na CUDA platformi, arhitektura CUDA programa i organizacija memorije u CUDA grafičkim uređajima.

U 4. glavi prikazana je implementacija hibridnog rešenja predstavljenog u ovom radu. Prikazan je programski kod napisan primenom MPI biblioteke i CUDA C programskog jezika.

U 5. glavi sumirane su najvažnije karakteristike implementiranog rešenja. Takođe, u ovoj glavi predloženi su i pravci budućeg rada.

2. OPIS REŠAVANOG PROBLEMA

2.1 Digitalna slika i obrada digitalne slike

Pojam digitalna slika obuhvata sve medije koje ljudi mogu da vide, kao što su slika, video, crteži, grafikoni, animacije i drugi mediji [1]. Pošto se veći broj informacija iz stvarnog sveta može dobiti putem digitalnih slika, velika pažnja se posvećuje obradi digitalnih slika [2]. Obrada digitalne slike predstavlja postupak menjanja detalja na nekoj digitalnoj slici. Kako bi se bolje prikazala slike ili videlo više detalja na njima, razvijene su razne tehnike obrade digitalnih slika koje se odnose na menjanje, poboljšavanje i promenu veličine i rezolucije slike.

Metode za obradu slike mogu se podeliti na operacije na pikselima (engl. point operations) i prostorne operacije (engl. spatial operations). U ovom radu biće diskutovane prostorne operacije.

Prostorne operacije predstavljaju memorijske operacije i često se izvode računanjem konvolucije između slike i filtera sa konačnim impulsnim odzivom (maska). Prostorne operacije se mogu podeliti na prostorno usrednjavanje, uklanjanje zamućenja (šuma), srednje filtriranje, filtriranje i interpolaciju slika. U ovom radu posebna pažnja se posvećuje filtriranju slika.

2.2 Filtriranje

Filtriranje slike predstavlja najčešći način obrade slika i svodi se na primenu algoritma diskretne konvolucije [3]. To je postupak kojim se iz originalne slike, primenom željenog filtera, upotrebom odgovarajućeg matematičkog algoritma nad pikselima obrađivane slike, stvara nova

(rezultujuća) slika. Tipovi filtriranja slika opisani u ovom radu su Gausov i Sobelov algoritam za filtriranje 2D slika.

2.2.1 Konvolucija

Konvolucija kao algoritam obrade slika predstavlja nizovnu operaciju koja za svaku tačku slike koja se obrađuje daje izlaznu tačku čija vrednost zavisi od susednih tačaka i od konvolucione maske [4]. Centar maske se množi sa tekućim pikselom, a drugi elementi maske sa odgovarajućim susednim pikselima, nakon čega se dobijeni proizvodi saberu i dobije vrednost piksela rezultujuće slike [5]. U ovom radu je predstavljen problem uklanjanja šuma i problem detekcije ivica na slici kao i način rešavanja tih problema upotrebom Gausovog i Sobelovog operatora respektivno.

2.2.1.1 Gausov algoritam (engl. *Gaussian filter*)

Uklanjanje šuma predstavlja važan zadatak u okviru obrade i analize slike. Postoji velik broj različitih algoritama za uklanjanje šuma. Koji algoritam se koristi zavisi od tipa šuma koji je potrebno ukloniti. Postoje linearni i nelinearni algoritmi za uklanjanje šuma.

Gausov filter spada u klasu linearnih algoritama [6]. Ovaj filter je nazvan po tome što su koeficijenti filtera definisani preko Gausove funkcije (engl. Gaussian function):

$$h(n, k) = C \cdot e^{-\frac{(n-N/2)^2 + (k-N/2)^2}{2\sigma^2}},$$

Pri čemu je konstanta S određena tako da filter ima jedinično pojačanje:

$$\sum_{n=0}^{N-1} \sum_{k=0}^{N-1} h(n, k) = 1.$$

2.2.1.2 Sobelov algoritam (engl. *Sobel edge detection filter*)

U kontekstu obrade slike jedan od fundamentalnih koraka jeste detekcija ivica (rubova). Svrha detekcije ivica jeste otkrivanje objekata na slici, interpretacija sadržaja slike i prepoznavanje osobina onoga što je na slici prikazano [7]. Jedan od najpopularnijih algoritama za detekciju ivica je Sobelov algoritam [7]. Osnovna ideja na kojoj se bazira Sobelov algoritam jeste korišćenje gradijenta (prvog izvoda) za meru da li neki piksel pripada ivici ili ne [8]. U ovom algoritmu prvi izvod po x-osi predstavlja meru za vertikalnu ivicu, a izvod po y-osi predstavlja meru za horizontalnu ivicu [8].

Sobel operatori za detekciju ivica su:

$$h_v(v, h) = \frac{1}{4} \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad h_h(v, h) = \frac{1}{4} \begin{bmatrix} -1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

2.3 Reprezentacija ulazno-izlaznih podataka

Učitane slike su predstavljene u BMP formatu [9]. BMP format slika je najkvalitetniji format jer se nad ovim formatom ne vrši nikakva kompresija. Nakon što je slika učitana, ona se konvertuje pomoću modula za čitanje slika u internu strukturu koja predstavlja niz PIXEL struktura. PIXEL struktura sadrži za svaku komponentu boje po

jednu vrednost [5]. Kako je boja predstavljena pomoću RGB modela boja, posebno se beleže vrednosti za crvenu, posebno za zelenu i posebno za plavu komponentu boje [5].

3. OPIS KORIŠĆENIH ALATA

3.1 Razvojno okruženje

Kao okruženje za razvoj softvera, korišćen je operativni sistem Linux, biblioteka OpenMPI, editor gedit i kompajler nvcc. Za izradu zadatka u ovom radu korišćen je programski jezik C u kombinaciji sa CUDA i MPI bibliotekama.

3.2 Paralelno programiranje

Program koji je napisan za sekvencijalno izvršavanje na jednoprocorskim računarima izvršava instrukcije jednu za drugom i na taj način uzima puno procesorskog vremena. Nasuprot sekvencijalnoj obradi, paralelna obrada je zasnovana na principu da se veliki problem često može podeliti na manje, koji se zatim rešavaju istovremeno (paralelno, konkurentno) i na taj način se minimizuje vreme izvršavanja i obezbeđuje maksimalna iskorišćenost procesora.

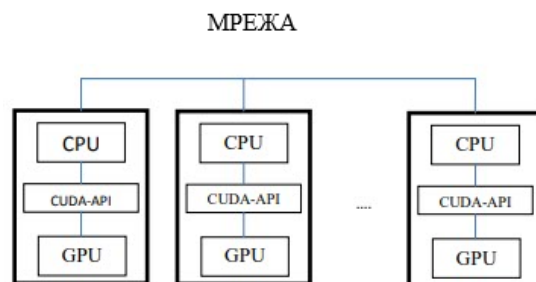
Hibridni pristup predstavljen u ovom radu predstavlja distribuirani GPU način računanja koristeći MPI biblioteku i CUDA platformu. CUDA kao platforma pogodna za paralelnu obradu slike danas postaje sve popularniji programski model za računanja na grafičkim procesorima. Takođe, algoritmi predstavljeni u ovom radu implementirani su za paralelno izvršavanje na klasteru pomoću biblioteke za prosleđivanje poruka MPI (engl. Message Passing Interface).

3.3 Interfejs za prosleđivanje poruka (engl. *MPI*)

Prosleđivanje, odnosno, komunikacija porukama je programska paradigma koja se široko koristi na super-računarima. Omogućava da programi komuniciraju pomoću procesa koji međusobno razmenjuju poruke. Poruke koje se razmenjuju predstavljaju podatke i zahteve. Ova tehnika se najčešće primenjuje kod paralelnih računara sa distribuiranom memorijom (super-računara).

MPI (engl. Message Passing Interface) je standardizovana i portabilna specifikacija biblioteke za razmenu poruka koju je dizajnirala grupa istraživača iz akademske zajednice i industrije, sa ciljem da funkcioniše na velikom broju paralelnih računara [11].

Kada je u pitanju hardver koji podržava hibridni pristup predstavljen u ovom radu, on se sastoji od skupa čvorova. Svaki čvor je predstavljen kao kombinacija centralnog i grafičkog procesora koji komuniciraju pomoću CUDA aplikacionog interfejsa (engl. API). Prethodno navedena hibridna arhitektura prikazana je na Slici 3.1.



Slika 3.1 Hibridna arhitektura MPI – CUDA programa.

3.3.1 Funkcionalnosti MPI biblioteke

3.3.1.1 Osnovne funkcije

MPI uključuje veliki broj funkcija, međutim, u nastavku je navedeno osnovnih 6 funkcija MPI biblioteke pomoću kojih se može postići većina funkcionalnosti:

- **MPI_Init()**
- **MPI_Comm_size()**
- **MPI_Comm_rank()**
- **MPI_Send()**
- **MPI_Recv()**
- **MPI_Finalize()**

MPI_Init - predstavlja MPI funkciju koja se poziva pre svih ostalih funkcija iz MPI biblioteke [11]. Inicijalizuje MPI okruženje tako što sistemu nalaže da uradi sva potrebna podešavanja paralelnog okruženja.

MPI_Finalize - predstavlja MPI funkciju koja se poziva na kraju procesa (nakon svih MPI funkcija) [11]. Dozvoljava sistemu da oslobodi sve resurse koje je MPI proces zauzeo prilikom izvršavanja MPI programa.

MPI_Comm_size - određuje ukupan broj procesa u komunikatoru (fiksni broj koji se zadaje pri pokretanju programa) [11].

MPI_Comm_rank - određuje rang procesa koji je pozvao ovu funkciju unutar prosleđenog komunikatora [11].

Kada je u pitanju razmena (prosleđivanje) poruka, osnovne funkcije za tu namenu su **MPI_Send** i **MPI_Recv**. Proces koji šalje poruku poziva funkciju **MPI_Send**, a proces koji prima poruku poziva funkciju **MPI_Recv**.

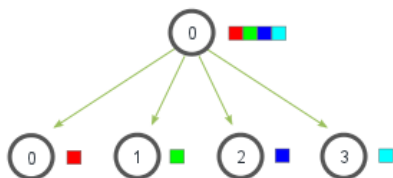
3.3.1.2 Međuprocesna komunikacija

3.3.1.2.1 Komunikacija između svih procesa komunikatora

Kako se u praksi javlja potreba za slanjem podataka svim procesima kao i potreba za skupljanjem podataka od više procesa, MPI nudi funkcije za komunikaciju između svih procesa u komunikatoru (engl. Collective communication) [11]. Jedna od osnovnih osobina kolektivne komunikacije jeste da uključuje sve procese u jednom komunikatoru.

Dva osnovna tipa kolektivne komunikacije implementirana u ovom radu su:

- Podela paketa podataka unutar grupe (engl. Scatter) - podrazumeva da se inicijalni podatak raspodeljuje na delove i svaki deo šalje različitom procesu iz komunikatora [12].

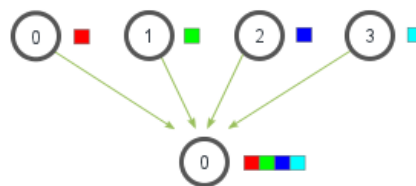


Slika 3.2 Prikaz rada MPI_Scatter funkcije.

Na Slici 3.2 se može videti da proces sa rednim brojem 0 deli paket podataka na delove koje zatim prosleđuje redom procesima 0, 1, 2 i 3.

- Objedinjavanje podataka od svih u grupi (engl. Gather) - predstavlja inverznu operaciju u

odnosu na Scatter [12]. Umesto razbijanja podataka na delove, ovaj tip kolektivne komunikacije uzima delove podataka od procesa i spaja ih u jedan podatak.



Slika 3.3 Prikaz rada MPI_Gather funkcije.

Slika 3.3 prikazuje objedinjavanje podataka od svih u grupi, gde su podaci predstavljeni kvadratićima različitih boja koji se objedinjuju u jedan podatak predstavljen kao niz čiji su elementi objedinjeni kvadratići različitih boja.

3.3.1.3 Grupe i komunikatori

Grupe i komunikatori predstavljaju objekte koje MPI koristi kako bi se moglo definisati koji procesi mogu međusobno da komuniciraju [11]. Grupa je uređeni skup od N procesnih identifikatora.

Komunikator je zatvoreni objekat koji obuhvata grupu procesa koji mogu međusobno da komuniciraju (razmenjuju poruke) [11]. To je kolekcija procesa koji mogu slati poruke jedni drugima.

Rang procesa je identifikator procesa unutar komunikatora [11]. Svrha ranga je adresiranje procesa, kako bi se mogli identifikovati procesi koji šalju i primaju poruke.

3.3.2 MPI podržani tipovi podataka

MPI tipovi podataka mogu se podeliti na:

- Osnovne
- Izvedene

Osnovni tipovi podataka odgovaraju osnovnim tipovima podataka koji se mogu pronaći u C programskom jeziku (**MPI_CHAR**, **MPI_SHORT**, **MPI_INT**, **MPI_LONG**, **MPI_UNSIGNED** – sve varijante, **MPI_FLOAT**, **MPI_DOUBLE**...).

Kada su u pitanju izvedeni tipovi podataka, MPI dozvoljava definisanje korisničkih struktura podataka baziranih na osnovnim MPI tipovima podataka. Izvedeni tipovi omogućavaju nekontinualnim i raznorodnim podacima da budu tretirani kao neprekidni [11].

3.4 CUDA – osnovni pojmovi i koncepti

CUDA (engl. Compute Unified Device Architecture) predstavlja hardversku i softversku računarsku arhitekturu namenjenu za paralelno izračunavanje [13]. Razvijena je od strane NVIDIA korporacije, koja je poznata po proizvodnji grafičkih procesora. To je programski model koji omogućava direktan pristup instrukcijama procesora kao i višestruko povećanje performansi izračunavanja efikasnim iskorišćenjem snage grafičke procesorske jedinice - GPU (engl. Graphical Processing Unit).

Implementacija rešenja predstavljenog u ovom radu je izvršena na NVIDIA GeForce GTX 970 grafičkoj kartici i na Intel(R) Xeon(R) CPU E5-1620 procesoru sa 32GB RAM.

Više informacija o CUDA platformi može se pronaći u radu [5].

4. PRIKAZ IMPLEMENTACIJE

Da bi se implementirala hibridna paralelizacija filtriranja 2D slika potrebno je primeniti sledeći niz koraka:

1. Deklarisanje i inicijalizacija svih promenljivih u programu
2. Alokacija potrebnog prostora u radnoj memoriji računara i memoriji grafičkog uređaja
3. Prenos podataka iz radne memorije računara u memoriju GPU
4. Inicijalizacija MPI okruženja
5. Kreiranje potrebnih MPI izvedenih tipova podataka (korisnički definisanih)
6. Podela paketa podataka unutar grupe (engl. Scatter)
7. Poziv i izvršavanje kernela
8. Prenos rezultata iz memorije GPU u radnu memoriju računara
9. Objedinjavanje podataka (delova slike) od svih u grupi (engl. Gather)

```
// Иницијализација MPI окружења.
MPI_Init(&argc, &argv);

// Декларација и иницијализација броја процеса.
int sizeMPI;
MPI_Comm_size(MPI_COMM_WORLD, &sizeMPI);

// Декларација и иницијализација ранга процеса.
int rank;
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
```

Figura 4.1 Inicijalizacija MPI okruženja.

Inicijalizacija MPI okruženja kao i deklaracija i inicijalizacija broja procesa i ranga svakog od procesa prikazane su na Figuri 4.1. Prvi MPI poziv u svakom MPI programu jeste poziv `MPI_Init(&argc, &argv)` funkcije. Ova funkcija sme biti pozvana samo jednom u programu i predstavlja inicijalizaciju MPI okruženja. Nakon inicijalizacije, svaki aktivan proces postaje član komunikatora. `MPI_COMM_WORLD` je standardni komunikator MPI biblioteke.

Pošto je efikasnost izvršavanja programa bitna stavka i ona u velikoj meri zavisi od broja procesa koji izvršavaju program, MPI biblioteka nudi `MPI_Comm_size(MPI_Comm comm, int *size)` funkciju koja određuje broj procesa koji se nalaze u komunikatoru. Funkcija `MPI_Comm_rank(MPI_Comm comm, int *rank)` dodeljuje svakom procesu jedinstven identifikator (rang).

Kao što je navedeno u poglavlju 2.3 **Reprezentacija ulazno-izlaznih podataka**, struktura kojom je opisana svaka tačka učitane slike predstavljena je na Figuri 4.2.

```
// PIXEL структура за смештање по једне
// вредности за сваку компоненту боје.
typedef struct {
    unsigned char R;
    unsigned char G;
    unsigned char B;
} PIXEL;
```

Figura 4.2 PIXEL struktura.

Da bismo mogli koristiti pomenutu strukturu i u pozivima MPI funkcija, moramo kreirati MPI tip podataka koji odgovara strukturi `PIXEL`. MPI biblioteka pruža mehanizam za definisanje korisničkih struktura podataka baziranih na prostim MPI tipovima podataka.

U ovom radu kreirana su dva izvedena tipa podataka: strukturirani i neprekidni (kontinualni). Postupak kreiranja korisnički definisanih tipova podataka implementiranih u ovom radu prikazan je na Figuri 4.3.

```
// Креирање MPI типа података који представља структуру PIXEL.
const int nitems = 3;
int blocklengths[3] = { 1, 1, 1 };
MPI_Datatype types[3] = { MPI_UNSIGNED_CHAR, MPI_UNSIGNED_CHAR, MPI_UNSIGNED_CHAR };
MPI_Datatype mpi_pixel_type;
MPI_Aint offsets[3];

offsets[0] = offsetof(PIXEL, R);
offsets[1] = offsetof(PIXEL, G);
offsets[2] = offsetof(PIXEL, B);

MPI_Type_create_struct(nitems, blocklengths, offsets, types, &mpi_pixel_type);
MPI_Type_commit(&mpi_pixel_type);

// Креирање MPI типа података који представља низ од newSize
// узастопних елемената mpi_pixel_type типа.
MPI_Datatype typel;
MPI_Type_contiguous(newSize, mpi_pixel_type, &typel);
MPI_Type_commit(&typel);
```

Figura 4.3 Kreiranje MPI izvedenih tipova podataka.

Figura 4.3. prikazuje tri poziva MPI funkcija: `MPI_Type_create_struct()`, `MPI_Type_commit()` i `MPI_Type_contiguous()` i njihovi argumenti će biti detaljnije objašnjeni u nastavku.

`MPI_Type_create_struct(int nitems, const int blocklengths[], const MPI_Aint offsets[], const MPI_Datatype types[], MPI_Datatype *newtype)` je funkcija koja vraća u `newtype` novi tip podataka koji se sastoji od `nitems` blokova, pri čemu se svaki blok i sastoji od `blocklengths[i]` elemenata koji su tipa `types[i]`. `MPI_Aint` je predefinisani tip podataka koji može da čuva memorijske adrese.

`MPI_Type_commit(MPI_Datatype *datatype)` je funkcija koja omogućava da "obavestimo" MPI da je korisnički definisan tip podataka kreiran i da planiramo da ga koristimo u komunikaciji.

`MPI_Type_contiguous(int count, MPI_Datatype oldtype, MPI_Datatype *newtype)` je funkcija koja vraća u `newtype` novi tip podataka. Novi tip podataka se stvara repliciranjem tipa `oldtype` na `count * sizeof(oldtype)` uzastopnih memorijskih lokacija.

U nastavku su prikazane definicije funkcija `MPI_Scatter()` i `MPI_Gather()` koje omogućavaju kolektivnu komunikaciju. Figura 4.4. i Figura 4.5. predstavljaju definicije `MPI_Scatter()` i `MPI_Gather()` funkcija, respektivno.

```
// Подела пакета података (слике) унутар групе.
MPI_Scatter(h_image, 1, typel, receivePixel, 1, typel, 0, MPI_COMM_WORLD);
```

Figura 4.4 Definicija MPI_Scatter funkcije.

```
// Обједињавање података (делова слике) од свих у групи.
MPI_Gather(receivePixel, 1, typel, h_image, 1, typel, 0, MPI_COMM_WORLD);
```

Figura 4.5 Definicija MPI_Gather funkcije.

Kao što je već navedeno, svi pozivi MPI funkcija se moraju nalaziti između poziva funkcija `MPI_Init()` i `MPI_Finalize()`. Stoga, poziv `MPI_Finalize()` funkcije predstavlja poslednji MPI poziv u svakom MPI programu. Njegova svrha je da oslobodi memoriju MPI. Figura 4.6 prikazuje oslobađanje memorije zauzete prilikom

izvršavanja MPI programa pomoću MPI_Finalize() funkcije.

```
// Ослобађање ресурса које је MPI процес  
// заузео приликом извршавања MPI програма.  
MPI_Finalize();
```

Figura 4.6 Poziv MPI_Finalize funkcije.

5. ZAKLJUČAK

Kako pristup paralelnom programiranju dobija sve veći značaj, u ovom radu prikazana je implementacija hibridne paralelizacije filtriranja 2D slika. Naime, ovaj rad predstavlja samo implementaciju hibridne paralelizacije filtriranja 2D slika. Performanse hibridnog rešenja u realnoj primeni na klasteru nisu testirane iz razloga što nije jednostavno pronaći infrastrukturu klastera na kome bi dato testiranje moglo biti obavljeno. Data infrastruktura postoji u okviru drugih istraživačkih centara u Srbiji (npr. Institut za fiziku u Zemunu), dok je na Fakultetu tehničkih nauka tek u nastanku.

Pošto podela posla između čvorova u klasteru dovodi do uzitnjavanja računanja koje svaki od čvorova treba da obavi, pretpostavlja se da bi to usitnjavanje doprinelo poboljšanju performansi i brzine izvršavanja pomenutog hibridnog rešenja. Za razliku od obrade sliken na jednom računaru gde je to izazovan posao, podela obrade slike između čvorova u klasteru omogućava da velike slike i veliki setovi slika (npr. hiljade slika) mogu biti relativno brzo obrađeni.

Kada su u pitanju pravci daljeg razvoja, sledeće stavke bi mogle biti uzete u obzir:

- Implementacija i testiranje hibridnog pristupa na različitim konfiguracijama u realnoj primeni na klasteru i
- Testiranje performansi hibridnog rešenja na nekom većem setu podataka (npr. velikom skupu slika).

Na osnovu svega prethodno navedenog u ovom radu, može se zaključiti da predstavljena hibridna arhitektura ima veliki potencijal u realnoj primeni kada je u pitanju efikasna i brza obrada velikih količina podataka.

6. LITERATURA

- [1] Postojeće tehnike za segmentaciju slike, http://www.cqm.rs/2014/cd2/pdf/papers/focus_1/10.pdf
- [2] Digitalna obrada slike, http://dsp.etfbl.net/multimediji/2017/11_digitalna%20obrada%20slike.pdf
- [3] Image Processing in C, <http://homepages.inf.ed.ac.uk/rbf/BOOKS/PHILLIPS/cips2ed.pdf>
- [4] Convolution, http://machinelearningguru.com/computer_vision/basics/convolution/image_convolution_1.html
- [5] Diplomski rad „CUDA bazirana paralelizacija – 2D konvolucija“, Jelena Janković
- [6] Gaussian and Sobel algorithm, <http://www.rtrk.uns.ac.rs/sites/default/files/materijali/lab/Ve%C5%BEBa%209%20-%20Potiskivanje%20%C5%A1uma%20u%20slici.pdf>

[7] Sobel edge detection algorithm,

<https://pdfs.semanticscholar.org/6bca/fdf33445585966ee6fb3371dd1ce15241a62.pdf>

[8] 2D konvolucija i detekcija ivica u slici,

http://www.rtrk.uns.ac.rs/sites/default/files/materijali/lab/Ve%C5%BEBa%207%20-%20D%20konvolucija%20i%20%20detekcija%20ivica%20u%20slici_0.pdf

[9] BMP File Format Specification,

https://en.wikipedia.org/wiki/BMP_file_format

[10] C programming language, Brian W. Kernighan and Dennis M. Ritchie,

http://www.dipmat.univpm.it/~demeio/public/the_c_programming_language_2.pdf

[11] MPI, <file:///C:/Users/User/Downloads/MPI-Complete%20Reference.pdf>

[12] MPI Scatter and Gather,

<http://mpitutorial.com/tutorials/mpi-scatter-gather-and-allgather/>

[13] CUDA, David B. Kirk, Wen-mei W. Hwu,

Programming Massively Parallel Processors, Elsevier Inc, 2013.

Kratka biografija:



Jelena Janković rođena je 1994. godine u Novom Sadu. Završila je srednju ekonomsku školu "Svetozar Miletić" u Novom Sadu 2013. godine. Osnovne akademske studije na Fakultetu tehničkih nauka u Novom Sadu je završila 2017. godine. Master akademske studije na Fakultetu tehničkih nauka u Novom Sadu je upisala 2017. godine. Ispunila je sve obaveze i položila je sve ispite predviđene studijskim programom.

GPU IMPLEMENTACIJA KONVOLUCIONE MREŽE ZA DETEKCIJU OBJEKATA U KONTINUALNIM MEDIJIMA**GPU IMPLEMENTATION OF CONVOLUTIONAL NETWORKS FOR OBJECT DETECTION IN CONTINUOUS MEDIA**

Nenad Milić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Ovaj rad analizira efikasnost sistema za detekciju objekata implementiranog upotrebom Tensorflow biblioteke koja omogućuje izvršavanje na grafičkom procesoru.

Ključne reči: TensorFlow, GPU, CNN, Detekcija objekata

Abstract – This paper analyzes an object detection framework implemented using Tensorflow library that enables it to run on a GPU.

Keywords: TensorFlow, GPU, CNN, Object detection

1. UVOD

Ovaj rad se bavi problemom detekcije objekata sa pripodnih slika što predstavlja jedan od glavih problema u računarskoj viziji danas. Zadatak detekcije objekata sa slike je da se pronađu svi objekti na slici za klase koje se razmatraju i da se svakom objektu odredi konkretna klasa i pozicija u slici. Slike nad kojima se vrši detekcija se koriste u digitalnom obliku, u većini slučajeva predstavljene kao trodimenzionalni nizovi brojeva, dimenzija širina $\times$ visina $\times$ 3, gde broj 3 predstavlja tri kanala boje: crvena, zelena i plava.

S obzirom da je zadatak prepoznavanja vizuelnog koncepta (npr. pas) relativno jednostavan za ljude, potrebno je ustanoviti probleme koji se javljaju iz perspektive algoritama računarske vizije [1]:

- **Varijacija ugla gledanja.** Jedan objekat može imati beskonačno mnogo uglova iz kojih se može gledati na njega.
- **Varijacije u veličini.** Vizuelne klase često ispoljavaju varijacije u veličini kako u realnom svetu tako i na slici.
- **Deformacije.** Veliki broj objekata od interesa nisu čvrsta tela i mogu biti deformisana na ekstremne načine.
- **Zaklonjenost.** Objekat od interesa zna često biti delom zaklonjen nekim drugim objektom.
- **Osvetljenost.** Efekti osvetljenosti mogu drastično da utiču na nivou piksela.

- **Pozadinske smetnje.** Objekat od interesa može biti kamufliran tj. može se stopiti sa pozadinom što otežava identifikovanje.

- **Varijacije unutar klase.** Klase od interesa često mogu da obuhvataju objekte koji se u velikoj meri razlikuju. Postoji veliki broj različitih objekata neke klase, gde svaki objekat ima svoju predstavu.

Dobar sistem za detekciju objekata mora da bude dovoljno opšt da prevaziđe sve ove probleme ali i u isto vreme dovoljno osetljiv da može da prepozna razlike u klasama. Pored ovoga karakteristika dobrog sistema za detekciju objekata su i brzina i uspešnost sa kojom mogu da izvrše svoj zadatak. Ovo u velikoj meri zavisi od toga na koji način sistem odabira delove slike na koje će da usmeri svoju pažnju, i kako će iskoristi dostupne hardverske resurse.

2. IMPLEMENTACIJA I OPIS SISTEMA ZA DETEKCIJU OBJEKATA

Kao što je definisano u [2] detekcija objekata predstavlja predviđanje liste graničnih okvira za željene klase objekata na slici, ako uopšte postoji ijedan objekat na slici. Svaki granični okvir je opisan koordinatama i dimenzijama pravougaonika u kojem se smatra da se nalazi objekat. Najboljim graničnim okvirom se smatra najmanji granični okvir koji u celosti obuhvata objekat sa slike. Pored graničnog okvira potrebno je predvideti i realan broj u opsegu $[0, 1]$ koji predstavlja sigurnost sa kojom pronađeni objekat pripada određenoj klasi, gde broj bliži 1 predstavlja veću sigurnost pripadnosti pronađenog objekta nekoj od klasa. Sistem može da bude obučen za proizvoljan broj klasa.

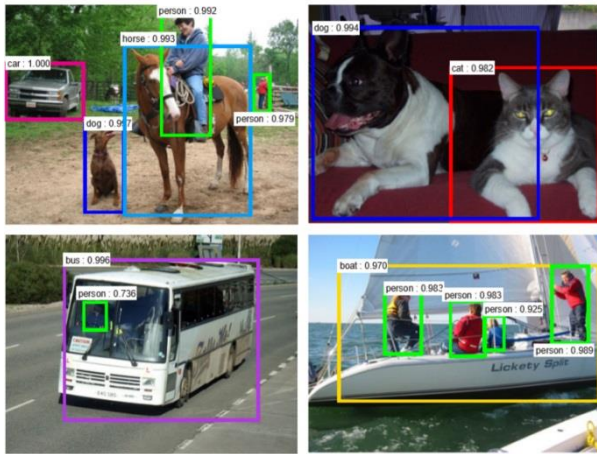
2.1. Faster RCNN

Faster RCNN [3] je rešenje iz familije konvolucionih neuronskih mreža zasnovanih na regionima. Ovi sistemi su se u početku oslanjali na algoritme koji predlažu regione slike, koji su klasno nezavisni, na osnovu kojih detektor određuje da li se u predloženom regionu nalazi objekat, i ako se nalazi, kojoj od mogućih klasa pripada. Primer detekcija ovoga sistema može se videti na slici 1.

Faster RCNN je nastao kao poboljšanje prethodnih sistema: *Fast RCNN* [4] i *RCNN* [5]. *RCNN* sistem koristi algoritme, kao što je selektivna pretraga, koji na osnovu originalne slike predlažu regione od interesa.

NAPOMENA:

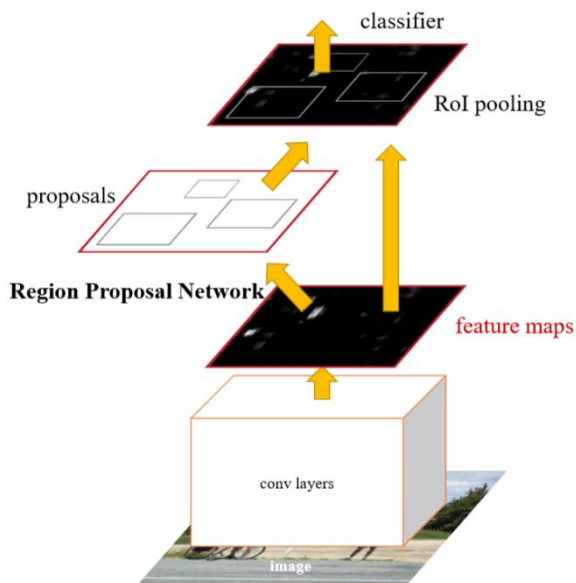
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivetić, red. prof.



Slika 1. Primer izlaza sistema za detekciju objekata [3]

Regioni od interesa se isecaju iz originalne slike i dalje šalju konvolucionoj neuronskoj mreži koja se završava sa potpuno povezanim slojevima koji klasifikuju ulaznu sliku.

Fast RCNN je ubrzana verzija *RCNN* rešenja. Zapažanjem da se dosta računanja ponavlja u konvolucionim slojevima, u *Fast RCNN* regioni se ne isecaju iz originalne slike nego se prvo cela slika propušta kroz konvolucione slojeve koji generišu mapu karakteristika, pa se tek odatle isecaju regioni, što znatno ubrzava rad sistema. Ovom optimizacijom, u *Fast RCNN* sistemu najsporiji deo postaje sam algoritam za predlaganje regiona koji se u *Faster RCNN* zamenjuje bržom mrežom za predlaganje regiona. Za mrežu za predlaganje regiona se može reći da predstavlja pažnju sistema. Skica *Faster RCNN* sistema se može videti na slici 2. gde se zapažaju dva glavna dela: mreža za predlaganje regiona i detektor koji dele konvolucione slojeve.



Slika 2. Izgled *Faster RCNN* sistema za detekciju objekata [3]

2.2. Ekstraktor karakteristika

Ekstraktor predstavlja deo sistema koji sadrži konvolucione slojeve. Njegov ulaz je slika na kojoj se

žele detektovati objekti, a izlaz mapa karakteristika ulazne slike.

Ulazna slika se prosleđuje kao tenzor veličine $visina \times širina \times 3$. S obzirom da ulazna slika može biti proizvoljne veličine, prvo se menja veličina sa održavanjem odnosa, a da je veća od dimenzija 500 piksela. Broj 3 predstavlja broj kanala slike, jedan za svaki iz RGB modela boja.

Mapa karakteristika predstavlja ulaznu sliku preslikanu u prostor karakteristika. Daljim slojevima sistema je ovakva prezentacija slike lakša za rad. Dimenzije ove mape zavise od konvolucione mreže koja se koristi u ovom sloju. U ovom radu kao ekstraktor osobina se koristi *ResNet* arhitektura sa 34 konvoluciona sloja koji na kraju daju mapu karakteristika koja je 4 puta manje širine i visine od početne slike, i dubine 512.

2.3. ResNet arhitektura

Duboke rezidualne mreže (engl. *Deep residual networks*, *ResNet*) [6, 7] predstavljaju familiju ekstremno dubokih arhitektura konvolucionih neuronskih mreža koje pokazuju dobru stopu konvergencije pri obučavanju, i veliku tačnost u rešavanju zadataka za koje se obučavaju.

Sastoje se od više rezidualnih jedinica koje se ređaju jedna na drugu, što ih čini modularnim. Svaka jedinica se može generalno izraziti u sledećoj formi:

$$y_l = h(x_l) + F(x_l, W_l), \quad (1)$$

$$x_{l+1} = f(y_l), \quad (2)$$

gde su x_l i x_{l+1} ulaz i izlaz jedinice l , F rezidualna funkcija, a f i h predstavljaju funkcije prečica.

Kod običnih konvolucionih mreža se javlja problem degradacije. Degradacija predstavlja povećanje greške pri obučavanju i kod testiranja, kako se povećava broj konvolucionih slojeva od kojih se mreža sastoji. *ResNet* arhitektura rešava degradaciju uvođenjem prečica. Pokazano je da prečice olakšavaju optimizaciju, jer u krajnjem slučaju da je izlaz jednog sloja već optimalan, lakše je proslediti ga nego modifikovati celu mrežu.

F predstavlja rezidualnu funkciju koja je potrebno da se nauči. F se sastoji od ReLU aktivacije i normalizacije, koje sprečavaju nastajanje premalih i prevelikih gradijenata, dva konvoluciona sloja, koji se sastoje od istog broja filtera veličine 3×3 . ReLU aktivacije i normalizacije se vrše pre konvolucije što ima efekat regularizacije. Regularizacija onemogućuje težinskim faktorima da postanu preveliki i time smanji mogućnost preobučavanja, što povećava sposobnost generalizacije.

U [6, 7] je pokazano da mreža najbolje uči kada su $h(x_l)$ i $f(y_l)$ funkcije identiteta: $f(y_l) = y_l$ i $h(x_l) = x_l$, jer u ovom slučaju signal direktno propagira od jedne do druge jedinice i u prolazu unapred i u prolazu unazad. Osim ovoga, ovako definisane f i h ne unose dodatne parametre.

Kao što je napomenuto, svaka rezidualna jedinica sadrži dva konvoluciona sloja koja imaju isti broj filtera koji je obično neki stepen broja 2. Nakon određenog broja jedinica broj filtera se povećava dva puta, što dovodi do toga da su ulaz i izlaz sloja različitog oblika. Ovo onemogućava jednostavno sabiranje ulaza i izlaza i rešava

se konvolucijom filterima veličine 1×1 kojih ima koliko i filtera u izlazu iz sloja. Sem ovoga veličina ulaza se smanjuje po širini i visini tako što se ova konvolucija vrši sa korakom od 2.

2.4. Mreža za predlaganje regiona

Mreža za predlog regiona predstavlja delom konvolucionu, delom potpuno povezanu neuronsku mrežu. Prvi deo mreže predstavlja konvolucionni sloj koji se u ovom radu sastoji od 512 filtera veličine 3×3 . Ovi filteri prelaze preko svakog piksela mape karakteristika i kao rezultat dobija se vektor od 512 vrednosti. Dalje ovaj vektor se šalje u dva nezavisna potpuno povezana sloja. Prvi potpuno povezani sloj na svom izlazu daje $k * 2$ vrednosti, dok drugi sloj daje $k * 4$ vrednosti.

Broj k predstavlja broj sidara koja su predlozi regiona na svakoj poziciji. Sidro je pravougaonik koji se opusuje određenom visinom i širinom, sa centrom u pikselu mape karakteristika na kojoj se konvolucionni sloj trenutno nalazi. Svako od k sidara ima različitu visinu i širinu, a odabiraju se tako da što bolje obuhvate objekte koji se pokušavaju detektovati.

Prvi potpuno povezani sloj, koji se naziva klasifikacionim, daje po 2 vrednosti za svako sidro, koje se propuštaju dalje kroz *softmax* sloj opisan sa funkcijom:

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}. \quad (3)$$

Uloga *softmax* funkcije je da na osnovu ulaznih vrednosti napravi verovatnosnu raspodelu, tako da se u ovom slučaju dobijaju 2 nove vrednosti čija suma je 1. Ova verovatnoća predstavlja sigurnost da li se na delu slike koji sidro pokriva nalazi objekat od interesa ili ne.

Drugi potpuno povezani sloj, koji se naziva regresionim, daje 4 vrednosti za svako sidro. Ove 4 vrednosti se interpretiraju kao pomeraj centra odgovarajućeg sidra po osama i promenu njegove visine i širine tako da se bolje obuhvati objekat, ako se nalazi neki u okviru sidra. Na kraju se ove vrednosti sabiraju sa odgovarajućim sidrima čime se dobijaju konačni regioni. Ovako se dobija po k predloga za svaki piksel mape karakteristika od kojih dalje treba izabrati one koji imaju najveću verovatnoću sadržanja objekta i koji međusobno imaju najmanje poklapanja da bi se izbegli predlozi koji sadrže isti objekat. Ovo se vrši upotrebom algoritma ne maksimalne supresije.

2.5. Ne maksimalna supresija

Ne maksimalna supresija (engl. *Non maximum suppression*, *NMS*) je pohlepni algoritam koji uzima podskup skupa graničnih okvira koji se najmanje poklapaju i koji imaju najveću verovatnoću sadržanja objekta. *NMS* pored graničnih okvira sa verovatnoćama zahteva još dva dodatna ulazna parametra. Prvi parametar predstavlja broj predloga koji je potrebno na kraju da se dobije, u [3] se pokazalo da je broj 300 najoptimalniji, jer odabirom broja manjeg od 300 mogu da se izostave neki od objekata na slici, dok broj veći samo dodaje nepotrebne granične okvire u kojima se ne nalazi objekat, i time usporava rad sistema. Drugi parametar predstavlja prag koeficijenta preklapanja 2 granična okvira za koji se

smatra da ta dva granična okvira obuhvataju isti objekat. Koeficijent preklapanja se određuje kao odnos površina preseka i unije 2 granična okvira. U [3] se pokazalo da prag od 0.7 daje dobre rezultate. Algoritam započinje tako što se pronalazi okvir sa najvećom verovatnoćom sadržanja objekta. Kada se pronađe računa se koeficijent preklapanja pronađenog sa svim ostalim graničnim okvirima i izbacuju se oni okviri koji imaju koeficijent veći od zadatog praga. Algoritam se dalje ponavlja iterativno: pronalazi se sledeći granični okvir od preostalih sa najvećom verovatnoćom i dalje se ponavlja iste procedura kao i u prvom koraku. Algoritam radi dok ne nađe broj zadatih graničnih okvira ili jednostavno ne ponestane graničnih okvira koji mogu da se izbace [8].

2.6. Detektor

Kada se dobiju regiona od mreže za predlog regiona i kada se propuste kroz *NMS*, regiona od interesa dalje idu u detektor. Detektor je poslednji deo sistema koji je dosta sličan mreži za predlaganje regiona.

Detektoru se na ulaz prosleđuju delovi već postojeće mape karakteristika. S obzirom da potpuno povezani slojevi zahtevaju ulaz fiksne veličine, u ovom slučaju $7 \times 7 \times 512$ vrednosti, isečenim regionima iz mape karakteristika je potrebno promeniti veličinu da odgovara ulazu detektora. Jedan od mogućih načina, koji je korišćen u ovom radu, je upotrebom metode za promenu veličine slike kao što je bilinearna interpolacija.

Detektor se sastoji od dva potpuno povezana sloja koji su jedan na drugom i svaki sadrži po 4096 neurona. Nakon toga se vezuju još dva nezavisna sloja gde prvi sloj, klasifikacioni, na izlazu daje $n + 1$ vrednost koje se propuštaju kroz *softmax* sloj da bi se dobila verovatnosna raspodela. Broj n predstavlja broj klasa za koje se sistem obučio, a 1 predstavlja klasu pozadine u slučaju da se detektoru prosledi region koji u sebi ne sadrži objekat. Drugi sloj, regresioni, na svom izlazu daje $n * 4$ vrednosti, slično kao i kod mreže za predlog regiona. n ukazuje da se dati region proračunavaju po 4 vrednosti za svaku klasu koje ukazuju, kao i kod mreže za predlog regiona, za koliko je potrebno pomeriti centar datog regiona po osama i promenu visine i širine regiona tako da se bolje obuhvati objekat. Krajnji rezultat se dobija sabiranjem odgovarajuće 4 vrednosti sa originalnim regionom, nakon što se utvrdi klasa objekta koji je u regionu.

2.7. Tensorflow

TensorFlow [9, 10] je softverska biblioteka otvorenog koda za numerička računanja upotrebom grafa toka podataka. Čvorovi u grafu predstavljaju matematičke operacije dok veze između njih predstavljaju multidimenzionalne nizove podataka (tenzore). Fleksibilna arhitektura omogućuje izvršavanje operacija na više centralnih ili grafičkih procesora upotrebom *API*-ja. Razvila ga je kompanija *Google* za potrebe istraživanja u polju mašinskog učenja i dubokih neuronskih mreža. *TensorFlow* je naslednik je *DistBelief* distribuiranog sistema za obučavanje neuronskih mreža. Osim mašinskog učenja ima primenu u svim poljima u kojima su potrebne efikasna numerička računanja.

TensorFlow biblioteka je primarno pravljen za Python programski jezik, ali postoje i *API*-ji za Java, C/C++, Haskell, Rust i Go programske jezike koji nisu ekstenzivni kao za Python već služe samo za *deployment* već obučenih modela. Jezgro *TensorFlow* biblioteke je implementirano u C++ programskom jeziku. Pored osnovnog *API*-ja niskog nivoa postoji i *API* srednjeg nivoa koji olakšava rad sa skupovima podataka i metrikama, i *API* visokog nivoa koji rad sa estimatorima koji predstavljaju kompletan model i enkapsuliraju obučavanje, evaluaciju, predikciju i eksportovanje modela.

Osim *TensorFlow* biblioteke postoje i druge koje se koriste u polju mašinskog učenja: Theano, Caffe, Keras, Matlab Neural Network Toolbox, Microsoft Cognitive Toolkit i mnoge druge. Zbog svog brzog napretka, fleksibilnosti i efikasnosti *TensorFlow* biblioteka postaje najpopularnija.

3. REZULTATI

Evaluirane su preformanse izvršavanja sistema opisanog u prethodnom poglavlju implementiranog pomoću verzije 1.8.0. *TensorFlow* biblioteke. Evaluacija je vršena na *Nvidia GeForce* 1060 grafičkom procesoru sa 6Gb VRAM-a. Prednost *Nvidia GeForce* grafičkih procesora je što sadrže veliki broj *CUDA* (engl. *Compute Unified Device Architecture*) jezgara koja predstavljaju paralelne procesore. *TensorFlow* koristi *CUDA* platformu za paralelno računanje koja predstavlja interfejs za pristup resursima koje nudi grafički procesor.

Prosečno vreme detekcije objekata za jednu sliku pomoću opisanog sistema na datoj platformi je 0.125 sekundi što predstavlja 8 frejmova u sekundi i može se smatrati kao pogodna brzina izvršavanja za obradu snimaka u realnom vremenu.

TensorFlow biblioteka dobro iskorištava ponuđene resurse. Najveća je iskorišćenost memorije u koju se smeštaju svi parametri modela i rezultati računanja koji potencijalno mogu da se upotrebe u narednim računanjima što ukazuje da bi ovom sistemu bilo pogodno da ima i više dostupne memorije. Iskorišćenost procesorske moći u velikoj meri varira i najviše se koristi u inicijalnom učitavanju modela.

4. ZALJUČAK

U ovom radu su analizirane preformanse *Faster RCNN* sistema za detekciju objekata sa *ResNet* ekstraktorom osobina implementiranog pomoću *TensorFlow* biblioteke uz oslonac na *Nvidia GeForce* 1060 grafički procesor relativno velike dostupnosti. Pokazalo se da preformanse sistema zadovoljavaju potrebe obrada kontinuiranih medija u realnom vremenu.

Ovo ukazuje da se sistem može dalje analizirati za potrebe obučavanja za zadatke u srodnim problemima računarske vizije.

Prednost opisanog sistema jeste to što se izmenom samo malog broja hiperparametara sistema lako može postići veća brzina u zamenu za manju tačnost prepoznavanja koja u nekim slučajevima može da bude prihvatljiva.

Dalji rad bi mogao biti u poređenju ovih rezultata sa drugim sistemima kao i poređenju ovog sistema sa drugim ekstraktorima osobina koji su sposobni da daju bolju tačnost u prepoznavanju.

4. LITERATURA

- [1] „Stanford University CS231n: Convolutional Neural Networks for Visual Recognition”, <http://cs231n.github.io/> (datum pristupa: 15.06.2018).
- [2] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman „The PASCAL Visual Object Classes Challenge: A Retrospective“, *Int J Comput Vis* (2015) 111:98–136
- [3] S. Ren, K. He, R. Girshick, J. Sun, „*Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*“, NIPS, 2016
- [4] R. Girshick, „*Fast R-CNN*“, ICCV, 2015
- [5] R. Girshick, J. Donahue, T. Darrell, J. Malik, „*Rich feature hierarchies for accurate object detection and semantic segmentation*“, CVPR, 2014
- [6] K. He, X. Zhang, S. Ren, J. Sun, „*Deep Residual Learning for Image Recognition*“, CVPR, 2016
- [7] K. He, X. Zhang, S. Ren, J. Sun, „*Identity Mappings in Deep Residual Networks*“, ECCV, 2016
- [8] R. Rothe, M. Guillaumin, L. Van Gool, „*Non-Maximum Suppression for Object Detection by Passing Messages between Windows*“, ACCV, 2014
- [9] „*TensorFlow*“, <https://www.tensorflow.org/> (datum pristupa 13.7.2018)
- [10] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, „*TensorFlow: A system for large-scale machine learning*“, OSDI, 2016

Kratka biografija:

Nenad Milić rođen je u Somboru 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Računarstvo i automatika odbranio je 2018.god.

PROGRAMSKI JEZIK OBJECTIVE-C

PROGRAMMING LANGUAGE OBJECTIVE-C

Mirko Mikać, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Ovaj rad opisuje programski jezik Objective-C, njegove osnovne funkcionalnosti kao i primere upotrebe.

Ključne reči: Objective-C, programiranje, jezik, funkcionalnost

Abstract – This paper describes Objective-C programming language, its most important features as well as examples of the application.

Keywords: Objective-C, programming, language, features

1. UVOD

Objective-C je programski jezik opšte namene koji se najviše koristi prilikom razvoja softverskih proizvoda koji je izvršavaju na Apple proizvodima kao što su OS X i iOS operativni sistemi. Zasniva se na programskom jeziku C i predstavlja njegovu striktnu nadogradnju pri čemu najveća razlika leži u objektno-orijentisanim mogućnostima koje Objective-C nudi kao i dinamičkoj tipizaciji.

2. UVOD

U ovom poglavlju biće opisani tipovi podataka koji su definisani u okviru Objective-C programskog jezika kao mehanizmi koji se koriste za podršku objektno-orijentisanih koncepata.

Tabela 1. Osnovni tipovi podataka

Tip	Primer
Celobrojni tip – <i>int</i>	12, -98, 100U, 0xFFE0
Decimalni tip – <i>float</i>	12.34f, 2.1e-5f, 0x1.5p10
Decimalni tip – <i>double</i>	12.34, 1p3
Karakterni tip – <i>char</i>	'a', 'A', '\n'

Osnovni tipovi podataka se sastoje od domena dozvoljenih vrednosti i operacija koje mogu da se izvrše. Na osnovu osnovnih tipova podataka grade se kompleksni tipovi podataka koji se koriste u objektno-orijentisanom razvoju rešenja. Uz osnovne tipove postoje i kvalifikatori koji se koriste kako bi se dodatno definisao domen vrednosti.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Aleksandar Kupusinac, vanr. prof.

Bitno je napomenuti da se kvalifikatori ne mogu koristiti uz sve osnovne tipove. Objective-C preuzima sve osnovne tipove iz programskog jezika C. Stoga ovi tipovi mogu da se vide i u mnogim drugim programskim jezicima kao što je C#, Java i C++.

2.1. Generički tip - *id*

Generički tip *id* se koristi ukoliko je neophodno da se čuva objekat bez informacije o njegovom tipu. Vrlo je bitan u Objective-C jeziku, jer se koristi u konceptima polimorfizma i dinamičkog povezivanja (eng. *dynamic binding*). Moguće je koristiti ovaj tip i prilikom deklaracije neke promenljive, kao i prilikom definisanja povratne vrednosti funkcije. Generički tip treba koristiti samo kad je to potrebno, a koristiti osnovne tipove u svim ostalim slučajevima zbog statičke tipizacije koja čini kod robusnijim.

2.2. Dinamičko povezivanje

Dinamičko povezivanje predstavlja koncept usko povezan sa generičkim tipom *id*. Upotrebom ovog mehanizma Objective-C omogućava da se u okviru iste promenljive čuvaju različiti tipovi objekata prilikom izvršavanja programa.

```
56 id dataValue;
57 Point *p1 = [[Point alloc] init];
58 Complex *c1 = [[Complex alloc] init];
59 [p1 setTo: 2 over: 5];
60 [c1 setReal: 10.0 andImg: 2.5];
61 // first dataValue gets a point
62 dataValue = p1;
63 [dataValue print];
64 // now dataValue gets a complex number
65 dataValue = c1;
66 [dataValue print];
```

Slika 1. Upotreba dinamičke tipizacije

2.3. Inspekcija objekata

Prilikom rada sa promenljivama koje su generičkog tipa, moguće je da te promenljive sadrže različite tipove objekata tokom izvršavanja. Stoga je potrebno da postoje mehanizmi koji omogućavaju da se dobiju informacije o klasama koje predstavljaju stvarni tip podataka tih promenljivih. Objective-C pruža mogućnost da se dobiju sledeće informacije za neki objekat:

- Da li objekat pripada nekoj konkretnoj klasi?
- Da li objekat podržava standardnu metodu za ispis?
- Da li objekat pripada nekoj konkretnoj klasi ili nekom od potamaka?

Method	Question or Action
<code>-(BOOL) isKindOfClass: class-object</code>	Is the object a member of <i>class-object</i> or a descendant?
<code>-(BOOL) isKindOfClass: class-object</code>	Is the object a member of <i>class-object</i> ?
<code>-(BOOL) respondsToSelector: selector</code>	Can the object respond to the method specified by <i>selector</i> ?
<code>*(BOOL) instancesRespondToSelector: selector</code>	Can instances of the specified class respond to <i>selector</i> ?
<code>*(BOOL) isKindOfClass: class-object</code>	Is the object a subclass of the specified class?
<code>-(id) performSelector: selector</code>	Apply the method specified by <i>selector</i> .
<code>-(id) performSelector: selector withObject: object</code>	Apply the method specified by <i>selector</i> , passing the argument <i>object</i> .
<code>-(id) performSelector: selector withObject: object1 withObject: object2</code>	Apply the method specified by <i>selector</i> with the arguments <i>object1</i> and <i>object2</i> .

Slika 2. Metode za dobijanje informacija o klasi [1]

2.4. Definisanje klase

Kako bi se definisala nova klasa potrebno je to uraditi u okviru odgovarajućih sekcija. Sekcije predstavljaju logički podeljene delove koda i sledeće sekcije su potrebne za definisanje klase:

- Interfejs sekcija – `@interface`
- Implementaciona sekcija – `@implementation`

Sekcija `@interface` predstavlja deo u okviru kojeg se opisuje klasa. Navodi se roditeljska klasa čije osobine klasa nasleđuje, zatim se definišu odgovarajući atributi klase koji predstavljaju stanje i deklariraju metode čijim se pozivom menja stanje konkretne instance klase. Takođe se mogu definisati i statički atributi i metode koji se dele u okviru klase.

```

41 #import <Foundation/Foundation.h>
42 //---- @interface section ----
43 @interface Complex: NSObject
44 {
45     int real;
46     int img;
47 }
48 -(void) print;
49 -(void) setReal: (int) r;
50 -(void) setImg: (int) i;
51 @end

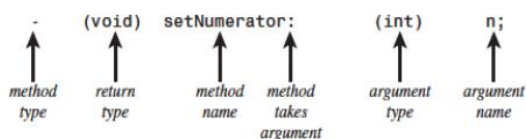
```

Slika 3. Interface sekcija

U implementacionoj sekciji `@implementation` se nalazi konkretna implementacija metoda, odnosno telo metode koje treba da se izvrši prilikom poziva.

2.5. Metode instance i metode klase

Objective-C ne poseduje ključnu reč za razlikovanje između metoda koje se primenjuju na nivou instance i metoda koje se primenju na nivou klase. Za ovo se koriste znaci plus (+) i minus (-) prilikom deklaracije metode. Ukoliko se pre deklaracije metode navede minus (-) tada se govori kompajleru da je u pitanju metoda na nivou instance, a ukoliko se navede (+) tada je u pitanju metoda na nivou klase i ne zahteva instanciranje objekta.



Slika 4. Delovi deklaracije metode [1]

2.6. Programska sekcija

Programska sekcija predstavlja kod u okviru kojeg se koriste objekti čije su klase opisane uz pomoć prethodnih sekcija ili objekti koji pripadaju klasama različitih biblioteka, kako bi pozivom njihovih metoda i međusobnom komunikacijom rešili određeni problem.

2.7. Odvajanje interfejsa od implementacije

Prvi korak u radu sa klasama je pravilna implementacija. Najbolja praksa jeste da u različite fajlove razdvoji `@interface` sekcija od `@implementation` sekcije. Interfejsi klasa se čuvaju u fajlovima sa ekstenzijom `.h` u praksi, mada nije nužno, dok se implementacije klasa čuvaju u fajlovima sa ekstenzijom `.m`. Kasnije povezivanje implementacije i interfejsa se radi upotrebom `#import` naredbe koja govori kompajleru da je za potrebe kompajliranja ovog fajla potreban i fajl naveden uz ovu ključnu reč. Odvajanje fajlova povlači i određene posledice u načinu kompajliranja, u smislu da kompajler nezavisno kompajlira svaki fajl.

```

53 #import "Complex.h"
54 @implementation Complex
55 -(void) print { NSLog(@"%i + %ii", real, img); }
56 -(void) setReal: (int) r { numerator = r; }
57 -(void) setImg: (int) i { denominator = i; }
58 @end

```

Slika 5. Implementaciona sekcija i `#import` upotreba

2.6. Nasleđivanje

Kao što je rečeno, sintaksa za predstavljanje nasleđivanja podrazumeva navođenje roditeljske klase u `@interface` sekciji nakon definisanja naziva klase iza znaka dvotače.

```

@interface Complex: NSObject
...
@end

```

Slika 6. Sintaksa nasleđivanja

Koncept nasleđivanja je hijerarhijske organizacije, što znači da postoji inicijalna roditeljska klasa koju svi direktno ili indirektno nasleđuju. U programskom jeziku Objective-C inicijalna klasa koja se naziva i korenska klasa je `NSObject`. Klasa koja nasleđuje neku klasu se naziva dečija ili subklasa, dok se roditeljska klasa naziva i superklasa.

2.6. @class direktiva

Direktiva `@class` se koristi kada je potrebno da objekat poseduje informacije koje su predstavljene u vidu drugih objekata, i u ovoj situaciji se koristi drugačiji pristup u odnosu na osnovne tipove i koriste se pokazivači. Drugi način za uvođenje objekta jeste `#import` koja sa sobom nosi drugačije mogućnosti.


```

61 #import <Foundation/Foundation.h>
62 @class Point;
63 @interface Triangle : NSObject
64 {
65     Point *a;
66     Point *b;
67     Point *c;
68 }
69 -(void) printVar;
70 @end
71 @implementation Triangle
72 -(void) printVar { ... }
73 @end

```

Slika 6. Direktiva @class

S obzirom na to da kompajler mora da zna za klasu *Point* prilikom procesiranja klase *Triangle* jedan od načina da se ovo uradi je upotrebom `#import` direktive. Upotrebom `#import` direktive dobijaju se na raspolaganju kako klasa tako i metode za objekat te klase. Upotrebom direktive `@class` kompajleru se govori da je u pitanju klasa *Point* i ništa više. Iz ovog razloga metode neće biti dostupne upotrebom ove direktive s obzirom na to da se nalaze u okviru `@implementation` sekcije.

2.7. Upravljanje memorijom

U situaciji kada objekat u sebi sadrži druge objekte kao atribute, neophodno je da se na pravi način vodi računa o memoriji s obzirom na to da je programer odgovoran za oslobađanje memorije unutrašnjih objekata kada završi sa radom. U primeru na slici 6. objekat klase *Triangle* koristio je objekat klase *Point* koji predstavlja zasebnu klasu koju je potrebno instancirati, odnosno zauzeti određenu memoriju, a na kraju rada sa objektom klase *Triangle*, istu tu memoriju treba osloboditi. Isto tako, može se primetiti da nije u pitanju jedan unutrašnji objekat, već tri objekta klase *Triangle*. Prilikom svakog oslobađanja memorije za objekat klase *Triangle* bilo bi potrebno proverati njegove *Point* objekte i oslobađati zauzetu memoriju. Kao elegantno rešenje za ovaj problem se redefiniše metoda *dealloc()* u okviru koje se proverava zauzeta memorija za unutrašnje objekte i potom oslobađa ako je potrebno.

```

76 -(void) dealloc
77 {
78     if (a)
79         [a release];
80     if (b)
81         [b release];
82     if (c)
83         [c release];
84     [super dealloc];
85 }

```

Slika 7. Metoda dealloc()

Redefinisanje metode *dealloc()* nosi sa sobom posledicu da je neophodno osloboditi memoriju svih preuzetih atributa koji su nasleđeni i instancirani. Stoga je potrebno pozvati roditeljsku metodu *dealloc()* što predstavlja poslednju liniju koda gde se metoda primenjuje nad roditeljem upotrebom ključne reči *super*.

2.8. Kategorije

Kategorije predstavljaju praktično rešenje kada je potrebno da se na jednostavan način modularizuje definicija klase u grupi ili kategorije povezanih metoda. Na ovaj način pruža i mogućnost da se nasledi postojeća klasa bez potrebe za pristupom originalnom izvornom kodu i bez potrebe da se kreira dečija klasa. Metode za standardne matematičke operacije se podrazumevaju nad kompleksnim brojevima. U ovom slučaju te metode se mogu izmestiti u kategoriju pod nazivom *Math*.

```

88 #import "Complex.h"
89 @interface Complex (Math)
90 -(Complex *) add: (Complex *) c;
91 -(Complex *) subtract: (Complex *) c;
92 -(Complex *) multiply: (Complex *) c;
93 -(Complex *) divide: (Complex *) c;
94 @end

```

Slika 8. Kategorija Math

Iako je na slici 8 upotrebljena ključna reč za interfejs sekciju, ne predstavlja zasebnu definiciju interfejsa, već ekstenziju na postojeću, tako da je kompajler upoznat sa samom *Complex* klasom. Kategorije mogu da vrše redefinisane postojećih metoda iz originalne definicije i ne predstavlja dobru praksu i ne treba da se koristi. Dodavanje novih atributa nije moguće i može samo da se pristupa postojećim, pri čemu bi za takve potrebe bio bolji mehanizam nasleđivanja.

2.8. Protokoli i delegiranje

Protokol predstavlja listu metoda koje se dele između više klasa. Kao takvi, protokoli nisu zamišljeni da poseduju konkretnu implementaciju, već je cilj da se konkretna implementacija dodaje prilikom kreiranja konkretne klase koja bi koristila protokol. Prilikom definisanja protokola, moguće je da se navede koje od metoda predstavljaju opcione, a koje obavezne metode za implementaciju. Za definisanje protokola koristi se direktiva `@protocol` nakon koje sledi sam naziv protokola koji je opcion, a posle imenovanja protokola navode se same metode. Sve metode koje su navedene do `@end` direktive predstavljaju metode koje pripadaju protokolu.

```

96 @protocol Math
97 - (id) add: (id) first to: (id) second;
98 - (id) subtract: (id) first with: (id) second;
99 - (id) multiply: (id) first with: (id) second;
100 - (id) divide: (id) first with: (id) second;
101 @end

```

Slika 9. Protokol Math

Svaka klasa koja koristi *Math* protokol prikazan na slici 9 je u obavezi da implementira metode koje su definisane u okviru istog. Da bi se uveo protokol za potrebnu klasu potrebno je da se navede u okviru interface sekcije. U ovom slučaju u pitanju su binarne matematičke operacije sabiranja, oduzimanja, množenja i deljenja.

```

103 @interface Complex : NSObject <Math, SomeProtocol2>

```

Slika 10. Navođenje protokola u interfejs sekciji

Moguće je definisati i metode koje su opcione za implementaciju upotrebom `@optional` direktive ispred metoda koje nisu neophodne za implementaciju. Prilikom

rada sa protokolima nekada je potrebno proveriti da li određeni objekat pripada klasi koja koristi određeni protokol. Za ove potrebe se koristi metoda `conformsToProtocol()`.

```
105 id currentObject;
106 // Do some programming stuff, assign something to currentObject
107 if ([currentObject conformsToProtocol:@protocol (Math)] YES)
108 {
109     // Invoke some of the functions that were defined in Math protocol
110 }
```

Slika 11. Provera da li klasa objekta koristi protokol *Math*

2.9. Delegati i neformalni protokoli

Delegati predstavljaju bitan koncept koji se koristi u iOS razvoju. U praksi protokoli predstavljaju interfejs za komunikaciju gde klasa koja definiše protokol može da se posmatra kao neko ko delegira rad koji je definisan metodama protokola klasi koja implementira iste. Neformalni protokoli predstavljaju kategorije koje sadrže listu metoda, ali ih ne implementiraju. Neformalni protokoli se nazivaju i apstraktni protokoli i definišu se kao kategorije. U ovom slučaju, implementacija metoda kategorije *Math* nedostaje i ostavlja se samoj dečijoj klasi koja odluči da ih implementira, pri čemu mora prvo ponovo da ih deklarise, a potom implementira neke od njih. Neformalni protokol je zapravo grupa metoda pod istim nazivom i prilikom njihove upotrebe Objective-C kompajler nije u stanju da izvrši bilo kakvu proveru validnosti upotrebe neformalnog protokola.

```
113 @interface Complex (Math)
114 -(Complex *) add: (Complex *) c;
115 -(Complex *) subtract: (Complex *) c;
116 -(Complex *) multiply: (Complex *) c;
117 -(Complex *) divide: (Complex *) c;
118 @end
```

Slika 12. Upotreba neformalnih protokola za grupisanje matematičkih operacija

2.10. Pretprocesorske direktive

Pretprocesor pruža alate koji omoguću razvoj programa koji su skalabilniji, lakši za održavanje i portabilni. Deo je Objective-C kompajlera u okviru kog se prepoznati specijalni iskazi koji mogu da se nalaze bilo gde u okviru programa. Pretprocesor ima zadatak da obradi ove iskaze pre nego što sam program krene sa izvršavanjem. Direktive ovog tipa se prepoznaju po znaku `#` koji se nalazi uz ime direktive. Najbitnije direktive koje se koriste u Objective-C jeziku su sledeće:

- `#define`
- `#import`
- `#include`

Direktiva `#define` predstavlja direktivu koja se koristi za dodelu simboličkih imena konstantama programa. Mesta gde se ova direktiva koristi jeste najčešće na početku programa. Direktiva `#import` se koristi kada je potrebno da se uvede neki skup makroa odnosno defincija sa jednim ili više parametara.

```
123 // part of constants.h
124 #define PI 3.14
125 #define E 2.718
126 #define G 9.81
127
128 // other file
129 #import "constants.h"
```

Slika 13. Upotreba direktiva `#define` i `#import`

3. ZAKLJUČAK

U radu su opisani osnovni tipovi podataka u programskom jeziku Objective-C kao i mehanizmi za podršku objektno-orijentisanim konceptima. Na osnovu mogućnosti jezika mogu se izvesti sledeći zaključci:

- Programski jezik Objective-C podržava najbitnije objektno-orijentisane koncepte i može da se posmatra kao validan izbor prilikom razvoja rešenja u objektno-orijentisanom obliku.
- Upotrebom direktiva koje su opisane u radu omogućava se modularizacija softverskih rešenja, skalabilnost i portabilnost, kao i mogućnost ponovne upotrebe delova softvera.
- S obzirom na to da predstavlja striktnu nadogradnju C programskog jezika može se iskoristiti u situacijama kada je i proceduralno programiranje u pitanju mada za to nije namenjen.
- Upravljanje memorijom predstavlja bitan deo prilikom razvoja i potrebno je voditi računa kako prilikom zauzimanja memorije, tako i prilikom oslobađanja memorije. Objective-C nema mehanizme koji postoje u okviru drugih programskih jezika za upravljanje memorijom gde je jedan od primera *Java* programski jezik.

Objective-C se koristi prilikom razvoja *Apple* proizvoda tako da bi glavni fokus trebalo da bude na konkretnim primerima razvoja softverskih rešenja za ove svrhe. Ono što su mane ovog programskog jezika, pored potrebe za ručnim upravljanjem memorijom, jeste dosta neintuitivna sintaksa kao i uzak domen upotrebe koji vrlo često navode programere da se odluče za upotrebu drugih programskih jezika.

4. LITERATURA

- [1] S.G. Kochan, "*Programming in Objective-C, 3rd edition*", United States of America, 2011.

Kratka biografija:



Mirko Mikać rođen je u Novom Sadu 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Primenjene računarske nauke i informatika odbranio je 2018.god.

kontakt: mmikac27@gmail.com

OBJEKTNO-ORIJENTISANA PARADIGMA U PROGRAMSKOM JEZIKU PYTHON
OBJECT-ORIENTED PARADIGM IN PYTHON PROGRAMMING LANGUAGEStefan Varajić, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – Ovaj rad opisuje objektno-orijentisanu paradigmu u programskom jeziku Python. Dat je i kratak uvod u sam programski jezik.

Ključne reči: Python, programski jezici, objektno-orijentisana paradigma

Abstract – This paper describes object-oriented paradigm in Python programming language. A brief introduction is given to the language.

Keywords: Python, programming languages, object-oriented paradigm

1. UVOD

Python je programski jezik visokog nivoa u kome može biti rešavan širok spektar problema. Naglašava čitljivost koda, koristeći *whitespace* karaktere, odnosno identaciju. Podržava više programskih paradigmi uključujući: imperativnu, proceduralnu, objektno-orijentisanu i funkcionalnu. Sadrži sve elemente karakteristične za jedan programski jezik: promenljive, tipove podataka, operatore, upravljačke strukture, kolekcije, funkcije i objekte. Klase predstavljaju glavni objektno-orijentisani alat Python programskog jezika. Objektno-orijentisano programiranje (OOP) je jednostavnije za razumevanje od većine drugih OOP jezika. Većina priče se svodi na izraz: *object.attribute*.

2. OSNOVNI ELEMENTI

U ovom poglavlju biće opisani najvažniji elementi programskog jezika Python verzije 3.

2.1. Promenljive

Promenljive se ne deklarišu, već se samo dodeli vrednost promenljivoj. Tip se ne navodi eksplicitno za promenljivu, biće zaključen na osnovu objekta koji joj je dodeljen. Promenljive nisu statički vezane za neki tip.

U jednom trenutku promenljiva može biti jednog tipa, zatim ako joj se dodeli objekat drugog tipa ona menja tip. Primer dodele vrednosti promenljivoj dat je na slici 1.

```
a = 53
b, c = 53 + 2.8, 80
a = "Hello" + str(a)
```

Slika 1. Dodela vrednosti promenljivoj

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Aleksandar Kupusinać, vanr. prof.

2.2. Osnovni tipovi podataka

Osnovni tipovi podataka su: brojevi i stringovi. Brojevi se dele na sledeće konkretne tipove: celobrojni tip (*integer*), decimalni tip (*float*), kompleksne brojeve (*complex*), decimalne sa fiksiranom preciznošću (*Decimal*) i racionalne sa brojiocem i imeniocem (*Fraction*). U Python-u 3 je izbačen *long* tip, a *integer* je preuzeo tu ulogu obezbeđivanja preciznosti za velike brojeve. Softverski inženjer nema kontrolu kog je promenljiva tipa, već to Python sam menja po potrebi. Na primer: rezultat svakog deljenja će biti *float* tipa, ukoliko se eksplicitno želi da bude *integer* tipa tu je operator dvostruki slash (*//*) koji izvršava celobrojno deljenje. *Float* tip nije precizan i ne treba ga koristiti za neke ozbiljnije matematičke ili naučne proračune. Umesto njega koristiti *Decimal* tip. String je niz karaktera i može se navesti između jednostrukih ili dvostrukih navodnika. U Python programskom jeziku stringovi su nepromenljivi (eng. *immutable*).

2.2. Kolekcije

Kolekcije su objekti koji mogu da sadrže više drugih objekata. Python podržava sledeće kolekcije: liste, torke, setove i rečnike.

2.2.1. Liste

Liste predstavljaju uređenu kolekciju proizvoljnih objekata. Kojim redom se objekti dodaju u listu, tim redom će biti i smešteni. Definišu se uglastim zagradama (`[ ]`). U jednoj listi mogu da se nađu objekti različitih tipova. Pristupanje elementu liste se vrši preko indeksa. Mogu biti ugnježdene, odnosno mogu sadržati proizvoljan broj drugih listi u sebi. Promenljive (eng. *mutable*) su, što znači da se elementi mogu proizvoljno dodavati, uklanjati i modifikovati. Takođe, liste su dinamičke, kad se dodaju novi elementi same se proširuju, a kad se uklanjaju same se skupljaju. Matrice se mogu kreirati ugnježdenim listama, tako što se napiše lista koja u sebi ima liste. Primer definisanje liste, iteriranja kroz listu i ispisa elemenata dat je na slici 2.

```
cities = ['London', 'Reykjavik', 'Rome', 'Paris', 'Berlin']
for city in cities:
    print(city)
```

Slika 2. Definisanje liste, iteriranje i ispis elemenata

2.2.2. Torke

Torke su vrlo slične listama, ali imaju dve bitne razlike: definišu se regularnim zagradama (`( )`) i nepromenljive su, dakle objekat torke se ne može menjati. Torka sa jednim elementom se definiše na sledeći način: `t = (1,)`.

Zapeta je neophodna, inače bi *Python* gledao promenljivu kao *integer*. Kad koristiti torku umesto liste? Kod torki je veća brzina izvršavanja programa, kad je potrebno da vrednosti u kolekciji budu konstantni. U rečnicima vrednosti moraju da budu nepromenljivog tipa, tada će se koristiti torke, a ne liste.

2.2.3. Setovi

Setovi su neuređeni tip kolekcije, što znači da se elementima ne može pristupiti pomoću indeksa. Sadrže samo unikatne, odnosno jedinstvene objekte. Imaju podršku za matematičke operacije poput: unije, preseka, razlike i simetrične razlike. Promenljivi su tip objekta. Definišu se putem vitičastih (`{ }`) zagrada ili sa ugrađenom funkcijom `set()`. Prazan set je neophodno kreirati sa funkcijom, a ne zagradama, jer bi *Python* gledao kao kreiranje praznog rečnika. Često se koriste kad je potrebno filtrirati elemente, da ostanu samo oni jedinstveni.

2.2.4. Rečnici

Pored lista, rečnici spadaju u najkorišćenije kolekcije u *Python* programskom jeziku. Imaju neke sličnosti poput: promenljivi su, dinamički su i mogu se ugnježdavati. Bitne razlike su u tome što rečnici predstavljaju neuređenu kolekciju objekata i što se elementima rečnika ne pristupa putem indeksa, već putem ključa.

Definišu se tako što se unutar vitičastih (`{ }`) zagrada navode parovi odvojeni dvotačkom (`:`), a svaki par je odvojen zarezom. Svaki par sadrži ključ-vrednost. Mogu se i definisati putem ugrađene `dict()` funkcije. Vrednost iz rečnika se dobija tako što se napiše prvo naziv rečnika, a potom odgovarajući ključ u uglastim zagradama (`[ ]`).

Na slici 3 dat je primer definisanja rečnika koji za ključ ima državu, a za vrednost glavni grad i definisanja liste gradova. Zatim se iterira po ključevima i vrši provera da li se grad iz rečnika nalazi u listi.

```
countries = dict([
    ('England', 'London'),
    ('France', 'Paris'),
    ('Italy', 'Rome'),
    ('Germany', 'Berlin')
])

cities_to_visit = ['Rome', 'Paris']
for name in countries.keys():
    if countries[name] in cities_to_visit:
        print(name)
```

Slika 3. Definisanje rečnika, iteriranje po ključevima i ispis odgovarajućih elemenata

2.3. Funkcije

U okviru *Python* programskog jezika, funkcije predstavljaju najjednostavniju programsku strukturu za što veću ponovnu iskoristivost. Funkcije omogućuju da se složen sistem rastavi na jednostavnije činioce, čime se implementacije olakšava i omogućava se da veći broj može da učestvuje u razvoju sistema. Postoje dva načina za definisanje funkcije, jedan je pomoću ključne reči `def`, a drugi pomoću ključne reči `lambda`. Pored ovoga postoje dva načina da se upravlja opsegom funkcije pri čemu se koriste ključne reči: *global* i *nonlocal*, kao i dva načina da se vrati rezultat funkcije upotrebom ključnih reči `return` i `yield`. Vrlo je često prosleđivanje argumenata na osnovi ključnih reči (eng. *keyword arguments*).

Ovo podrazumeva da se argumenti prosleđuju kao par koji predstavlja naziv parametra kojem se dodeljuje konkretna vrednost. Takođe, postoji mehanizam za definisanje podrazumevanih vrednosti u slučaju da argument nije prosleđen. Primer takve funkcije je prikazan na slici 4.

```
def print_msg(msg, message_type='info'):
    print('Message: ' + msg)
    print('Message type: ' + message_type)

print_msg(msg='You are being informed')
```

Slika 4. Definisanje funkcije, poziv, parametri sa podrazumevanom vrednošću

3. KLASA I OBJEKTI

Objektno-orientisano programiranje je jedno od najefektivnijeg pristupa pisanju softvera. Klase su način za definisanje novih stvari, reflektujući realne objekte u programski domen. Recimo da se želi napraviti automobil, potrebno je modelovati njegovu strukturu i odnose prema drugim objektima. Dva aspekta OOP-a koja su korisna u ovom primeru su: nasleđivanje i kompozicija. Svaki automobil je vozilo, tako da poseduje osobine vozila.

Funkcionalnosti koje važe za vozilo, važe i za automobil. To čini kod ponovo iskoristivim, ukoliko je neophodno napisati neku drugu klasu koja nasleđuje vozilo, a funkcionalnosti su iste. Ako je nešto specifično, moguće je redefinisati ponašanje.

Takođe, automobil predstavlja kolekciju komponenata koje funkcionišu zajedno. Da bi automobil ispravno funkcionisao, mora da ima komponente poput: motora, guma, karoserije.

Ovo predstavlja kompoziciju. pakovanje logike i podataka. Klase definišu imenski prostor, poput modula. Ali klase imaju i neke bitne razlike, i korisne su kad se grade novi objekti:

- Višestruko instanciranje – klase se mogu posmatrati kao fabrike koje prave jedan ili više objekata.
- Prilagođavanje putem nasleđivanja – klasa potomak može da redefiniše ponašanje klase roditelja
- Preklapanje operatora – ova osobina predstavlja prilagođavanje ponašanja operatora konkretnoj klasi

3.1. Kreiranje i korišćenje klasa

Klase u programskom jeziku *Python* počinju sa ključnom rečju *class*. Zatim ide naziv klase, koji po konvenciji počinje velikim slovom.

Dalje se pišu zagrade koje su prazne, ukoliko klasa ne nasleđuje nijednu drugu klasu. Na slici 5 je prikazana klasa koja modeluje osobu koja od osobina ima: ime i prezime i godine, i obavlja neke radnje poput: spavanja i rada na poslu.

```
class Person():
    """A simple model of Person."""
    def __init__(self, fullname, age):
        """Initialize fullname and age attributes."""
        self.fullname = fullname
        self.age = age

    def sleep(self):
        """Simulate person sleeping."""
        print(self.fullname.title() + " is now sleeping.")

    def work(self):
        """Simulate person working."""
        print(self.fullname.title() + " is now working!")
```

Slika 5. Klasa *Person*

Funkcija koja pripada klasi naziva se metoda. Metode su slične kao i funkcije, s tim što se drugačije definišu i pozivaju. Konstruktor, odnosno `__init__()` metoda je specijalna metoda koju *Python* pokreće kada se instancira objekat neke klase. Ima dve donje crte na početku i na kraju kako bi se izbegao konflikt između predefinisanih metoda i onih koje piše programer. Metoda `__init__()` ima tri parametra: *self*, *fullname* i *age*. *Self* parametar se ne prosleđuje manuлно, *Python* ga sam prosledi. On predstavlja referencu na konkretnu instancu objekta, odnosno daje pristup atributima i metodama u klasi. Takođe, postoje i dve metode koje imaju samo *self* parametar i vrše običan ispis na konzolu.

3.2. Kreiranje instance klase

Pravljenje instance klase, koja predstavlja specifičan objekat klase *Person* data je u primeru na slici 6. Nema ograničenja koliko puta se jedna klasa može instancirati.

```
person = Person('Pera Peric', 20)
print("My name is " + person.fullname.title() + ".")
print("I'm " + str(person.age) + " years old.")
person.sleep()
person.work()
```

Slika 6. Instanciranje klase *Person*

Za pristup atributima i metodama se koristi tačka (`.`). Atribut se može promeniti direktno, kako je prikazano u primeru. Zatim, može se modifikovati preko metode, odnosno može postojati metoda koja menja vrednost nekog atributa (eng. *setter*).

3.3. Promenljive klase i promenljive objekta

U prethodnom primeru su bile prikazane dve promenljive: *fullname* i *age*. One spadaju u promenljive objekta. Postoje dve vrste promenljivih: promenljive klase i promenljive objekta. Promenljive objekta su u vlasništvu svakog zasebnog objekta, odnosno instance klase. Svaki objekat ima svoj primerak promenljive, ne dele se i nemaju veze sa drugim instancama. Promenljive klase su deljive, odnosno mogu biti dostupne svim instancama te klase. Postoji samo jedan primerak promenljive klase i kad neki objekat napravi promenu na promenljivoj klase, ta promena će biti vidljiva svim objektima te klase. Na primeru *Person* klase, promenljiva klase može biti *population*. Ta promenljiva bi brojala koliko trenutno ima instanciranih objekata klase *Person*. Dakle, svaki put kad se instancira klasa, ta promenljiva će se uvećati. Svaki put kad se objekat uništi, ta promenljiva bi trebalo da se smanji. Kad objekat prestane da postoji, koristi se `__del__()` metoda, koja predstavlja destruktor koji se

automatski poziva. U *Python*-u postoji *garbage collector* koji vodi računa o memoriji. Nije neophodno kao u nekim drugim OOP jezicima da programer vodi računa o oslobađanju memorije i piše destruktore. Stoga se `__del__()` metoda retko koristi u *Python*-u. Primer promenljive klase i `__del__()` metode dat je na slici 7.

```
class Person():
    """A simple model of Person."""

    population = 0

    def __init__(self, fullname):
        """Initialize fullname and age attributes."""
        self.fullname = fullname
        self.age = 0
        Person.population += 1

    def __del__(self):
        Person.population -= 1
```

Slika 7. Promenljive klase i destruktor

Za razliku od većine programskih jezika, svi članovi klase su javni i sve metode su virtuelne, što znači da se bilo kom polju može pristupiti iz bilo kog dela programskog koda.

3.4. Nasleđivanje

Pisanje klasa se ne počinje uvek od početka. Ukoliko klasa koja se piše predstavlja specijalizovanu verziju neke druge klase koja je već napisana, tada se može koristiti nasleđivanje.

3.4.1. Init metoda deteta

Kad se kreira instanca deteta, prvo što se uradi je dodela vrednosti svim atributima u roditeljskoj klasi. Kad se kreira potomak, roditeljska klasa mora biti deo trenutnog fajla i mora se nalaziti pre potomka u fajlu. Ime roditeljske klase se nalazi unutar zagrade u definiciji klase potomka. Funkcija *super()* omogućava konekciju između roditelja i deteta. Na osnovu nje *Python* poziva `__init__()` metodu roditeljske klase. Primer nasleđivanja dat je na slici 8.

```
class Engineer(Person):

    def __init__(self, fullname):
        """Initialize attributes of the parent class."""
        super().__init__(fullname)
```

Slika 8. Primer nasleđivanja

3.4.2. Definisane atributa i metoda za klasu potomka

Jednom kad je kreirana klasa potomak, mogu se dodati atributi i metode kako bi se napravila razlika između roditelja i potomka. Svaka metoda iz roditeljske klase može biti pregažena (eng. *override*). To se radi ukoliko ne odgovara implementacija iz roditeljske klase potomku. Potrebno je definisati metodu u potomku sa istim imenom kao i metoda u roditeljskoj klasi. Pri pozivu te metode *Python* neće gledati roditelja, već implementaciju u potomku.

3.4.3. Instance kao atributi

Kada se modeluje nešto iz realnog sveta i dodaje u kod, vremenom klasa postaje nepregledna. Nagomilava se gomila atributa i metoda, a fajl postaje preobiman. U tim situacijama velika klasa može da se pretoči u više manjih. Na primer izvede se klasa *Student* iz klase *Person*.

Student ide na fakultet koji ima ime, adresu, mesto. Kako se ne bi nagomilavale informacije u klasi *Student*, napraviće se klasa *College*, a zatim kompozicijom modelovati odnos. Klasa *Student* će sadržati instancu klase *College*, odnosno posedovati njen objekat. Primer je dat na slici 9.

```
class College():
    def __init__(self, name="FTN", city="NS"):
        self.name = name
        self.city = city

class Student(Person):
    def __init__(self, fullname):
        self.index = 123
        self.college = College()
        super().__init__(fullname)
    def study(self):
        print("I'm studying.")
    def sleep(self):
        print(self.fullname.title() + " doesn't have time to sleep")
```

Slika 9. Primer kompozicije, definisanje novih atributa i metoda za klasu potomka i gaženje roditeljske metode

3.4.4. Višestruko nasleđivanje

Python podržava jedan neuobičajen koncept objektno-orijentisanog programiranja, a to je višestruko nasleđivanje. U mnogim drugim programskim jezicima, ovaj koncept se izbegava s obzirom da povlači sa sobom poznati problem koji se naziva problem dijamanta. U definisanju klase je potrebno navesti sve roditelje odvojene zapetom. Kako bi se izbegla dvosmislenost, prilikom potrage za atributima koji su nasleđeni iz hijerarhije koristi se pretraga u širinu i koja pretražuje sleva nadesno. Prvo se gleda u klasi deteta, ukoliko se ne nađe tu atribut, prelazi se na prvu roditeljsku klasu. Ukoliko se ni ne pronade u prvoj roditeljskoj, tad se gleda u okviru podstabla roditeljske prve klase. Ako ni tu ne nađe, prelazi se na sledeću roditeljsku klasu. Dinamički odabir i raspored je neophodan, s obzirom da svi primeri upotrebe višestrukog nasleđivanja uključuju jedan ili više slučaj dijamantske veze u hijerarhiji. Najbolji primer ovoga jeste situacija da u okviru *Pythona*-a sve klase nasleđuju klasu *object*, tako da prilikom upotrebe bilo kog slučaja višestrukog nasleđivanja postoji više od jednog puta koji vodi do klase *object*.

3.5. Upotreba klasa iz drugih fajlova

Python omogućava mehanizam za modularizaciju koji omogućava da se klase uvedu u druge delove programa iako se nalaze u različitim fajlovima. Sintaksa je sledeća: *from student import Student*. Ključna reč *from* govori iz kog modula je potrebno uključiti, a ključna reč *import* šta se uključuje.

3.6. Preklapanje operatora

Preklapanje operatora je mogućnost da se za većinu operatora, definišu drugačije uloge ako su operandi objekti, odnosno instance klase. Na primer postoji klasa *Car* koja sadrži ime samog automobila i ljude u kolima. Operator *“+=”* bi imalo smisla da puni automobil, odnosno da dodaje ljude. Potrebno je redefinisati metodu *__iadd__()* koja predstavlja operator *“+=”* i napisati kako treba da se ponaša. Kao rezultat *print()* funkcije dobijaju se tri osobe u automobilu. Primer je dat na slici 10.

```
class Car:
    def __init__(self, people, name):
        self.people = list(people)
        self.name = name
    def __iadd__(self, person):
        self.people.append(person)
        return self

car = Car(['Pera', 'Zika'], 'Fiat Punto')
car += 'Sima'
print(car.people)
```

Slika 10. Primer preklapanja operatora

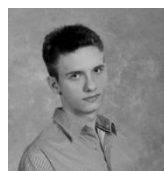
4. ZAKLJUČAK

U radu je opisan programski jezik *Python*, njegovi osnovni elementi, a poseban akcenat je dat na objektno-orijentisanoj paradigmi. Danas, većina koda je napisana u objektno-orijentisanim jezicima. Filozofija dizajna programskog jezika *Python* je jednostavnost i čitljivost koda. Osnovni tipovi i organizacija sa modulima, omogućavaju da se piše modularan kod. Iako, na prvi pogled možda deluje kao sintaksno jednostavan jezik, *Python* radi sve što i drugi OOP jezici, čak i više. Ima podršku za preklapanje operatora i za višestruko nasleđivanje, što mnogi jezici nemaju. Objekat nikad nije vezan za jednu klasu, može se dinamički promeniti, odnosno preuzeti neki drugi tip. Mana *Pythona* je što narušava jedan od principa objektnog-orijentisanog programiranja: enkapsulaciju. Nema ključne reči *“private”*, *“protected”* i *“public”*. Sve je javno. Postoji notacija *name mangling* koja lokalizuje imena promenljivih, ali ne sprečava pristup van klase. To je tehnika koja se koristi da reši razne probleme kada treba razrešiti unikatna imena za programske entitete. Takve promenljive se mogu nazvati pseudo-privatne.

5. LITERATURA

- [1] Learning Python, Marc Lut, Fifth edition 2013
- [2] Python crash course, Eric Matthes, 2016
- [3] Python Cookbook, David Beazley and Brian K. Jones, Third edition 2013
- [4] Zvanična dokumentacija: <https://docs.python.org/3/>
- [5] <https://python.swaroopch.com>
- [6] <https://realpython.com/>

Kratka biografija:



Stefan Varajić rođen je u Novom Sadu 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Primenjene računarske nauke i informatika odbranio je 2018.god.kontakt: kontakt: stefans94@hotmail.com

IMPLEMENTACIJA APLIKACIJE ZA GENERISANJE TESTOVA ZA PROGRAM OTISAK UZ OSLONAC NA SPRINGBOOT I REACTJS**IMPLEMENTATION OF THE APPLICATION FOR GENERATING TESTS FOR THE OTISAK SOFTWARE USING SPRINGBOOT AND REACTJS**

Olja Miljatović, Zorica Suvajdžin Rakić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je prikazana specifikacija i implementacija sistema za generisanje testova za proveru znanja. Testovi su kolekcije pitanja i parametara koji ga opisuju. Moguće je dodavati ili menjati predmete, oblasti, pitanja i odgovore u cilju da se formira odgovarajući test. Testovi koji se generišu koriste se za teorijske provere putem programa Otisak.

Ključne reči: Generisanje testova, SpringBoot, Otisak

Abstract – This paper presents the specification and the implementation of a system for generating examination tests. Tests are collections of questions and parameters. It is possible to add or change subjects, sections, questions and answers with the aim of forming an appropriate test. Tests that are generated are used for theoretical checks through the program Otisak.

Keywords: Generating tests, SpringBoot, Otisak

1. UVOD

Usled potrebe za velikim brojem testova za proveru znanja potrebno je što kraće vreme za pripremu pojedinačnog testa. Za svaki test potrebno je definisati kombinaciju pitanja iz istih ili različitih oblasti, odrediti težinu pitanja i dopuniti parametre koji opisuju test. Sadržaj za testove su prethodno uneta i sačuvana pitanja koja su razvrstana po predmetima i oblastima.

Pri kreiranju pitanja javlja se potreba različitih tipova pitanja kao što su pitanja sa više tačnih odgovora, samo jednim tačnim odgovorom ili ručno unošenje tačnog odgovora. Takođe se javljaju potrebe za parcijalnim dodeljivanjem bodova ali i negativnih bodova.

Aplikacija za generisanje testova omogućava primenu navedenih stavki i kao rezultat formira test fajl koji služi kao ulaz za program Otisak. Program Otisak se koristi na katedri za primenjene računarske nauke i informatiku i služi za vizuelizaciju testa i evidentiranje studentskih odgovora prilikom testiranja.

2. OPIS INFORMACIONOG SISTEMA

Formirani test, kao krajnji rezultat, povlači sa sobom sledeće pojmove :

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Zorica Suvajdžin Rakić.

2.1 Oblasti

Oblasti predstavljaju celine u organizaciji pojedinačnog predmeta. Svaki predmet se sastoji od 1 ili više oblasti dok oblast u sebi može imati proizvoljan broj pitanja. Funkcija oblasti je lakše snalaženje i formiranje testa za određenu celinu ispitivanja.

2.2 Pitanja

Osnovu za rad ovog informacionog sistema predstavljaju pitanja. Pitanja su raspoređena u nastavne oblasti. Pitanje se s toga nalazi u sledećoj ugnježdenoj strukturi:
predmet -> oblast -> pitanje

Formiranje pitanja se može izvršiti na sledeće načine :

- Parsiranje gotovih fajlova sa pitanjima
- Ručno dodavanje pitanja uz pomoć forme

Fajlovi koji se učitavaju su fajlovi prethodno korišćeni u procesu formiranja testova koji su služili kao baza podataka. Odabrani fajlovi sadrže listu pitanja, u striktno definisanom formatu. Ovaj način implementacije se javlja kao potreba za iskoristivost prethodno formiranih pitanja.

Ručno dodavanje pitanja zasnovano je na popunjavanju forme u kojoj su ponuđene mogućnosti za formulisanje teksta i naziva pitanja, vrste pitanja, ponuđenih tačnih i netačnih odgovora, težine pitanja i oblasti kojoj pripada. Generičnost ove forme omogućava da se pitanja na jednostavan način dodaju za bilo koji predmet.

Izdvojene su tri vrste pitanja :

1. pitanja sa izborom 1 od n (*radio button*). Njegova oznaka je /t i primer je dat na slici 1:

```
*shell*pl  
/t ili /te  
Listanje direktorijuma u <i>bash shell</i>-u se ostvaruje komandom:  
=slika.png  
-cat  
-pwd  
-cd  
-less  
+ls
```

Slika 1. Pitanje sa jednim tačnim odgovorom

Radio button pitanje je pitanje kod kog je potrebno označiti jedan tačan odgovor od više ponuđenih.

Kod pitanja sa jednim tačnim odgovorom, u ponuđenim odgovorima može se nalaziti samo jedan sa oznakom plus (+) i on predstavlja tačan odgovor. Ostali odgovori ispred teksta sadrže oznaku minus (-) i može ih biti proizvoljno mnogo.

2. pitanja sa izborom m od n, gde je $m < n$ (*checkbox*). Oznaka ove vrste je **/n** i dat je primer na slici 2:

```
*algebra*2
/c ili /ce
Rezultat sabiranja <c>5+3</c> je isti kao rezultat:
+oduzimanja 13-5
+deljenja 16/2
-mnozenja 2*5
+stepenovanja 2^3
-sabiranja 2+7
```

Slika 2. Pitanje sa više tačnih odgovora

Suprotno od pitanje sa jednim tačnim odgovorom, kod ove vrste pitanja postoji proizvoljan broj i tačnih i netačnih odgovora.

3. pitanja sa unošenjem teksta kao tačnog odgovora (*freetext*). Oznaka ove vrste je **/u** a primer je dat na slici:

```
*p1
/u ili /ue
Ako imaš 127 jabuka i nekom daš sve osim 39, koliko ti ostaje?
(odgovor upisati kao <b>dekadni broj</b> bez vodećih nula)
```

\Preostalo jabuka\39

Slika 3. Pitanje sa unošenjem teksta kao odgovora

Kod ove vrste pitanja ne postoji mogućnost odabira ponuđenih odgovora već se odgovor unosi popunjavanjem polja za unos.

Težine pitanja su zasnovane na osnovu kategorija: teški, srednji i laki. Moguće je ručno dodeliti težinu pitanju, a u slučaju da se ona izostavi, pitanje će pripasti lakoj kategoriji.

2.3 Testovi

Testovi predstavljaju celinu sastavljenu od pitanja. Pored pitanja, test obuhvata informacije o testu kao što su naziv, oznaka i grupa. Osim pitanja i informacija, potrebno je definisati vrednosti parametara koje program Otisak koristi pri realizaciji testa.

Primer takvog parametra je NEGOD koji određuje od koliko netačno odgovorenih pitanja će se računati negativni poeni. Ako je ovaj parametar npr. 3 i ako je neko na testu pogrešno odgovori na manje od 3 pitanja, neće se računati negativni poeni, nego će za ta pitanja dobiti 0 poena.

3. KORISĆENE TEHNOLOGIJE

Pri projektovanju rešenja odabrano je da se aplikacija bazira na klijent-server arhitekturi a da se za sam razvoj koristi *SpringBoot*, uz *Maven*. Front-end aplikacije bazira se na *ReactJS*-u. Baza podataka nad kojom se operiše je *H2* baza.

3.1 Klijent-server arhitektura

Model klijent-server je struktura distribuiranih aplikacija koja zadatke deli između dobavljača resursa nazvanih server i objekata koji zahtevaju usluge, nazvanih klijent [1]. Klijent i server najčešće komuniciraju preko mreže nalaze se na različitim hardverskim komponentama a komunikacija radi tako što klijent preko mreže šalje resurs ili zahtev ka serveru.

Zahtev se procesira i nakon toga se rezultat dostavlja klijentu koji je uputio zahtev.

3.2 SpringBoot

Spring je *open-source* Java platforma sa izuzetno kvalitetnom podrškom za razvoj aplikacija. Programiranje u *Spring*-u je vezano za koncept inverzije kontrole, poznatim pod nazivom *Dependency Injection*. Ovaj koncept je praktično obezbedio kreiranje loosely coupled objekata i njihovo automatsko povezivanje u aplikacionu celinu od strane framework-a [2].

3.3 ReactJS

React je *JavaScript* biblioteka za kreiranje korisničkog interfejsa. Izgradnjom komponenti za svako stanje u aplikaciji doprinosi da je *React* u mogućnosti da vrši iscrtavanje i menjanje tačno onih komponenti u kojima se podaci menjaju [3].

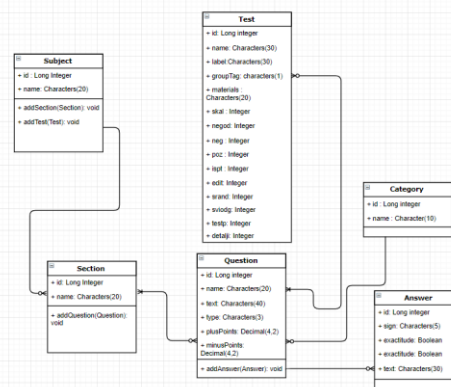
3.4 H2 baza podataka

Baza podataka je organizovana kolekcija podataka [4]. Najznačajniji razlog korišćenja ove baze u toku razvoja rešenja je njena brzina. To je takozvana *in-memory* baza podataka koja predstavlja bazu smeštu u glavnoj memoriji računara. Pristup podacima u memoriji eliminiše vreme traženja prilikom upita nad podacima, što omogućava brže i pouzdanije performanse od diska [5].

4. IMPLEMENTACIJA I DEMONSTRACIJA

Kao počećni korak u procesu implementacije softverskog rešenja neophodno je projektovati sistem i modelovati bazu podataka sa kojom će se rukovati. Razvoj konceptualnog modela omogućava razumevanje fizičkih procesa sistema. Iz ovog modela tj. Iz njegovih entiteta-veza relacija dobija se relaciona baza podataka.

Na slici 4 prikazan je konceptualni dijagram sistema.



Slika 4. Konceptualni dijagram

Entitet *Question* predstavlja pitanje i centralni je deo konceptualnog modela. Ovaj entitet sadrži sve informacije koje su potrebne da se definiše pitanje u realnom sistemu. Povezan je sa entitetima *Section* (oblast) tako što se svako pitanje mora nalaziti u bar jednoj oblasti i indirektno sa entitetom *Subject* (predmet) jer se oblast uvek nalazi u okviru nekog predmeta.

Odgovor je predstavljen entitetom *Answer*. Atributi omogućavaju označavanje da li je odgovor tačan ili netačan (atribut *exactitude*) kao i vizuelni prikaz tačnosti (atribut *sign*).

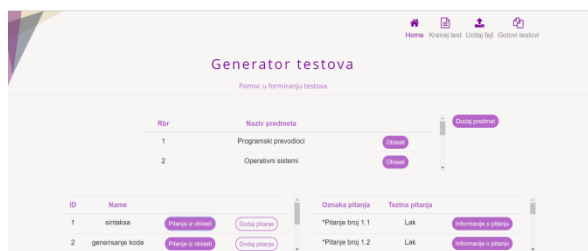
Jedno pitanje može imati proizvoljan broj odgovorova a od govor se mora nalaziti u tačno jednom pitanju. Ova veza prikazana je na slici 4.

Test, kao jedan od glavnih pojmova, modelovan je entitetom *Test*. Povezanost entiteta *Test* i *Question* omogućava da se test ne može kreirati ako nema odabranih pitanja ali da pitanja mogu da postoje nezavisno od postojanja testa. Veza koja prikazuje ovaj odnos je *many-to-many* veza sa restrikcijama prikazana na slici 4.

U ovoj sekciji takođe su prikazani i procesi implementirani u sistemu i to su:

- Prikaz predmeta, oblasti i pitanja u okviru oblasti
- Implementacija učitavanje fajla sa pitanjima
- Implementacija funkcionalnosti dodavanje pitanja
- Implementacija funkcionalnosti dodavanja testova
- Izlistavanje gotovih testova

Stranica za izlistavanje predmeta, oblasti u okviru predmeta i pitanja u okviru odabrane oblasti prikazana je na slici 5.



Slika 5. Izlistavanje predmeta, oblasti i pitanja

Na slici 5 prikazana je tabela sa izlistanim predmetima dok je za svaki predmet moguće odabrati prikaz oblasti klikom na dugme *Oblasti*. Osim mogućnosti za prikaz, postoji opcija dodavanja novog predmeta. Nakon odabira oblasti iz koje se žele pregledati pitanja, moguće je pogledati i informacije o pitanjima. Informacije o pitanju omogućavaju pregled tačnih i netačnih odgovora, broja bodova i kategorije kojoj pitanje pripada.

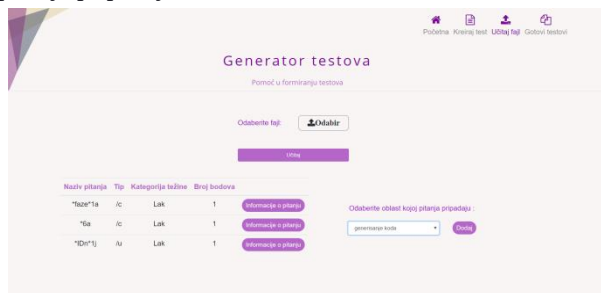
Na slici 6 prikazana je stranica na kojoj se nalazi forma za kreiranje testa. Crvenom bojom oivičene su sekcije u postupku kreiranja. Prva je sekcija za odabir osnovnih informacija o testu. Nakon nje sledi odabir parametara a zatim ponuđena i odabrana pitanja za test.



Slika 6. Forma za kreiranje testa

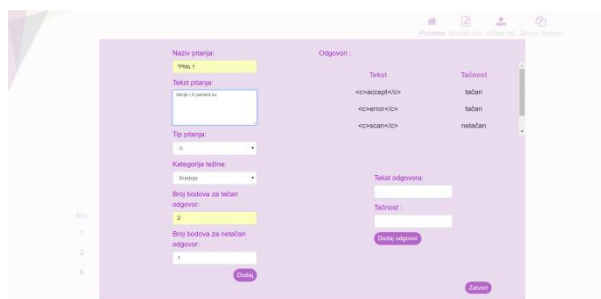
Na slici 7 predstavljena je stranica za učitavanje fajla sa gotovim pitanjima. Učitavanje *txt* dokumenta tačno

određenog formata predstavlja jedan od načina za dodavanje pitanja u sistem. Nakon odabira fajla klikom na Učitavanje, potrebno je odabrati oblast kojoj učotana pitanja pripadaju.



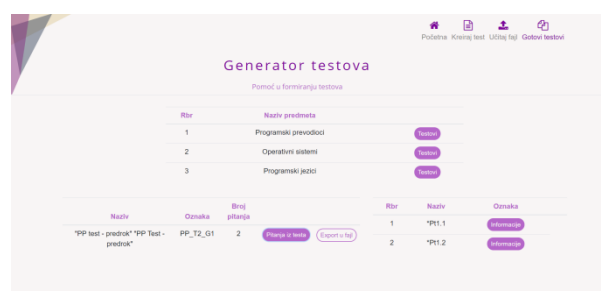
Slika 7. Učitavanje fajla sa pitanjima

Dodavanje novog pitanja sastoji se iz više koraka. Akcija dodavanja pitanja počinje na odabir dugmeta *Dodaj pitanje* a rezultira otvaranjem novog dijaloga sa formom za unos kao sa slike 8.



Slika 8. Forma za dodavanje pitanja

Aktivacija izlistavanja svih testova vrši se odabirom u glavnom meniju, u delu sa oznakom *Gotovi testovi*. Na slici 9 prikazana je stranica sa tabelom koja sadrži sve predmete a nakon toga i testove za odabrani predmet. Na istom mjestu moguć je i uvid u pitanja za test.



Slika 9. Izlistavanje svih testova

5. ZAKLJUČAK

Kreirana je aplikacija koja olakšava ručno generisanje kako jednog tako i velikog broja testova za veoma kratko vreme. Sadržaj koji se koristi za formiranje testova predstavljen je u obliku pitanja koja se na jednostavan način mogu dodati u sistem.

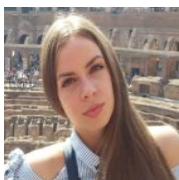
Intuitivan interfejs omogućuje formiranje tekstualnih fajlova sa pitanjima i testovima i eliminiše brigu o formatu fajlova.

Kao sledeći korak u realizaciji biće omogućeno automatsko generisanje testa. Automatsko generisanje zahteva odabir oblasti, broja pitanja i eventualno koliko pitanja iz koje kategorije težina.

6. LITERATURA

- [1] "Distributed Application Architecture", Sun Microsystem (Preuzeto 2009-06-16)
- [2] Almir Pehratović(2017). „Uvod u SpringFramework“ [Internet] Dostupno na: <https://www.info.ba/ostalo/specijal/8291/uvod-u-spring-framework>
- [3] <https://reactjs.org/> (pristupljeno u avgustu 2018.)
- [4] „Database – Definition of database by Merriam-Webster“ [Internet]. Dostupno na : <http://www.-webster.com/dictionary/database>
- [5] Michael Vizard „The Rise of In-Memory Databases“, Slashdot, January 2013

Kratka biografija:



Olja Miljatović rođena je u Sarajevu 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Računarstvo i informatika odbranila je 2018.god.

Kontakt: olja.miljatovic@uns.ac.rs

MODEL ZA SIMULACIJU RADA TOPLOTNE PODSTANICE SA INDIREKTNOM RAZMENOM TOPLOTE REALIZOVAN U PROGRAMSKOM PAKETU LABVIEW**SIMULATION MODEL OF HEAT SUBSTATION WITH INDIRECT HEAT EXCHANGE IMPLEMENTED IN LABVIEW SOFTWARE PACKAGE**

Lazar Starčević, *Fakultet tehničkih nauka, Novi Sad*

ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je kreiran simulacioni model realne toplotne podstanice, sa indirektnom razmenom toplote, u programskom paketu LabView.

Ključne reči: Labview, toplotna podstanica, izmenjivač toplote, simulacija, model, Matlab

Abstract – In this paper simulation model of real heat substation with indirect heat exchange was created in LabView software package.

Keywords: Labview, heat substation, heat exchanger, simulation, model, Matlab

1. UVOD

U ovom radu je detaljno analiziran rad toplotne podstanice sa indirektnom razmenom toplote i na osnovu analize i analitički dobijenih podataka kreiran simulacioni model koji oslikava ponašanje realne toplotne podstanice. Kada se kaže model toplotne podstanice misli se na krucijalne elemente svake toplotne podstanice sa indirektnom razmenom toplote a to su izmenjivač toplote i mešni ventili.

Uloga izmenjivača toplote je da prenese toplotnu energiju iz primarne mreže toplifikacije grada i toplifikacione mreže individualnih stambenih objekata.

Mešni ventili se koriste za prilagođavanje temperature vode sa izmenjivača toplote za različite tipove grejnih tela u objektima (radijatori, podno grejanje, ventilator-konvektorsko itd.).

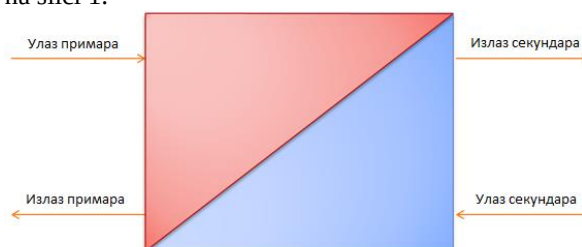
Model toplotne podstanice je rađen sukcesivno. Najpre je u programsku paketu Matlab, u Simulink blok-dijagram okruženju kreiran model izmenjivača toplote a zatim i model mešnog ventila. Kada su modeli međusobno povezani i istestirani pristupljeno je implementaciji modela i upravljanja radom toplotne podstanice u programskom paketu Labview. U istom okruženju je urađen i grafički korisnički interfejs-vizuelizacija za interakciju sa operaterima. Nakon uspešno realizovanih prethodno navedenih faza izvršeno je i povezivanje sa jednim od sistema za upravljanje zgradama (eng. Building Management System).

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Velimir Čongradac.

2. ANALIZA I REALIZACIJA MODELA**2.1. Model izmenjivača toplote**

Na samom početku bilo je neophodno izračunati i interpretirati temperature na izlazu primara i sekundara na osnovu poznatih vrednosti temperatura na ulazu istih, kao na slici 1.



Slika 1. – Temperature u modelu

U daljem tekstu ćemo sa T1prim označavati ulaz primara odnosno vodu/vodenu paru koja dolazi od toplane, sa T2prim izlaz primara (vodu koja se vraća u toplanu), sa T1sek ulaz sekundara (vodu koja se vraća iz objekta koji grejemo) a sa T2sek izlaz sekundara tj. vodu koja ide ka grejnim telima u objektu.

Na samom početku bilo je neophodno izračunati i interpretirati temperature na izlazu primara i sekundara na osnovu poznatih vrednosti temperatura na ulazu istih.

Da bismo izračunali temperaturu na izlazu sekundara (T2sek) iskoristili smo sledeću relaciju-formulu prenosa toplote.

$$T2sek[n+1] = T2sek[n] + (T1prim[n] - T1sek[n]) * k \quad (1)$$

Temperatura u narednom trenutku bila bi jednaka zbiru temperature na izlazu sekundara u prethodnom trenutku i proizvoda razlike temperatura na ulazu primara i sekundara i koeficijenta prenosa toplote.

Ova relacija bi bila važeća kada ne bismo uzeli u obzir gubitke energije odnosno u idealnom slučaju. Kada uzmemo u obzir sledeće parametre:

- gubitke energije,
- vrednosti temperature na ulazu primara i sekundara (T1prim, T1sek),
- koeficijent prenosa toplote (k),
- temperaturu spoljašnjosti (Tspolj)

i izvršimo simulaciju, dobije se temperatura na izlazu iz sekundara (T2sek) i prikazana je na grafiku tj. slici u nastavku.

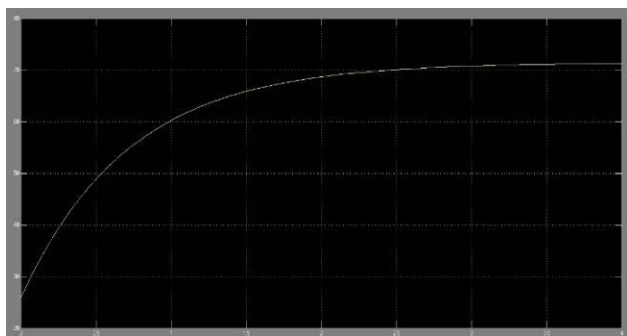
Na grafiku sa slike 2.a vidi se da za poznate vrednosti T_{1prim} ($90^{\circ}C$), T_{1sek} ($26^{\circ}C$), T_{spolj} ($25^{\circ}C$), temperatura T_{2sek} se posle određenog vremena ustali na $72^{\circ}C$, što je maksimalna temperatura koja se može dobiti na izlazu sekundara izmenjivača toplote koji razmatramo i koji smo uzeli kao primer za opservaciju.

Dalje u radu je za potrebe izračunavanja temperature na izlazu iz primara, iskorišćena sledeća formula održanja energije:

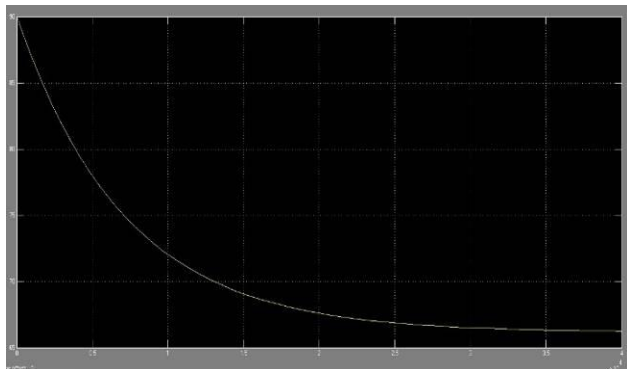
$$m_1 * (T_{1prim} - T_{2prim}) = m_2 * (T_{2sek} - T_{1sek}) \quad (2)$$

Ukupna utrošena količina energije na primaru jednaka je ukupnoj utrošenoj količini energije na sekundaru, odnosno, proizvod ukupnog protoka i razlike temperatura na primaru jednak je proizvodu ukupnog protoka i razlike temperatura na sekundaru.

Za unapred poznate vrednosti temperatura na ulazu primara i sekundara i izračunate temperature na izlazu sekundara, kao i za poznate protoke, dobije se karakteristika-temperatura na izlazu primara prikazana na slici 2.b.



Slika 2.a - Temperatura vode na izlazu sekundara

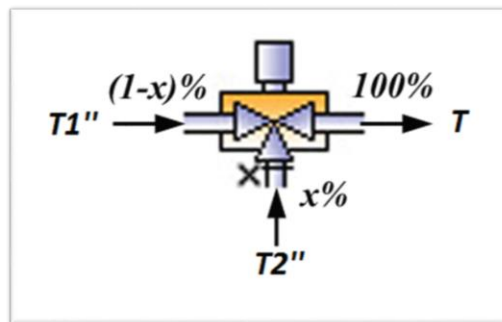


Slika 2.b - Temperatura vode na izlazu iz primara

Na slici se može videti da se za poznate vrednosti T_{1prim} ($90^{\circ}C$), T_{1sek} ($26^{\circ}C$), T_{spolj} ($25^{\circ}C$), kao i poznate masene vrednosti protoka na primaru m_1 ($42kg/s$), i ukupnog masenog protoka na sekundaru m_2 ($22kg/s$ - zbir protoka za sva tri tipa grejanja), dobije temperatura na izlazu primara koja se posle određenog vremena ustali na $66,4^{\circ}C$.

2.2. Model mešnog ventila

Da bi se dobila voda odgovarajuće temperatura u zavisnosti od tipa grejanja (radijatorsko, podno ili ventilator-konvektorsko) koristi se mešni ventil koji, kako mu samo ime kaže, meša vodu iz potisne grane u objekat (izlaz sekundara) i povratne grane iz objekta (ulaz sekundara).



Slika 3. - Skica mešnog ventila

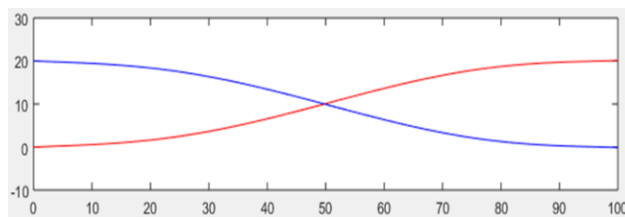
Kada govorimo o mešnim ventilima kod sistema grejanja najčešće razlikujemo dva tipa ventila po broju priključaka a to su trokraki i četvorokraki mešni ventili. Skica trokrakog mešnog ventila data je na slici iznad, gde je:

- T – ostvarena temperatura vode
- $T_{1''}$ – temperatura ulazne vode sekundara
- $T_{2''}$ – temperatura izlazne vode sekundara
- X – otvorenost ventila u procentima

Ako uzmemo u obzir da su pritisci u obe ulazne grane mešnog ventila jednaki, ostvarena temperatura vode na izlazu mešnog ventila računa se po sledećoj formuli:

$$T = T_{1''} * (1 - x) + T_{2''} * x \quad (3)$$

Ako ulazne grane trokrakog mešnog ventila obeležimo sa A i B kao na slici iznad, voda na izlazu ventila AB neće imati temperaturu koja će biti proporcionalna odnosno linearno zavisna od otvorenosti ventila. Imaće nelinearnu karakteristiku koja je prikazana na slici ispod.



Slika 4. - Karakteristika ventila u slučaju kada su pritisci u granama A i B izjednačeni ($p_A = p_B$)

Navedena karakteristika važi u slučaju kada su pritisci u ulaznim granama isti. Temperatura vode na izlazu mešnog ventila, kada su pritisci na ulaznim granama različiti i poznati, se računa na sledeći način:

$$T = (\text{protokA} * T_{2''} + \text{protokB} * T_{1''}) / \text{protokAB} \quad (4)$$

gde je:

- $T_{1''}$ – temperatura vode koja ulazi u sekundar (hladna)
- $T_{2''}$ – temperatura vode koja izlazi iz sekundara (topla)
- T – temperatura vode dobijena mešanjem $T_{1''}$ i $T_{2''}$

Da bismo izračunali temperaturu vode na izlazu mešnog ventila u slučaju kada pritisci na ulazu nisu jednaki, moramo izračunati protok na ulaznim granama. Protok na ulaznim granama se računa na sledeći način:

$$x = p1 \cdot O^5 + p2 \cdot O^4 + p3 \cdot O^3 + p4 \cdot O^2 + p5 \cdot O + p6 \quad (5)$$

$$\alpha = pB/pA \quad (6)$$

$$X = x \cdot (\alpha + (1 - \alpha) \cdot x^2)^{-0.5} \quad (7)$$

$$\text{protokA} = X \cdot \text{protokAB} \quad (8)$$

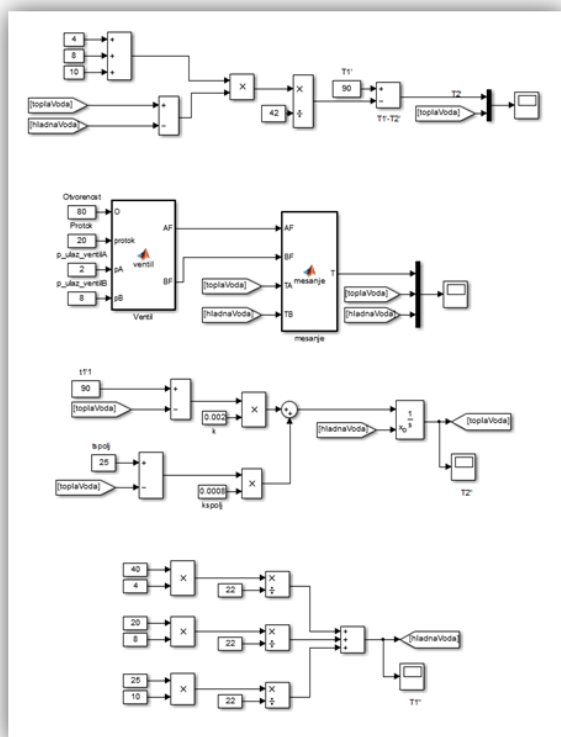
$$\text{protokB} = \text{protokAB} - \text{protokA} \quad (9)$$

gde je:

- x – aproksimirana karakteristika mešnog ventila
- O – otvorenost ventila
- pA – pritisak u grani A
- pB – pritisak u grani B

2.3. Realizacija modela u MATLAB-u

Za realizaciju modela toplotne stanice, pored standardnih Matlab funkcija i skripti korišćen je i Simulink dodatak za simulaciju. Na slici 29. je prikazan model toplotne podstanice kreiran u Simulink-u.



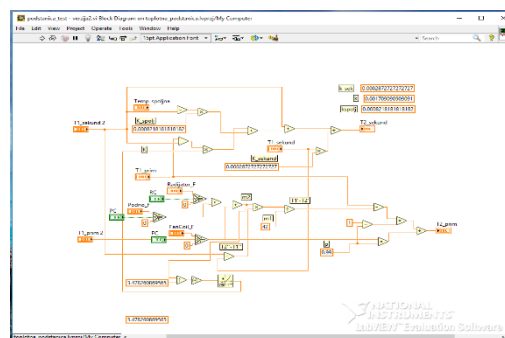
Slika 5. – Model toplotne podstanice kreiran u Simulink-u

2.4. Realizacija modela u LABVIEW-u

Nakon implementacije modela toplotne podstanice u Matlab-u, sledeći korak je implementacija modela u LabVIEW-u.

Na slici 6. prikazan je jedan od načina implementacije modela izmenjivača toplote, i to u posebnom VI-u. Ovaj VI ima više ulaza i izlaza ali četiri najvažnija za dalji tok programa i za dalju realizaciju su:

- temperatura vode koja ulazi u primar - analogni ulaza
- temperatura vode koja ulazi u sekundar - analogni ulaza
- temperatura vode koja izlazi iz primara - analogni izlaz i
- temperatura vode koja izlazi iz sekundara) analogni izlaz.



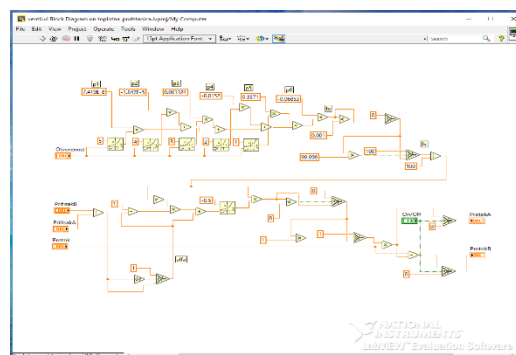
Slika 6. – LabVIEW blok dijagram za izmenjivač toplote

Na slici 7. je prikazan VI u kojem je imlementiran model mešnog ventila. Ulazi u ovaj VI su:

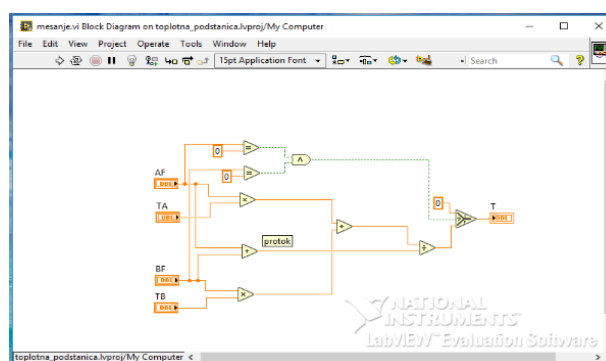
- otvorenost ventila (%),
- pritisak u grani A (bar),
- pritisak u grani B (bar),
- ukupni protok kroz ventil (kg/s).

Izlazi su:

- protok kroz granu A (kg/s) i
- protok kroz granu B (kg/s).



Slika 7. – LabVIEW blok dijagram u kome je realizovan mešni ventil

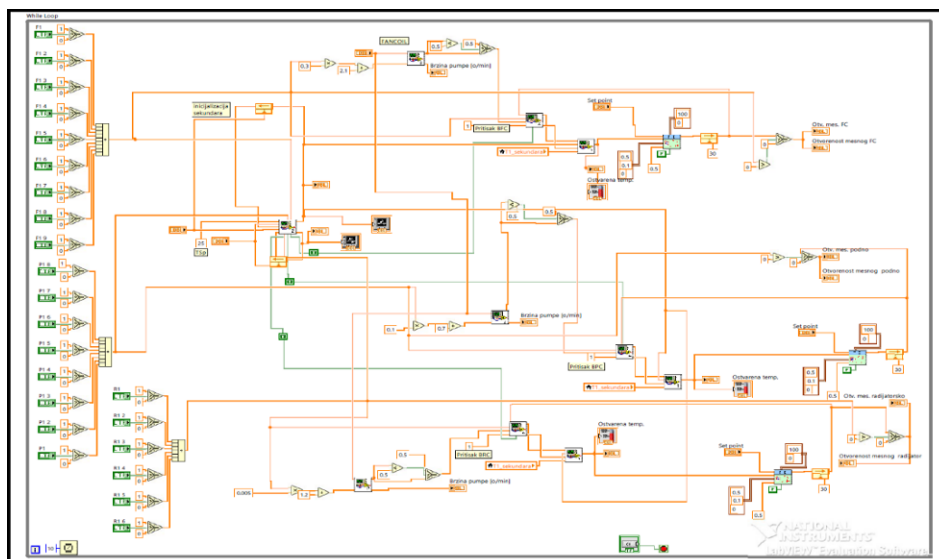


Slika 8. – LabVIEW blok dijagram u kome je realizovan mešni ventil

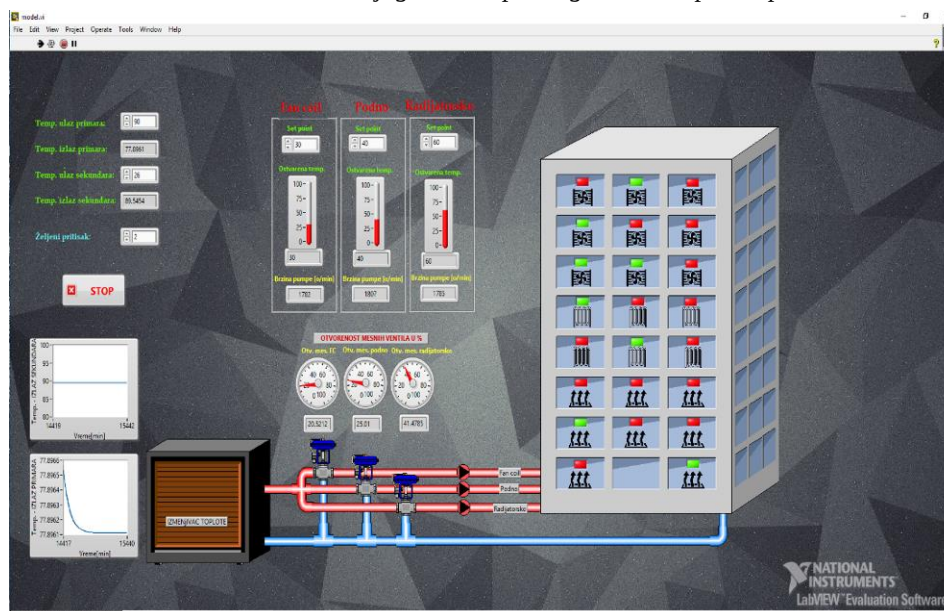
Na slikama u nastavku (slika 9. i slika 10.) prikazani su blok dijagram i front panel koji predstavljaju model toplotne podstanice gde su objedinjeni svi podsistemi koji su predstavljani u prethodnom tekstu (izmenjivač toplote, ventil, mešanje).

Implementirane su tri vrste grejanja: podno, radijatorsko i ventilator-konvektorsko. Za svaku vrstu grejanja imamo poseban mešni ventil, jer su potrebne različite temperature vode.

Temperature su podešive i korisnik ih može menjati i podešavati po želji.



Slika 9. – LabVIEW blok dijagram kompletnog modela toplotne podstanice



Slika 10. – LabVIEW front panel - vizuelizacija kompletnog modela toplotne podstanice

3. ZAKLJUČAK

U današnje vreme kada je energija skupa, simulacioni model toplotne podstanice je sa aspekta uštede energije veoma interesantan. Pored pravilnog dimenzionisanja elemenata podstanice, na modelu se mogu vršiti testovi kako bi se došlo do optimalnog upravljanja podstanicom, a pri tom su testovi i simulacije skoro pa besplatni u odnosu na uštede koje se na osnovu njih mogu ostvariti.

4. LITERATURA

- [1] Dominik Rutz, Christian Doczekal, Richard Zweiler, Morten Hofmeister, Linn Laurberg Jensen (2017): Modularni sistemi daljinskog grejanja i hlađenja
- [2] [6] Reknagel, Šprenger, Šramek, Čeperković (1996): Grejanje i klimatizacija, Interklima
- [3] Yaran Wang, Shijun You, Huan Zhang, Xuejing Zheng, Shen Wei, Qingwei Miao, Wandong Zheng

(2017): Operation stability analysis of district heating substation from the control perspective

Kratka biografija:



Lazar Starčević je rođen 17. avgusta 1993. godine u Šapcu. Nakon završene osnovne škole "Jevrem Obrenović" upisuje Tehničku školu Šabac u Šapcu, smer Elektrotehničar računara koju završava 2012. godine. Iste godine upisao je Fakultet tehničkih nauka u Novom Sadu, odsek elektrotehnika i računarstvo, smer Računarstvo i automatika, usmerenje Automatika i upravljanje sistemima i završio osnovne akademske studije 2016. kontakt: starceviclazar@gmail.com

INTEGRACIJA SCADA UREĐAJA**INTEGRATION OF SCADA DEVICES***Čaba Šagi, Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – Ovaj rad predstavlja primer automatskog konfigurisanja procesnih kontrolera u SCADA sistemu. Nakon prikazanog teorijskog dela opisan je i predlog rešenja za automatsko konfigurisanje.

Abstract – This paper presents an example of automatic configuration of process controllers. After the theoretical part a proposal for the solution is also presented.

Cljučne reči: SCADA, integracija, DNP3, IEC 60870-101/104, Modbus, SCL

1. UVOD

Razvoj i unapređenje tehnologije je omogućio nadzor i upravljanje današnjih proizvodnih sistema i postrojenja. Nadzor i upravljanje ovakvih sistema se obavlja pomoću računara. Ti računarski sistemi se nazivaju akviziciono-upravljački sistemi tj. SCADA (*Supervisory Control and Data Acquisition*) sistemi.

Automatizacija je bila potrebna jer je smanjila troškove proizvodnje, povećala raspoloživost i integritet, kao i fleksibilnost, proširivost, pouzdanost i performanse sistema. Fizički procesi koji se nadziru mogu biti različitog nivoa složenosti, na koje utiče veličina sistema kao i geografska rasprostranjenost. Veoma je važno uočiti da ovakvi sistemi imaju i hardverske i softverske komponente.

Kontrolni centri su povezani sa udaljenim telemetrijskim stanicama (*RTU – Remote Terminal Unit*) i programabilnim kontrolerima (*PLC – Programmable Logic Controller*). Komunikacija se vrši pomoću industrijskih komunikacionih protokola.

Da bi ta komunikacija bila moguća, model sistema u softveru koji se izvršava na serverima u kontrolnim centrima i model sistema na udaljenim uređajima moraju biti usklađeni.

Tema ovog rada jeste generisanje i otpremanje konfiguracije potrebne za udaljene telemetrijske stanice, na osnovu unetog SCADA modela u kontrolnim centrima.

2. TEORIJSKE OSNOVE**2.1. Osnove SCADA sistema**

Pod akviziciono – upravljačkim sistemom (AUS) podrazumeva se skup namenskih prostorno distribuiranih, međusobno povezanih računarskih modula, čiji je zajednički cilj ostvarenje funkcija automatskog nadzora i/ili upravljanja fizičkim procesom u realnom vremenu.

Osnovna funkcija AUS sistema je ciklična akvizicija digitalizovanih odmeraka različitih fizičkih veličina koje određuju stanje proizvoljnog tehnološkog (fizičkog) procesa. Akvizicija podataka je dvosmerna, zato što obuhvata ulazne podatke (merjenja) i izlazne komande (komande). Formiranjem baze merenih podataka u računaru stvara se osnova za proveru i prikaz trenutnog stanja fizičkog procesa, odnosno za obavljanje efikasnog nadzora nad njim. Istovremeno, izvršenjem aplikativne upravljačke podrške moguće je odrediti i zahtevati (kontrolne) aktivnosti, tj izvršiti upravljanje nad fizičkim sistemom. Funkcije nadzora i upravljanja uključuju komunikaciju AUS sistema sa ljudskom posadom. [1]

U pogledu geografske topologije, moguće je podeliti SCADA sisteme na dve grupe.

Prva grupa SCADA sistema predstavlja akviziciono upravljačke sisteme koji upravljaju geografski razmeštenim sistemima. Kompleksnost kod ovakvih sistema često uzrokuje greške u merenjima i komunikacione probleme. Takvi problemi sprečavaju da krajnju odluku donese računar, pa zbog toga uključuju ljudsku posadu u donošenje krajnje odluke.

U drugu grupu akviziciono upravljačkih sistema spadaju sistemi koji upravljaju tehnološkim operacijama u prostorno ograničenim postrojenjima. Kod ovakvih sistema komunikacioni problemi su jako retki zahvaljujući pouzdanosti komunikacione mreže. Algoritmi za upravljanje su često vrlo složeni i ne zahtevaju ljudsku kontrolu.

2.2 Smart Grid

Smart Grid (pametna mreža) je električna mreža koja obuhvata razna operativna i energetska merila uključujući pametne merače, pametne uređaje, energetske efikasne resurse i obnovljive izvore energije.

Smart Grid se pojavio početkom 21. veka. Ekonomičniji obnovljivi izvori energije i reorganizacija električne mreže je uticalo na kreiranje ovog pojma.

Smart grid je kompleksan sistem koji ima razvijene različite napredne funkcionalnosti, funkcionalnosti za optimizaciju rada električne mreže i funkcionalnosti koje olakšavaju posao operatorima mreže. *Smart Grid* obuhvata niz upravljačkih sistema, kao što su:

- *Energy Management System (EMS)* – sistem za upravljanje prenosom električne energije;
- *Distribution Management System (DMS)* –

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Pavlica, vanr. prof.

sistem za upravljanje distribucijom električne energije;

- *Outage Management Sytem (OMS)* – sistem za praćenje otkaza u elektroenergetskoj mreži;
- *Work Management Sytem (WMS)* – sistem za organizacija rada intervetnih ekipa;
- *Geographic Information Sytem (GIS)* – sistem za geografski prikaz elektroenergetske mreže;
- *Meter Data Management (MDM)* – sistem za naplatu potrošnje električne energije;
- *Advanced Metering Infrastructure (AMI)* – sistem za komunikaciju sa kućnim brojilima.

Jedna buduća komponenta Smart grid sistema je *Device Management System*, koja služi za konfigurisanje i praćenje izmena procesnih kontrolera. Funkcionalnost ovog sistema, koja automatski konfigurira udaljene terminalne jedinice je tema ovog rada.

2.3. Procesni kontroler

Procesni kontroler je računarska stanica koja služi za prikupljanje mernih signala sa senzora i generisanje pobuda na aktuatorima. Procesni kontroleri su smešteni u samom postrojenju i napravljeni su tako da mogu da rade u uslovima ekstremne temperature i elektromagnetskih smetnji.

Postoji nekoliko vrsta procesnih kontrolera: *Programmable Logic Controller*, *Remote Telemetry Unit*, *Intelligent Electronic Device*.

2.3.1. PLC (Programmable Logic Controller)

PLC kontroleri su i dalje osnovna komponenta za automatizaciju industrijskih postrojenja. U industrijskim pogonima PLC se najčešće bavi automatizacijom pojedinačnih mašina. Uključeni u fieldbus mreže, PLC uređaji mogu međusobno komunicirati i autonomno učestvovati u nadzoru i upravljanju nad celim postrojenjem [1].

2.3.2 RTU (Remote Telemetry Unit)

Udaljena telemetrijska stanica je uređaj koji je namenjen za prikupljanje i praćenje geografski dislociranih procesa. Primeri takvih sistema su elektroenergetski, gasovodni i naftovodni sistemi. Najčešće se ti uređaji kače na WAN (*Wide Area Network*) i prosleđuju prikupljene podatke nadzorno upravljačkom sistemu.

2.3.3. IED (Intelligent Electronic Device)

Inteligentni elektronski uređaj je specijalizovani procesni kontroler koji je namenjen za kontrolu prekidača, transformatora ili baterija kondenzatora. Takođe mogu upravljati i nad zaštitnim relejima.

3. INDUSTRIJSKI PROTOKOLI I INDUSTRIJSKE MREŽE

Uobičajeno ime za klase protokola specifičnih za SCADA sisteme je industrijski protokoli i industrijske mreže. Na slici 1 je prikazana kategorizacija najpopularnijih industrijskih protokola. Industrijski protokoli se mogu podeliti na nekoliko kategorija prema tipu učesnika u komunikaciji i njihovim ulogama:

- Senzorski protokoli
- Fieldbus protokoli
- Telemetrijski protokoli

● Međusistemski protokoli

Ime	Klasa	Senzorski	Fieldbus	Telemetrijski	Međusistemski	Komentar
HART		✓				Analogna i digitalna sprega sa smart senzorima
Modbus		✓	✓	✓		Protokol sa najširojom primenom
CAN			✓			Potekao iz auto industrije - Bosch
CIP/DeviceNet			✓			Aplikativni protokol iznad CAN-a
ProfiBus		✓	✓			Familija protokola za primenu u procesnoj industriji, izvorno Siemensov - kasnije DIN i IEC standard
Interbus			✓			Sprega sa distribuiranim UI modulima i kontrolerima
Foundation Fieldbus		✓	✓			Distribuirano upravljanje na nivou senzora-aktuatora, za kritične aplikacije poput rafinerija
AS-i		✓	✓			Automatizacija zgrada (HVAC), 2 žična sprega
OSGP		✓		✓		Evropski protokol za smart grid integraciju
IEC 60870-5-103			✓			IED sprega, trafo stanice
IEC 60870-5-101				✓		Evropski standardni EES protokol, UART
IEC 60870-5-104				✓		Evropski standardni EES protokol, TCP
DNP3				✓		Američki standardni EES protokol, UART/TCP
Sindec H1				✓		Siemensov PLC protokol
IEC 61850			✓			Automatizacija transformatorskih i razvodnih postrojenja
ICCP/IEC 60870-6				✓		Povezivanje SCADA sistema – kontrolnih centara
OPC				✓		Integracija sistema u Windows/DCOM okruženju
OPC UA				✓		Integracija sistema, unificirana arhitektura

Slika 1: Kategorizacija industrijskih protokola [1]

3.1. DNP3

Westronic Incorporated je razvio *DNP3* standard između 1992. i 1994. godine, sa namerom da definišu prvi otvoreni, koristan standard u industriji. *Westronic* je dobro poznavao tu oblast, i pokušao je da ujedini dobre odlike tadašnjih protokola [2]:

- Mogućnost slanja iste poruke na različite uređaje;
- Beleženje vremena podataka;
- Sinhronizacija vremena na uređajima;
- Oznaka validnosti podataka;
- Različiti formati podataka;
- Definisanje grupe podataka;
- Slanje poruka samo pri izmeni vrednosti;
- Indikacije stanja uređaja.

Na slici 2 vidi se model *DNP3* udaljenih tačaka. Jasno se vidi da tačku jedinstveno identifikuje njen tip i indeks.

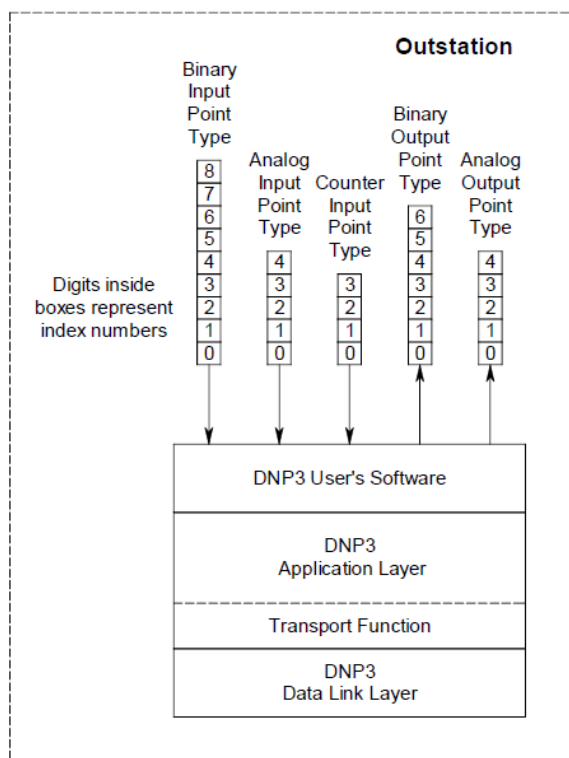
3.2. IEC 60870-5-101/104

Međunarodna tehnička komisija *IEC (International Electrotechnical Commission)* je svetska organizacija za standardizaciju. Jedan od ciljeva komisije je standardizacija komunikacione sprege *SCADA* stanice i *PLC* kontrolera. Od sredine 90-ih godina prošlog veka, *IEC* je objavio četiri združena standarda komunikacionih protokola koji se oslanjaju na *IEC 60870-5* skup standarda. Od ovih standarda za čitaoca je najbitniji *60870-5-104* protokol. Standard *IEC 104* definiše telemetrijski protokol za komunikaciju sa udaljenim telemetrijskim stanicama i procesnim kontrolerima. *IEC 60870-5-104* predstavlja kombinaciju aplikativnih funkcija *IEC 60870-5-101* protokola i funkcija podržanih *TCP/IP* protokolom.

3.3. Modbus

Modbus je industrijski komunikacioni protokol aplikativnog nivoa *ISO-OSI* modela. Primarno se koristi za klijent-server komunikaciju između *SCADA* stanice i udaljenih telemetrijskih stanica i programabilnih kontrolera. *Modbus* je najstariji industrijski protokol koji je i danas u primeni. Razvijen je 1979. godine od strane

kompanije *Modicon*, koja je danas u vlasništvu kompanije *Schneider Electric*.



Slika 2: DNP3 model udaljenih tačaka [2]

4. KONCEPT REŠENJA

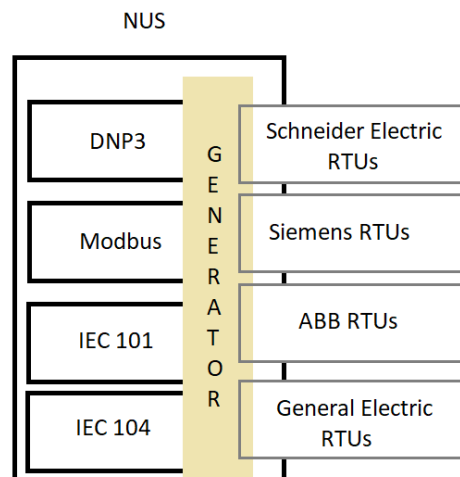
Nakon nabavke novog procesnog kontrolera prvi korak je montiranje istog u proces koji se automatizuje. Posle završenog ožičavanja kontrolera kreira se konfiguracija za uređaj pomoću specifičnih softverskih alata. Kreirana konfiguracija mora da bude sačuvana za kasnije korišćenje.

Sledeći korak jeste unos iste konfiguracije u NUS (nadzorno upravljački sistem). U najvećem broju slučajeva taj unos se vrši ručno. Kreiranje konfiguracije je često dugotrajan i mukotrpan proces, i neretko se dešavaju greške pri unosu.

Najčešće se te greške dešavaju zbog pada koncentracije SCADA inženjera prilikom prekucavanja podataka iz fajla kreiranog tokom ožičavanja i konfigurisanja procesnog kontrolera. Greške se kasnije otkriju tokom *PTP (Point to point) testiranja*. *Point to point* provere su deo *FAT (Factory Acceptance Test)* ili *SAT (Site Acceptance Test)* testiranja. Ako se tokom ovog procesa nađe na neki problem, potrebno je ispraviti konfiguraciju u nadzorno upravljačkom sistemu, a ako je problem nastao tokom konfigurisanja procesnog kontrolera onda je potrebno otići na teren i ispraviti grešku. Takve greške su skupe, ali skoro je nemoguće da se greške neće desiti, s obzirom da može biti i nekoliko miliona SCADA tačaka.

Da bi se smanjila verovatnoća greške, moglo bi da se inicijalna konfiguracija unese direktno u nadzorno upravljačku stanicu, a da se zatim generiše konfiguracija koja je specifična za procesni kontroler. Generisan fajl bi mogao da se prekopira na uređaj pre montiranja u postrojenje ili bi mogao da se prebaci na kontroler pomoću *HTTP* ili *SSH* komunikacije. Na slici 3 je

prikazana arhitektura rešenja.



Slika 3: Arhitektura rešenja

5. IMPLEMENTACIJA REŠENJA

Prvi korak u generisanju konfiguracije jeste odabir tipa i proizvođača uređaja. Nakon odabira se unose pravila za mapiranje udaljenih tačaka u NUS-u na udaljene tačke na procesnom kontroleru. Na uređaju kao što je Schneider Electric T300 konfiguracija je u *XML* formatu i svaka udaljena tačka ima jedinstveno ime. Ime je sastavljeno iz više delova:

- Prvi deo označava podmodul procesnog kontrolera,
- Drugi deo identifikuje procesni ulaz/ilaz

Pomoću te informacije je moguće uraditi mapiranje tačaka iz NUS-a na registre u uređaju. Za uređaje starijeg tipa može da se uradi mapiranje na osnovu rednog broja registra, ili neke druge konvencije u imenu tačke.

Nakon uspešnog mapiranja upisuju se gore navedeni atributi u datoteku. Naravno, to se vrši u zavisnosti od protokola uređaja. Kada se uspešno završi generisanje fajla, može da se radi sledeći korak tj. isporuka datoteke na uređaj.

Postoji nekoliko načina da se izvrši isporučivanje datoteke.

- Isporuka pomoću protokola koji podržava transfer datoteke;
- Pomoću *HTTP* protokola;
- Uspostavom *SSH* veze

Neki protokoli kao *DNP3* podržavaju slanje datoteke. Ta funkcionalnost može da se koristi ako već postoji uspostavljena veza sa procesnim kontrolerom. U tom slučaju, nakon generisanja modifikovane konfiguracije datoteka se šalje pomoću protokol specifičnih komandi, a zatim se daljinski ponovo pokreće uređaj.

Noviji kontoleri podržavaju pristup preko veb interfejsa. Ti uređaji iz fabrike izlaze sa fabričkom *IP* adresom, i na taj način može da im se pristupi pomoću *HTTP*-a.

Treći pristup jeste povezivanje sa udaljenom telemetrijskom stanicom pomoću *Secure Shell* protokola i slanje konfiguracionog fajla kroz specifične komande koje sam uređaj nudi.

6. ZAKLJUČAK

U ovom radu predstavljen je predlog *Device Management* sistema, tačnije funkcionalnosti za automatsko generisanje konfiguracije za procesne kontrolere koje koriste komunikacione protokole *Modbus*, *IEC 60870-5-101/104* ili *DNP3*.

Ideja rada jeste da čitalac razume kakvi su trenutni postupci postavljanja i konfigurisanja sistema, koje su mane tog načina i kako se mogu poboljšati.

Rad bi dalje mogao da bude proširen sa drugim već pomenutim *Device Management* funkcionalnostima kao što su:

- Procesiranje konfiguracije učitane sa procesnog kontrolera;
- Praćenje istorije konfiguracionog fajla;
- Rukovanje lozinkama;
- Prepoznavanje greške u konfiguraciji;
- Ažuriranje softvera na procesnim kontrolerima;
- Daljinska manipulacija konfiguracijom.

7. LITERATURA

- [1] Branislav Atlagić, Fakultet tehničkih nauka, Novi Sad, Softver sa kritičnim odzivom, 2015
- [2] DNP3 Specification Volume 2 Part1, Basics Application Layer
- [3] Modbus Application Protocol Specification v1.1b3
- [4] International standard IEC 870-5-1: Telecontrol equipment and system - Part 5 - Edition 2
- [5] IEC 61850-6:2009 - Configuration language for communication in electrical substations
- [6] Practical Modern SCADA Protocols: DNP3, 60870.5 and Related Systems, Gordon R. Clarke,

Kratka biografija

Čaba Šagi rođen je 1994. godine u Novom Sadu. Završio je gimnaziju „Svetozar Marković“ u Novom Sadu 2013. godine. Osnovne akademske studije na Fakultetu tehničkih nauka u Novom Sadu je upisao 2013. godine i završio 2017. godine. Iste godine je upisao master studije na smeru Primenjeno softversko inženjerstvo na Fakultetu tehničkih nauka. Ispunio je sve obaveze i položio je sve ispite predviđene studijskim programom.

**JEDNO REŠENJE SOFTVERSKE ARHITEKTURE PRIJEMNIKA AUDIO I VIDEO
SADRŽAJA ZASNOVANO NA DIGITALNOM SIGNAL PROCESORU****ONE SOLUTION OF SOFTWARE ARHITEKTURE FOR AV RECEIVER BASED ON
DIGITAL SIGNAL PROCESSOR**

Marko Antić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je predstavljena softverska arhitektura sistema koji je zadužen za upravljanje audio i video sadržajem i njihovom obradom. Sistem je baziran oko Cirrus Logic DSP CS49844 procesora za obradu audio signala i ATSAM3N0B mikrokontrolera kao glavna podrška pri radu DSP procesora. Predstavljeno rešenje je izloženo po funkcionalnim slojevima.

Abstract – This work presents microsoftware architecture of a system which is responsible for managing the system of AV receiver. The system is based on Cirrus Logic CS49844 DSP audio processor and ATSAM3N0B microcontroller which is used as main support for controlling the DSP processor. Code is presented by functional layers.

Ključne reči: DSP, mikrokontroler, obrada digitalnog signala

1. UVOD

Stalan razvoj informacionih tehnologija, kao i potrošačke elektronike se posebno video u protekloj deceniji. Sa njima su se takođe razvile industrije koje uveliko koriste dobrobiti ovih tehnologija.

Među ovim industrijama našle su se i filmska i muzička industrija koje su polako prelazile na kreiranje svojih sadržaja pomoću najatraktivnijih audio tehnologija koje se mogu naći na tržištu. To je dovelo do razvoja tzv. kućnih bioskopa čiji je zadatak bio da uspešno reprodukuju audio i video sadržaj. Sa njihovim razvojem i željom da se funkcija više različitih uređaja objedini u jedan, nastao je prijemnik audio i video sadržaja AVR (eng. *Audio-VideoReceiver*).

Prijem i obrada audio i video signala iz raznih izvora, predstavlja glavnu ulogu AVR-a. Obradeni signali se dalje prosleđuju na zvučnike i displej (TV, projektor). Najveća prednost AVR-a je da izvori signala koji dovode signal do ovog uređaja mogu biti razni drugi uređaji, poput satelitskog prijemnika, TV, radio prijemnika, DVD plejera, Blue-Ray plejera, video rekordera i sl.

Takođe, cilj sistema AVR-a je da poseduje mogućnost dekodovanja jednog ili više tipova audio formata koji se nalaze trenutno na tržištu (Dolby Digital, Dolby Digital Plus, Dolby TrueHD, DTS, DTS-ES, DTS-HD...). Danas se proizvođači takmiče da u svojoj ponudi imaju uređaje koji mogu dekodovati sve postojeće tehnologije, dok se te tehnologije svakodnevno unapređuju, nudeći veći broj izlaznih kanala, verniju reprodukciju zvuka, itd. što iziskuje sve moćnije sisteme za obradu audio signala.

Celokupni sistem je zasnovan na svojoj centralnoj jedinici koju predstavlja procesor za digitalnu obradu signala DSP (*Digital Signal Processor*), koga nadgleda zaseban sistemski mikroracunar (*host*).

DSP, kao glavna jedinica za obradu signala, predstavlja sistem sa ograničenim resursima. Upravo zbog toga, upravljanje njegovim radom, što predstavlja učitavanje programa sa perifernih jedinica i podešavanje sistema, obavlja mikrokontroler.

U ovom radu predstavljen je prototip prijemnika audio i video sadržaja, gde su glavne komponente četvorोजezgarni DSP procesor CS49844 firme Cirrus Logic i mikrokontroler ATSAM3N0B, firme Microchip. Pomenuti DSP se nalazi u uređajima poslednje generacije proizvođača kao što su Denon, Marantz, Onkyo i podržava dekodovanje svih postojećih audio tehnologija namenjenih kućnom bioskopu, pa i poslednje generacije Dolby i DTS audio tehnologije (Dolby Atmos, DTS:X). Ovaj prototip treba da poseduje na sebi integrisane sve periferne uređaje: SPDIF prijemnik i predajnik, AD i DA konvertore, HDMI prijemnik, te kao takav u potpunosti da predstavlja zaokružen sistem AV prijemnika.

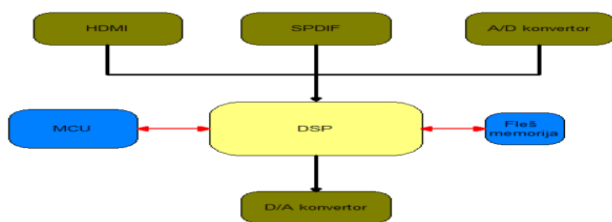
2. SISTEM AUDIO I VIDEO PRIJEMNIKA

Prijemnik audio i video sadržaja ima ulogu da primi signal na ulazu, koji može biti analogni ili digitalni, obradi ga (po potrebi dekoduje), i tako obraden signal da pretvori u analognu veličinu, koja se dalje vodi na audio pojačavač. Izlaz iz audio pojačavača predstavlja signal za pobudu zvučnika. Blok šema sistema je data na slici 1 [1].

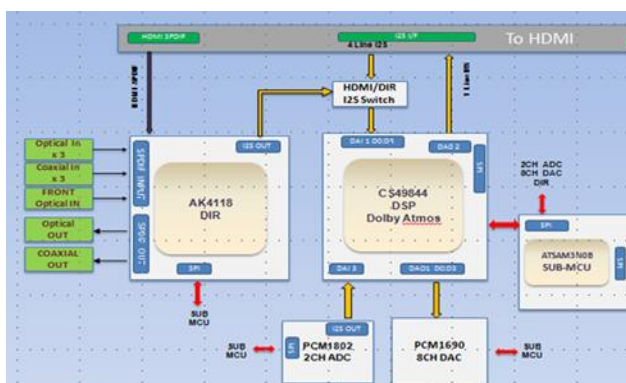
U ovom sistemu, akcenat se stavlja na digitalnu obradu signala koja je u potpunosti potisnula analognu obradu. Jedina analogna obrada signala koja se vrši u ovom sistemu je pojačavanje na izlaznom stepenu (audio pojačavač), kako bi se korisni signal mogao sprovesti na zvučnike.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Rajs, docent.



Slika 1. Uprošćena blok šema sistema prijemnika audio i video sadržaja



Slika 2. Blok šema prototipa prijemnika audio-video signala

Referentni dizajn prijemnika audio-video signala sa DSP procesorom CS49844 se dobija uz pomenuti DSP procesor. Prema referentnom dizajnu osmišljena je blok šema prototipa koja se može videti na slici 2.

Sistem je baziran na četvorojezgarnom DSP procesoru CS49844. Takođe, sastoji se i od višekanalnog audio kodeka AK4118 DIR, 16-bitnog mikrokontrolera ARM, SPI fleš memorije MX25P64, osmokanalnog DA konvertora PCM1690, stereo AD konvertora PCM1802 i HDMI primopredajnika Sil9573. U ovom sistemu DSP može da primi audio signal do 8 kanala, sa Sil 9573, AK4118 ili PCM1802 čipa, kao izvora signala.

DSP komunicira sa perifernim uređajima preko SPI ili I²C magistrala. U ovom sistemu, DSP je SPI magistralom povezan sa eksternom fleš memorijom, u kojoj je smešten tzv. *image* fajl sa celokupnim DSP firmverom (dekoderima, post-procesorima...) i svim podešavanjima potrebnim za rad DSP-a. Mikrokontroler ima zadatak da šalje komande DSP-u, u zavisnosti od sadržaja podataka koji se nalaze na audio ulazima u DSP, odnosno koje podatke treba da učita iz fleš memorije. Ovaj zadatak mikrokontroler obavlja takođe preko SPI magistrale.

2.1 Arhitektura softverske podrške mikrokontrolera

Mikrokontroler [4] ima ključnu ulogu u AVR sistemu. On je odgovoran za konfiguraciju i pravilan rad sistema (HDMI primopredajnika, SPDIF prijemnika, kodeka, ADC/DAC...), puštanje u rad DSP-a (tzv. *boot-up*), kao i upravljanjem celokupnim radom sistema, reagovanje na greške, itd. Ovo je vrlo kompleksan i slojevit zadatak.

Stoga je softverska podrška mikrokontrolera urađena kao spoj funkcionalnih slojeva.

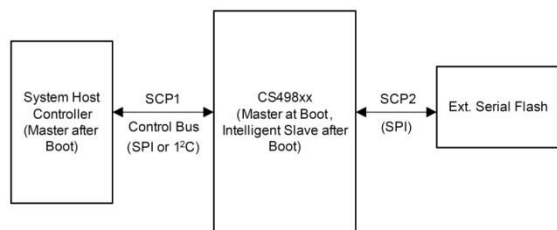
Generalno gledano, arhitektura se sastoji iz tri celine, koje se zatim dodatno dele u blokove. Prva celina je hardversko - upravljački sloj koji ima za cilj pravilno konfigurisanje periferija ovog sistema, druga je DSP aplikacija koja obavlja kontrolu samog DSP-a i treća celina je *Sil9573* aplikacija, koja predstavlja upravljanje HDMI modulom. Kod *Sil9573* aplikacije je dobijen gotov od proizvođača čipa, kompanije Silicon Image, licenciran je, te stoga nije ni razmatran u ovom radu.

Hardversko – upravljački sloj objedinjuje 2 sloja:

- **Hardverski sloj** - vrši inicijalizaciju na registarskom nivou mikrokontrolera. Ovo je firmver niskog nivoa, koji može da se automatski generiše sa alatom koji se zove Konfigurator koda (zavistan je od korišćenog mikrokontrolera) ili da se ručno konfiguriše prateći uputstva dokumentacije. Ovaj sloj inicijalizuje periferije mikrokontrolera i obezbeđuje prekidne servisne rutine niskog nivoa za periferije. Takođe ovaj sloj obezbeđuje set makroa i tzv. funkciju omotača za svaki funkcionalni blok mikrokontrolera. Oni izvlače interakcije između hardverskih registara, objedinjujući specifična pravila njihovog korišćenja.
 - **Upravljački sloj**- podrška komponentama sistema koje rade u realnom vremenu. Sadrži set modula koje implementiraju viši nivo funkcionalnosti potreban da upravlja resursima mikrokontrolera. Drajveri objedinjuju kompletnu funkcionalnost za interakciju sa integrisanim periferijama mikrokontrolera, uključujući obrađivače prekida mikrokontrolera. Takođe, ovaj sloj predstavlja apstrakciju fizičkih komponenti sistema kao funkcionalnih jedinica u softveru. Sadrži set modula koji nude servise višeg nivoa bazirane na procedurama hardverskog sloja, koje su nižeg nivoa. Ovaj sloj obezbeđuje podršku za komponente kao što su SPI komunikacija, UART komunikacija, kodek, DAC, itd.
- Sloj DSP aplikacije se može podeliti na dve celine:
- **Mašina stanja** - koja je bazirana na porukama koje stižu od strane DSP-a. Ovde se vrši inicijalizacija DSP-a, podešavanje izvora signala, primanje poruka koje stižu od DSP-a, kao i slanje potrebnih komandi DSP-u koje u suštini predstavljaju uputstva kako pravilno konfigurisati obradu signala. Ukoliko na ulazu mikrokontrolera nema dolazeće poruke od DSP-a, mikrokontroler je u stanju pripravnosti i čeka sledeću poruku.
 - **Sloj tehnologija** - ima za cilj parsiranje poruka koje odgovaraju trenutnoj tehnologiji koja se koristi za obradu ulaznog signala i slanje komandi DSP-u u zavisnosti od tehnologije primljenog signala.

2.2 Inicijalizacija DSP-a i moguća stanja DSP-a sa gledišta mikrokontrolera

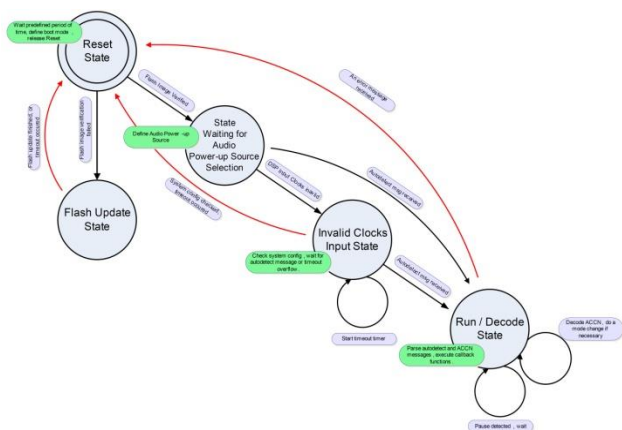
U ovom sistemu DSP radi u tzv. *master-boot* modu. Blok dijagram rada DSP-a je prikazan na slici 4. Primarna SPI magistrala je zadužena za komunikaciju sa mikrokontrolerom, a sekundarna SPI magistrala je povezana sa eksternom fleš memorijom u kojoj se nalazi svi potrebni konfiguracioni fajlovi. DSP je konfigurisan da učitava podatke sa eksterne SPI fleš memorije kada izađe iz stanja reseta. Posle inicijalizacije, u toku koje ima glavnu ulogu (*Master*), DSP se prebacuje u stanje tzv. inteligentnog podređenog uređaja (*Intelligent Slave*) [3].



Slika 4. Način rada DSP-a u master-boot modu [3]

Nakon inicijalizacije, DSP uvek šalje poruku vezanu za stanje DSP SPI fleš memorije, tj. da li su podaci u njoj ispravni. Ako je sadržaj fleš memorije oštećen, očekuje se ažuriranje. Ako je sve ispravno, nastavlja se sa podešavanjem audio izvora. DSP dalje šalje poruku o trenutnom statusu audio sadržaja (ako uopšte postoji nešto na audio ulazu u DSP), odnosno šalje poruku o statusu takta ulaznog signala ili tzv. *autodetect* i ACCN poruke (*autodetect* je poruka na osnovu koje se utvrđuje tip audio sadržaja, u skladu sa IEC61937 standardom, ACCN (*Audio Configuration Change Notification*) je obaveštenje o promeni audio konfiguracije). U zavisnosti od ulaznog audio sadržaja, DSP se prebacuje između stanja dekodovanja i stanja čekanja. *Autodetect* i opcionalna ACCN poruka se prijavljuju u stanju dekodovanja, dok se poruka za tišinu dobija kada DSP vrati u stanje za čekanje signala.

Na slici 5 je prikazan detaljan grafik stanja DSP-a.



Slika 5. Stanja DSP-a sa gledišta mikrokontrolera [3]

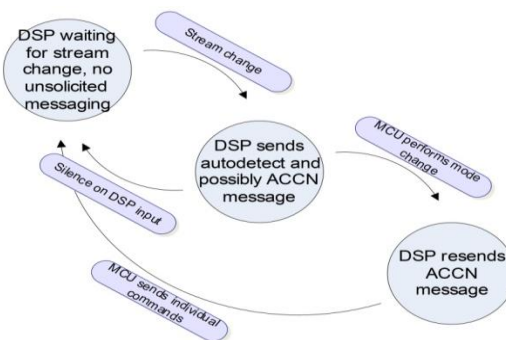
2.3 Parsiranje poruka od DSP-a

Mikrokontroler, praćenjem stanja DSP IRQ signala, zna da li postoji neka poruka poslata od strane DSP-a. DSP poruke se primaju koristeći SPI magistralu i za vreme aktivnog signala DSP IRQ (DSP IRQ je aktivan na logičkoj nuli) poruke se pamte u FIFO baferu. Odatle se obrađuju u glavnoj DSP aplikaciji.

Parsiranje DSP poruka je odrađeno preko tzv. *look-up* *tabele*. Tabela sa značenjima poruka sadrži tri kolone: Prva kolona sadrži identifikaciju tipa poruke (*Message ID*). Druga kolona sadrži očekivanu veličinu poruke (dužina je u 32-bitnim rečima). Treća kolona predstavlja poziv odgovarajuće procedure za parsiranje(*Callback procedure pcMsG_Parser<decoder>*). Posle pronalaženja rezultata po identifikacionoj oznaci, izvršava se data *Callback* procedura, koristeći parametar dužine poruke i pokazivač na memorijsku lokaciju koja sadrži podatke prikupljene iz FIFO bafera, kao parametre procedure. *Callback* procedura analizira njene ulazne parametre i odgovara na DSP poruku koristeći proceduru za upis u bafer. Procedura za raščlanjivanje poruka može uvek da proveri da li ima još pridošlih DSP poruka u FIFO baferu izvršavajući za to predviđenu proceduru. Pošto je ACCN poruka primljena i razložena, mikrokontroler šalje odgovor u dva koraka:

- izvršavaju se promene DSP moda; nakon što su poslate poruke za promenu moda DSP-a, DSP šalje obaveštenje o prijemu ovih poruka slanjem nove ACCN poruke;
- nova ACCN poruka se opet razlaže i šalju se individualne konfiguracione poruke.

Dijagram toka komunikacije pri slanju poruka između DSP-a i mikrokontrolera je prikazan na slici 6.



Slika 6. Komuniciranje između mikrokontrolera i DSP-a [2]

Za PCM signal se šalje samo tzv. *autodetect* poruka, dok se za kompresovan signal, pored *autodetect* poruke, šalje i ACCN poruka. Dok *autodetect* poruka sadrži samo informacije o tipu ulaznog audio signala (bazirano na IEC61937 označavanju audio signala), ACCN poruka sadrži specifične podatke o signalu kao i inicijalna podešavanja dekodera zaduženog za obradu tog signala. ACCN poruke su specifične za same dekodere, od broja reči, do značenja svake reči (ne postoji nikakav standard za njih).

Za svaku od dekoderskih tehnologija postoje njima namenjene procedure za raščlanjavanje ACCN poruka i konfigurisanje DSP-a, koje se nalaze u sledećim fajlovima:

- *cs498.c* - sadrži procedure za parsiranje poruka od OS modula i menadžer DSP ACCN poruka;
- *atmos.c* - sadrži procedure za podršku Dolby dekoderskih tehnologija: Dolby Digital Plus, TrueHD, Dolby Atmos...
- *ac3.c* - sadrži procedure za podršku AC3 (Dolby Digital) tehnologije
- *dts.c* - sadrži podršku za DTS tehnologije dekodera: DTS, DTS-HD...
- *pcm.c* - sadrži podršku za obradu PCM signala

Svaka dekoderska tehnologija je podržana sa procedurama za parsiranje poruka kao i sa procedurama koje su zadužene za promene moda DSP firmvera i slanje individualnih poruka DSP firmveru. Nakon što su primljene ACCN poruke isparsirane, primljeni sadržaj je analiziran i (bazirano na podešavanjima korisnika preko UART serijske komunikacije), firmver DSP-a se konfigurira. Konfigurisanje DSP-a vrši mikrokontroler tako što:

- šalje zahtev za rekonfigurisanje DSP-a koristeći logiku promene moda DSP firmvera,
- šalje individualne komande DSP-u.

Bitna stvar u konfigurisanju DSP-a predstavlja *flash.h* fajl. Ovaj fajl se dobija generisanjem DSP firmvera u za to predviđenom softveru (*Cirrus Logic DSP Condenser*). U njemu se nalaze podaci o mapiranju simbolskih imenima različitih komponenti firmvera u programskom interfejsu DSP-a (API). Takođe se nalaze podaci o simbolskim imenima različitih radnih modova komponenti firmvera. Ovaj fajl se direktno koristi u mikrokontrolerskom kodu. Ove podatke mikrokontroler koristi pri komuniciranju sa DSP-om, i zbog toga se u kodu mikrokontrolera mora koristiti pomenuti fajl, generisan za firmver koji se koristi u radu sistema (i koji se čuva na FLASH memoriji, koja je deo sistema).

2.4 Korisnički interfejs

Ponovno konfigurisanje DSP-a može biti pokrenuto i od strane korisnika. U ovom sistemu to je omogućeno slanjem komandi preko UART komunikacije. Moguće komande pokrivaju celokupnu strukturu svih firmver modula DSP-a koje su dostupne u odgovarajućem *image* fajlu. Svako slanje komandi od strane korisnika će resetovati DSP i pokrenuti ponovnu inicijalizaciju DSP procesora i uraditi rekonfiguraciju sistema, ali sada sa novim, podešenim parametrima koji su zahtevani od strane korisnika. Logika korisničkog interfejsa se nalazi u fajlu *host.c* gde se vrši parsiranje dolazećih poruka od korisnika.

3. MOGUĆA DALJA UNAPREĐENJA SOFTVERA MIKROKONTROLERA

Trenutno rešenje softverske podrške od korisnika prima instrukcije za podešavanje sistema preko UART

komunikacije i terminala na PC-u. Uvođenjem dodatnog mikrokontrolera za kontrolu korisničkog interfejsa sa upotrebom displeja i odgovarajućih dugmića, omogućilo bi znatno lakšu upotrebu i znatno lakše podešavanje.

Postojeće rešenje programske podrške, realizovano metodom super petlje, gde se svi procesi izvršavaju u beskonačnoj petlji, nije najbolje optimizovano sa strane upravljanja prioritetima izvršavanja procesa. Proces koji se izvršavaju u sistemu nisu raspodeljeni po prioritetima, već se izvršavaju redom, u predefinisanim vremenskim okvirima, ili na određene sistemske prekide. Ovo se može unaprediti implementacijom RTOS operativnog sistema, koji bi vodio računa o pravilnoj raspodeli resursa između procesa. *FreeRTOS* se zadnjih par godina izdvojio u izvesnoj meri kao industrijski standard za RTOS rešenje kod mikrokontrolera i mikroprocesora.

4. ZAKLJUČAK

U ovom radu prikazana je realizacija softverske arhitekture prijemnika audio-video sadržaja. Izložena je arhitektura softvera, podeljena po funkcionalnim slojevima. Akcenat je stavljen na upravljanje DSP-a, kao i način obrade poruka pristiglih od DSP-a. Predstavljeno rešenje je potpuno funkcionalna aplikacija realizovana na relativno pouzdanom mikrokontroleru, koji obezbeđuje uspešno kontrolisanje DSP-a poslednje generacije, kao i ostatka sistema (HDMI i SPDIF primopredajnika, DA/AD konvertora, itd.).

Nagovešteno je i moguće unapređenje realizovanog softvera implementacijom *FreeRTOS* sistema, čime bi se optimizovalo izvršavanje procesa, što bi poboljšalo robusnost sistema i otpornost na otkaz.

Takođe, primenom dodatnog mikrokontrolera, moguće je i implementirati grafički korisnički interfejs, koji bi se prikazivao na monitoru/displeju preko HDMI izlaza, što je jedna od poželjnih opcija u slučaju komercijalne realizacije audio-video prijemnika.

5. LITERATURA

- [1] V. Kovačević, M. Temerinac, M. Popović, N. Teslić, *Arhitekture i algoritmi DSP-a I*, Novi Sad, FTN, 2004.
- [2] *CRD498x4 Board User's Manual*, Cirrus Logic, 2014.
- [3] *CS498xx System Designers Guide*, Cirrus Logic, 2013.
- [4] *Atmel | SMART ARM-based MCU*, USA, Atmel Corporation, 2015.
- [5] Ivan Mezei, Veljko Malbaša, *Mikroprocesorska elektronika-predavanja*, Novi Sad, FTN, 2015.

Kratka biografija:



Marko Antić rođen je u Loznici 1990. godine. Osnovne studije je završio 2015. godine na smeru Mikroracunarska elektronika - Primenjena elektronika. Ovaj rad je proistekao iz master rada koje će biti odbranjen u 2018. godini.

PRIMENA REGRESIONIH MODELA U PREDVIĐANJU ISHODA GLASANJA NA PESMI EVROVIZIJE**APPLICATION OF REGRESSION MODELS TO PREDICT THE VOTING OUTCOME OF THE EUROVISION SONG CONTEST**

Maksim Lalić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je prikazan regresioni model namenjen predviđanju ishoda glasanja na Pesmi Evrovizije. Prikazane su i prodiskutovane vrednosti njegovih koeficijenata i performanse koje ostvaruje pri predviđanju. Dato je objašnjenje skupa podataka nad kojim je model formiran. Prikazani su zaključci deskriptivne statističke analize nad dobijenim podacima.

Ključne reči: Regresioni algoritmi, analiza i obrada podataka, deskriptivna statistika, Pjesma Evrovizije

Abstract – In this paper, a regression model for predicting voting outcome of the Eurovision Song Contest is shown. Coefficients and prediction performance of the model are presented and discussed. The dataset on which the model was formed is described. Conclusions of a descriptive statistical analysis of the dataset are provided.

Keywords: Regression algorithms, data analysis and transformation, descriptive statistics, Eurovision

1. UVOD

Pjesma Evrovizije je poznati internacionalni muzički festival takmičarskog karaktera u kom svaka država može poslati jednog predstavnika, koji će je predstavljati jednom pjesmom. Kao i mnogi drugi događaji iz industrije zabave, takmičenje poprima sve veći medijski značaj. Postoje sajtovi na kojima se pišu članci o događajima u vezi sa takmičenjem [1]. Države koje učestvuju na takmičenju organizuju nacionalna takmičenja na kojima se odlučuje koga će poslati kao svog predstavnika. Sve je više država koje učestvuju na takmičenju, pa su, osim glavne, finalne večeri, vremenom uvedene prva i druga polufinalna večer [2]. Sa razvojem kladenja na Internetu, stvorila se mogućnost ulaganja novca na ishode takmičenja [1].

Sistem glasanja je takav da, nakon izvođenja svih pjesama, svaka država poseduje određeni skup poena koji raspodeljuje između ostalih država. Najpoznatiji, i najduže praktikovan način bodovanja, od 1975. do 2015. godine, nalagao je da, u toku jedne takmičarske večeri, svaka država odabere deset država koje će oceniti redom sa 1, 2, 3, 4, 5, 6, 7, 8, 10 i 12 poena, a ostale države od posmatrane dobijaju 0 poena [3].

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Ivančević, docent

Do 1998. godine je poene koje jedna država treba da raspodeli raspodeljivao žiri [3]. Nakon toga je uvedeno pravilo da se glasanje država sprovodi na osnovu telefonskog glasanja njenih građana [3]. Takav sistem ocenjivanja se zadržao do 2008. godine bez izmena [3]. Nakon uvođenja glasanja građana telefonskim putem, počelo je da se spekuliše o postojanju šablona kako države glasaju i tzv. blokovskog glasanja, fenomena postojanja grupa država koje se redovno ocenjuju natprosečno velikim brojem poena. To se objašnjava time da glasanje građana nije u tolikoj meri pod uticajem pjesama i performansa izvođača već simpatija koje građani imaju prema samim državama za koje se glasa. Sumnja se da različiti faktori utiču na ocenjivanje između država, a najčešće spominjani faktori su geografska bliskost država i kulturnološka i genetska sličnost naroda koji žive u različitim državama, kao i demografski i ekonomski odnosi država [4].

Postoje istraživanja u kojima su, spram raznih kriterijuma, uočeni i objašnjeni određeni klasteri država koje se redovno ocenjuju sa visokim brojem poena ili su uvedeni indikatori kojima su iskazivani stepeni izražajnosti određenih šablona [5]. U drugim istraživanjima se pokušalo sa formiranjem modela za predviđanje ishoda narednih takmičenja [6]. Vršena su i ispitivanja uticaja različitih faktora na ishode glasanja [6].

U ovom radu, posmatrana su isključivo glasanja iz perioda od 1998. do 2008. godine jer je nakon toga došlo do promene u sistemu ocenjivanja, a pre toga poeni nisu deljeni na osnovu glasova publike. Cilj ovog rada je pronalaženje pogodnog modela iz klasa linearne i ordinalne logističke regresije koji predviđa ishode glasanja, prvenstveno na osnovu podataka o odnosima samih država između kojih se glasa. Na osnovu strukture tog modela može se zaključivati o povezanosti ishoda glasanja sa faktorima na osnovu kojih je model formiran.

Da bi se postigao zadati cilj, potrebno je formirati skup podataka nad kojim će regresioni modeli biti formirani i testirani. Pomoću tehnika deskriptivne statistike treba ispitati povezanost određenih faktora sa dodeljenim brojem poena na takmičenju. Potrebno je definisati kriterijume podesnosti regresionih modela i prema tim kriterijumima odabrati najpodesniji model.

Osim uvoda i zaključka, ovaj rad ima tri poglavlja. U poglavlju 2, dat je opis skupa podataka na osnovu kog su formirani modeli. Poglavlje 3 prikazuje zaključke deskriptivne statističke analize određenih šablona u glasanjima država. Postupci formiranja regresionih

modela, procesiranje predviđanja koja modeli daju, metode evaluacije modela, kao i performanse i struktura odabranog modela, sa tumačenjem vrednosti njegovih koeficijenata, dati su u poglavlju 4.

2. SKUP PODATAKA

Skup podataka na osnovu kog su formirani prediktivni modeli sadrži podatke o glasanjima realizovanim u periodu između 1998. i 2008. godine. Dobijen je brojnim transformacijama i spajanjima izvornih podataka, koji su mahom preuzeti sa Interneta [7-9]. Krajnji skup podataka ima 12435 zapisa o konkretnim glasanjima od kojih svako ima 17 obeležja. U statističkoj i regresionoj analizi je korišteno jedanaest obeležja (*Prosek_dosad*, *Frekv_dosad*, *Glasanja_dosad*, *Upoznatost*, *Tip_granice*, *Udeo_imigracije*, *Geograf_bliz*, *Jezička_bliz*, *Religijska_bliz*, *BDP_odnos* i *Poena*). Na osnovu preostalih šest pomoćnih obeležja (*Ocenjivač*, *Takmičar*, *Godina*, *Faza*, *Pesma* i *Izvođač*) moguće je lakše razumeti kontekst pojedinačnog glasanja.

Obeležje *Poena* predstavlja broj poena koji je dodeljen u konkretnom glasanju. Obeležja *Prosek_dosad*, *Frekv_dosad* i *Glasanja_dosad* redom označavaju koliko je u proseku država koja ocenjuje davala poena državi koja se ocenjuje, sa kolikom frekvencijom joj je davala preko nula poena, i koliko je takvih glasanja prethodno bilo u posmatranom periodu.

Procena odnosa ekonomskih stanja država između kojih se glasa, računata kao prirodni logaritam količnika njihovih procenjenih BDP-ova (bruto domaćih proizvoda) po glavi stanovnika, izražena je kroz obeležje *BDP_odnos*. Broj doseljenika iz države za koju se glasa u državu koja glasa podeljen ukupnim brojem doseljenika u državu koja glasa predstavlja vrednost obeležja *Udeo_imigracije*.

Vrednosti obeležja *Tip_granice* uzimaju vrednosti 0, 1, 2 i 3, koje redom odgovaraju slučajevima kada države nemaju zajedničku granicu, imaju isključivo morsku, isključivo kopnenu, ili dele i morsku i kopnenu granicu. Za određeno finalno glasanje, vrednost obeležja *Upoznatost* je jedan ukoliko je pesma države za koju se glasa izvođena i u polufinalnoj večeri, a dva, ukoliko je, osim prethodnog uslova, i pesma države koja glasa izvođena u polufinalnoj večeri. Ako posmatrano glasanje nije finalno, ili jeste ali ne postoji polufinale u toj godini, ili država za koju se glasa nije učestvovala u njemu, vrednost ovog obeležja je nula.

Svaka država je povezana sa vrednostima koje predstavljaju nazive najzastupljenijih jezika i religija unutar njene populacije, kao i istorijsko-geografskog regiona kom pripada. Skupovi mogućih vrednosti po ova tri kriterijuma su hijerarhijski klasterovani. Bliskost svake dve vrednosti određenog kriterijuma je izračunata kao broj vrednosti koje ne ulaze u najmanji klaster koji obuhvata posmatrane vrednosti, podeljen ukupnim brojem vrednosti u datom klasteru. Na osnovu toga, bliskost dve države po nekom kriterijumu je računata kao najveća bliskost vrednosti pridruženih jednoj i drugoj državi po tom kriterijumu. Obeležja *Jezička_bliz*, *Religijska_bliz* i *Geograf_bliz* predstavljaju redom bliskosti država spram jezika, religija i matičnih geografsko-istorijskih regiona.

3. STATISTIČKA ISTRAŽNA ANALIZA

Prosečan broj poena koji se razmenjuje između država je 2,52. Utvrđeno je da postoje čak 22 para država ocenjivač-takmičar gde se dodeljuje u proseku preko 11 poena, na uzorku od minimum 3 glasanja. Takođe je utvrđeno postojanje grupa država povezanih sa istim religijama i jezicima, koje prosečno međusobno razmenjuju više poena nego što je prosečan broj poena razmenjen u svim glasanjima u skupu podataka.

Ispitivanje povezanosti vrednosti određenog faktora i ishoda glasanja, izvršeno je na osnovu vrednosti Pirsonovog koeficijenata korelacije. Osim toga, kreirani su i ispitani kutijasti i linijski dijagrami koji prikazuju raspodelu i prosečan broj dodeljenih poena između država u zavisnosti od vrednosti posmatranog faktora.

Zaključci proistekli na osnovu rezultata statističke analize uglavnom sugeriraju postojanje povezanosti posmatranih faktora sa ishodima glasanja. Primećena je jaka veza između vrednosti obeležja *Prosek_dosad* i *Frekv_dosad*, sa jedne strane, i vrednosti obeležja *Poena*, sa druge strane. Dati fenomen ukazuje na sličnosti tekućih i prethodnih glasanja, odnosno na prisustvo šablona glasanja koji se ponavljaju iz godine u godinu, nezavisno od konkretnih izvođača pesama i njihovih performansa koji se menjaju na godišnjem nivou.

Pokazano je da se, sa porastom geografske, jezičke i religijske bliskosti dve države, može očekivati i da te države razmene veći broj poena između sebe. Potvrđen je fenomen patriotskog glasanja gde doseljenici u određenu državu često glasaju za državu iz koje su se doselili. Dati fenomen je utvrđen posmatranjem raspodele i prosečno dodeljenog broja poena između država u zavisnosti od vrednosti obeležja *Udeo_imigracije*. Razmatrana su samo glasanja nesusednih države gde je država koja glasa bogatija od države za koju se glasa, što su uslovi koji treba da budu zadovoljeni da bi smo sa većom pouzdanošću rekli da je migracija stanovništva iz jedne države u drugu motivisana ekonomskim, a ne nekim drugim razlozima.

Nije uočena veza između ekonomskih stanja država i poena koje države razmenjuju, što je zanimljivo jer je uočena veza između intenziteta ekonomskih migracija i glasanja među državama. Takođe, uočeno je postojanje blage veze između pojavljivanja pesme u polufinalu sa poenima koje ona dobija u finalu.

4. REGRESIONA ANALIZA

Formirano je sedam skupova obeležja koja sadrže neka od polaznih obeležja i, opciono, neka izvedena obeležja koja predstavljaju izraze interakcije (eng. *interaction term*) polaznih obeležja, indikatore prisustva vrednosti određenog polaznog obeležja u određenom intervalu (eng. *step regression*), ili stepenovane vrednosti polaznih obeležja (eng. *polynomial regression*). Nad svakim od skupova obeležja formirana su po tri modela linearne regresije, pri čemu je za procenu parametara modela kod jednog korišćena klasična metoda najmanjih kvadrata, kod drugog metoda *ridge* regresije, a kod trećeg metoda *lasso* regresije [10]. Osim modela linearne regresije, nad svakim skupom obeležja je formiran po jedan model ordinalne logističke regresije [11]. Modeli su formirani na

obučavajućem skupu koji sadrži podatke o svim glasanjima osim glasanja finalnih rundi poslednje tri posmatrane godine, koja su izdvojena za testni skup podataka.

Vrednosti predikcija dobijenih pomoću regresionih modela su obrađene tako da svaka država, tokom jedne večeri, po jednom daje 1, 2, 3, 4, 5, 6, 7, 8, 10 i 12 poena, što je ograničenje koje nameće sistem glasanja na takmičenju. Postprocesiranja predviđenih ishoda glasanja su obrađivala jedan po jedan podskup glasanja i njihovih predikcija, gde svaki podskup predstavlja glasanja jedne države u okviru jedne takmičarske večeri. Predikcije modela linearne regresije su postprocesirane tako da se nakon njih predviđa da će 12 poena dobiti država za koju je regresioni model predvideo da će dobiti najviše poena, 10 država za koju je regresioni model predvideo da će dobiti najviše bodova nakon države kojoj je dodeljeno 12 poena, i tako redom. Predikcije modela ordinalne logističke regresije su postprocesirane tako da se nakon njih predviđa da će 12 poena dobiti država koja ima najveću procenjenu verovatnoću za to, 10 poena će dobiti država koja ima najveću procenjenu verovatnoću za to uzimajući u obzir samo države kojima još nije određeno koliko će poena dobiti, i tako redom. Za sve države za koje se ne predviđa da će dobiti neki pozitivan broj poena pretpostaviće se da su dobile nula poena.

Merenje uspešnosti modela je izvršeno nakon postprocesiranja njihovih predikcija. Kao kriterijumi spram kojih je merena uspešnost formiranih modela u pogađanju ishoda pojedinačnih glasanja, korištene su srednja apsolutna greška i srednja kvadratna greška za modele linearne regresije, i Makfadenov pseudo R^2 indeks (eng. *McFadden's pseudo R^2 index*) za modele ordinalne logističke regresije [11]. Osim što je merena uspešnost pogađanja rezultata samih glasanja, merena je i udaljenost stvarne rang-liste koja je rezultat određene večeri sa rang-listom kreiranom na osnovu predviđanih ishoda glasanja posmatrane večeri. Računanje udaljenosti između dve rang-liste je vršeno na dva načina.

Funkcija *Leven_udalj* računa udaljenost dve rang-liste, koje su joj prosledene kao parametri. U funkciji se svaka država predstavi jedinstvenim karakterom, a dve rang-liste stringovima tako da redosled karaktera u stringovima oslikava redosled država na rang-listama. Povratna vrednost funkcije je Levenštajnova udaljenost (eng. *Levenshtein distance*) datih stringova [12], podeljena brojem država na dužoj rang-listi.

Funkcija *Pomer_pozic* prima tri parametra, dve rang-liste čija udaljenost predstavlja njenu povratna vrednost, i pomoćni skup država, pri čemu sve države pomoćnog skupa moraju biti na obe rang liste. U funkciji se računa suma apsolutnih vrednosti razlika u pozicijama koje svaka od država pomoćnog skupa zauzima na jednoj i drugoj rang listi. Dobijena suma se množi sa vrednošću $2/n^2$ ukoliko je n parno, odnosno sa $2/(n^2-1)$ ukoliko je n neparno, pri čemu je n broj elemenata pomoćnog skupa država.

Performanse svih modela unutar istih klasa su bile veoma bliske po svim kriterijumima. Nije postojao nijedan model koji je bio najbolji po svim kriterijumima. Odabran je po jedan model linearne i ordinalne logističke regresije koji su se pokazali među uspešnijim, svaki u svojoj klasi.

Odabrani modeli su upoređeni nakon njihovog ponovnog formiranja nad celokupnim skupom podataka. Pokazali su slične performanse po kriterijumima po kojima su oba evaluirana. Zbog svoje strukture koja je pogodnija za tumačenje, kao najpodesniji model odabran je model linearne regresije. Odabrani model je formiran klasičnom metodom najmanjih kvadrata nad skupom obeležja koji ne sadrži obeležja *Frekv_dosad* i *Glasanja_dosad*, a sadrži sve moguće izraze interakcije obeležja *BDP_odnos*, *Udeo_imigracije* i *Tip_granice*.

Za svaki koeficijent odabranog regresionog modela, u tabeli 1 mogu se videti naziv obeležja kom je koeficijent pridružen, vrednost koeficijenta, kao i p vrednost za dati koeficijent. Poslednja četiri koeficijenta u tabeli 1 su pridružena obeležjima koja predstavljaju izraze interakcije polaznih obeležja. *Ekonom_inter_h0* pretstavlja izraz interakcije obeležja *BDP_odnos*, *Tip_granice* i *Udeo_imigracije*, a *Ekonom_inter_h1*, *Ekonom_inter_h2* i *Ekonom_inter_h3* predstavljaju redom izraze interakcije drugog i trećeg, prvog i trećeg, i prvog i drugog od tri navedena obeležja.

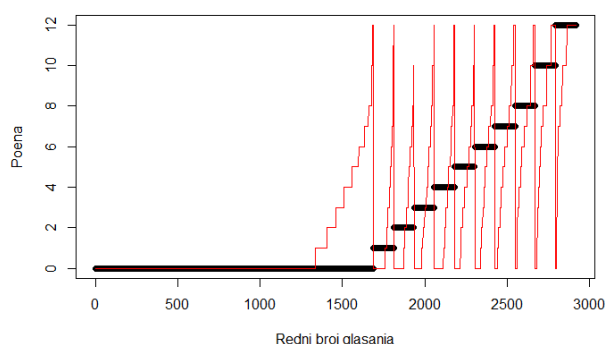
Tabela 1. Prikaz koeficijenata odabranog modela

Naziv obeležja	Vrednost koeficijenta	p vrednost
Slobodan koeficijent	2,39	<0,001
<i>Prosek_dosad</i>	1,15	<0,001
<i>Upoznatost</i>	0,17	<0,001
<i>Geograf_bliz</i>	0,26	<0,001
<i>Religijska_bliz</i>	0,04	0,153
<i>Jezička_bliz</i>	0,07	0,451
<i>Udeo_imigracije</i>	0,66	<0,001
<i>BDP_odnos</i>	0,23	<0,001
<i>Tip_granice</i>	0,32	<0,001
<i>Ekonom_inter_h1</i>	-0,12	<0,001
<i>Ekonom_inter_h2</i>	0,48	<0,001
<i>Ekonom_inter_h3</i>	0,07	<0,001
<i>Ekonom_inter_h0</i>	-0,12	<0,001

Izmerena srednja apsolutna greška odabranog modela je 1,66, što je u rang-u sa greškom koja je praviljena korišćenjem savremenijih metoda mašinskog učenja nad sličnim skupovima podataka. Izmerena srednja kvadratna greška je 8,24. Prosečna udaljenost predviđanih i stvarnih rang listi je bila 0,78 ako se udaljenost računa pomoću funkcije *Leven_udalj*, odnosno 0,23 ako se udaljenost računa pomoću funkcije *Pozic_pomer*. Dobijene udaljenosti stvarnih i prediktovanih rang-listi, su po oba kriterijuma manje od udaljenosti rang-lista finalnih večeri uzastopnih godina. Na slici 1 su, crnim tačkama, prikazane rastući sortirane stvarne vrednosti ishoda glasanja iz testnog skupa podataka, a, crvenom linijom, vrednosti koje su prognozirane nakon postprocesiranja predikcija odabranog modela. Na slici 1 se vidi da su najbolje prognozirani ishodi onih glasanja u kojima je u stvarnosti dodeljivano nula ili dvanaest poena.

Na osnovu strukture modela može se uočiti da se veliki broj poena predviđa u slučaju kada su države bez zajedničke granice, kada je država koja glasa bogatija, i u njoj je prisutno puno doseljenika iz države za koju se glasa. U takvim situacijama, može se očekivati da su doseljenici u državu koja glasa prebivalište promenili iz ekonomskih razloga. Na osnovu toga se pretpostavlja da

je fenomen patriotskog glasanja moguće prepoznati i kod odabranog linearnog regresionog modela.



Slika 1. Prikaz raspodele stvarnog i prediktovanog broja poena među državama

Za geografski bliske države, između kojih je izvesnije mešanje populacije, predviđa se bolje međusobno ocenjivanje nego za geografski udaljene države sa jezički ili religijski sličnim dominantnim narodima. Na osnovu toga, i činjenice da se patriotsko glasanje pokazalo stvarnim, kao i da u graničnim predelima između dve države ima velikog mešanja naroda, zaključuje se da natprosečno dobro glasanje među susednim državama najčešće nije uzrokovano simpatisanjima susednih naroda, već glasanjima nacionalnih manjina koje žive u okolnim državama u odnosu na svoju matičnu državu.

Na osnovu rezultata analize reziduala, zaključeno je da dobijeni modeli linearne regresije ne zadovoljavaju sve formalne pretpostavke koje se u teoriji povezuju sa linearnom regresionom analizom. Modeli ordinalne logističke regresije, ukoliko se ne izvrše postprocesiranja njihovih predikcija, imaju veliku sklonost ka predviđanju ekstremnih vrednosti, a pogotovo nule. Na osnovu toga zaključujemo da sistem glasanja na takmičenju nije najpogodniji za aproksimaciju modelima koje pretpostavljaju linearna i ordinalna logistička regresija. Ipak, nakon postprocesiranja predviđanja koja modeli daju, dobijaju se zadovoljavajuće procene ishoda glasanja.

5. ZAKLJUČAK

U ovom radu su opisani rezultati istraživanja o povezanosti, s jedne strane, određenih dugotrajnih odnosa koji postoje između država i njihovih naroda i, s druge strane, glasanjima država na Pesmi Evrovizije. Dat je opis skupa podataka nad kojim su vršena istraživanja. Opisane su primenjene tehnike deskriptivne statistike i dobijeni statistički rezultati. Dat je opis postupaka formiranja, evaluacije i obrade predikcija regresionih modela. Odabran je model linearne regresije koji se pokazao najpodesnijim za davanje predviđanja. Prikazana je i prodiskutovana njegova struktura.

Zaključci analiza ukazuju na povezanost posmatranih faktora sa rezultatima glasanja. Čini se da je fenomen patriotskog glasanja realan i da su najsigurniji glasovi koje jedna država može da dobije glasovi populacije koja živi izvan te države, a koja tu državu doživljava kao državu svog naroda.

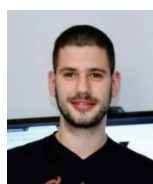
Postojeće istraživanje se može proširiti uvođenjem novih faktora koji detaljnije opisuju odnose država. Evidentiranje objektivnih karakteristika nastupa i

izvođača bi imalo velikog značaja ako se istraživanje proširi i u pravcu ispitivanja šablona u glasanjima nakon 2008. godine, jer se od tada poeni dodeljuju u zavisnosti i od mišljenja žirija i od glasova građana. Predviđanja je moguće vršiti i pomoću regresionih modela složenijeg tipa, kao i primenom neke od brojnih drugih metoda mašinskog učenja.

6. LITERATURA

- [1] Eurovision news, <https://eurovisionworld.com/>, (pristupljeno u avgustu 2018)
- [2] Why does the contest have semi finals and when they are, <https://metro.co.uk/2018/05/08/eurovision-2018-why-does-the-contest-have-semi-finals-and-when-are-they-7526735/>, (pristupljeno u avgustu 2018)
- [3] Voting systems in Eurovision history, <https://eurovisionworld.com/esc/voting-systems-in-eurovision-history>, (pristupljeno u avgustu 2018)
- [4] L. Spierdijk, and M.H. Vellekoop, "Geography, Culture, and Religion: Explaining the Bias in Eurovision Song Contest Voting", Department of Applied Mathematics, Faculty of EEMCS, University of Twente, The Netherlands, 2006.
- [5] D. Garcia, and D. Tanase, "Measuring cultural dynamics through the Eurovision song contest", Advances in Complex Systems vol. 16, no. 8, World Scientific Publishing Company, 2013.
- [6] V. Ginsburgh, and A. Noury, "The Eurovision Song Contest. Is voting political or cultural?", European Journal of Political Economy, vol. 24, pp. 41-52, 2008.
- [7] Eurovision Song Contest Database, <http://eschome.net>, (pristupljeno u avgustu 2018)
- [8] Foreign population (non-citizens) by country of citizenship, age and sex, UN Data, <http://data.un.org/Data.aspx?d=POP&f=tableCode%3A127>, (pristupljeno u avgustu 2018)
- [9] Nation Master, GDP per capita <http://www.nationmaster.com/country-info/stats/Economy/GDP/Per-capita>, (pristupljeno u avgustu 2018)
- [10] T. Hastie, R. Tibshirani, and J. Friedman, "The Elements of Statistical Learning", Springer, 2009., ISBN: 978-0-387-84857-0
- [11] A. Agresti, "Categorical Data Analysis", John Wiley & Sons, 2002, ISBN: 0-471-36093-7
- [12] Levenshtein Distance, in Three Flavors, <https://people.cs.pitt.edu/~kirk/cs1501/Pruhs/Spring2006/assignments/editdistance/Levenshtein%20Distance.htm>, (pristupljeno u avgustu 2018)

Kratka biografija:



Maksim Lalić rođen je u Novom Sadu 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Računarstvo i automatika, odbranio je 2018. god.

kontakt: lamaksim@uns.ac.rs

NAMENSKI JEZIK I OKRUŽENJE ZA MODELOVANJE I GENERISANJE VIDEO IGARA SA NASUMIČNO GENERISANIM NIVOIMA**A DOMAIN SPECIFIC LANGUAGE FOR MODELING AND GENERATION OF VIDEO GAMES WITH RANDOM GENERATED LEVELS**

Dušan Radisavljević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu opisan je razvijeni softverski alat koji omogućava specificiranje i generisanje video igara sa nasumično generisanim nivoima i elementima igara koje pripadaju takozvanom roguelike žanru. Sistem se sastoji iz namenskog jezika i dva seta Model-To-Text transformacija. Upotrebom opisanog sistema moguće je generisati dvodimenzionalne video igre u Java2D ili generisati skript datoteke za Unity alat.

Ključne reči: Namenski jezici, razvoj softvera vođen modelima, video igre, roguelike

Abstract – In this paper we present a system for specification and generation of video games with randomly generated levels and elements of roguelike video game genre. The system comprises of domain specific language and two sets of model-to-text transformations. Through usage of the presented system it is possible to generate playable two dimensional games in Java2D or generate scripts for the Unity game engine.

Keywords: Domain-Specific Languages, Model-Driven Software Development, video games, roguelike

1. UVOD

Industrija video igara trenutno pretstavlja jednu od najprofitaibilnijih grana industrije. Vrednost industrije se meri u milijardama dolara u Sjedinjenim Državama, dok se predviđa da će do 2020. godine ona dostići čak 20 milijardi [1].

Razlog je taj što se igre danas ne posmatraju samo kao razonoda, već sredstvo preko kojeg velike kompanije mogu da reklamiraju i promovišu svoje proizvode. Pored toga, danas su sve popularniji takozvani e-sportovi (eng. *eSports*), turniri u video igrama koji se organizuju u velikim arenama i prenose na televiziji, a na koje je moguće kladiti se.

1.1. Motivacija

Iako industrija video igara postoji već skoro četrdeset godina, danas još uvek ne postoji namenski jezik za specifikaciju mehanike igre. Pravila igre, koja predstavljaju glavni alat kada je dizajniranje igre u pitanju, specificiraju se upotrebom prirodnog jezika.

Ovo može dovesti do jaza između dizajnera i programera, zbog činjenice da pravila specificirana na ovaj način mogu biti nedovoljno jasno objašnjena. Zbog ovoga, na novou implementacije, gde je ispravka grešaka skupa, sam programski kôd postaje specifikacija igre, što može dovesti do igara lošijeg kvaliteta sa tehničke strane.

Pored samih pravila igre, postoje i drugi aspekti koji su međusobno slični od igre do igre, a čije kreiranje bi moglo biti dodatno olakšano upotrebom jezika specijalizovanog za njihovu specifikaciju. Neki od njih su animacije, izgled korisničkog interfejsa i dizajn nivoa, dok se za specifikaciju fizike, kolizije i kretanje tela unutar igre uspešno koriste *Game Engine* komponente.

1.2. Cilj i zadaci

Jedan od glavnih zadataka biće nasumično generisanje nivoa unutar dvodimenzionalne video igre na osnovu specifikacije, čime bi se igra učinila interesantnijom kroz više prelaza. Ovaj pristup inspirisan je takozvanim roguelike video igrama, koje su danas popularne jer se nivoi ne mogu naučiti napamet, čime zahtevaju dodatnu veštinu od igrača. *Roguelike* se odnosi na podžanr video igara koje karakteriše navigiranje kroz veliki broj proceduralno generisanih lavirinata, potezno bazirano kretanja i grafika koja podseća na starije igre [2].

Roguelike video igre su svoj naziv dobile po igri *Rogue* koja se pojavila 1980. godine. *Rogue* predstavlja igru koja se pokreće u terminalu i igra preko tekstualnog unosa. Iako nije prva igra koja je imala elemente koji danas karakterišu roguelike video igre, ova igra je jedna od prvih popularnih naslova. Danas, iako broj roguelike naslova u punom smislu nije mnogo, sve je više onih igara koje obuhvataju bar neke elemente ovog podžanra. To dovodi do toga da se ove igre nazivaju *roguelike-like* ili *rogue-lite*. Neki od popularnijih naslova koji spadaju u *rogue-lite* kategoriju su *The Binding of Isaac*, *FTL: Faster Than Light*, *Spelunky* i *Diablo*.

Cilj ovog rada jeste specifikacija i generisanje video igre upotrebom platformski nezavisnih modela. Upotrebom ovog tehnološki nezavisnog pristupa koji bi zamenio prirodni jezik, olakšao bi se posao dizajniranja video igara i komunikacija dizajnera sa programerima. Upotrebom pristupa razvoju softvera zasnovanom na modelima i upotrebom namenskih jezika (*Domain-Specific Language*) omogućuje se lakše rešavanje domenskih problema i skraćuje se vreme neophodno za razvoja softvera.

Za potrebe ostvarenja ovog cilja, neophodno je razviti namenski jezik i skup transformacija koji će koncepte iz namenskog jezika prebaciti u neki od jezika opšte namene

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Ivan Luković, red. prof.

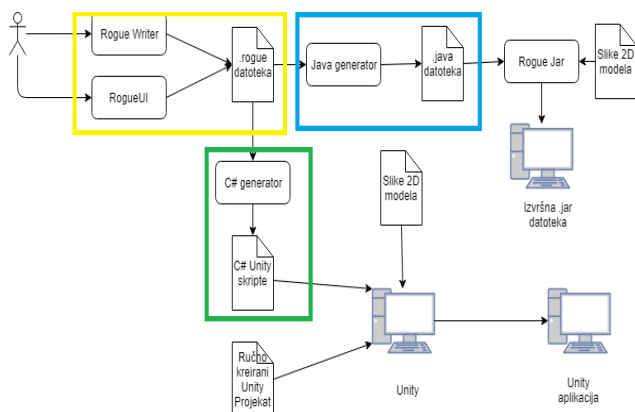
preko kojeg će se igra izvršavati. Takođe je neophodno razviti niz pomoćnih alata koji mogu doprineti daljoj automatizaciji procesa kreiranja video igara.

1.3. Struktura rada

Osim uvoda i zaključka, ovaj rad sadrži još tri poglavlja. Poglavlje 2, pod nazivom *Arhitektura alata*, predstavlja opis delova razvijenog alata. U poglavlju 3 opisana su apstraktna i konkretna sintaksa, dok poglavlje 4 daje primer izgenerisanih video igara.

2. ARHITEKTURA ALATA

U ovom poglavlju biće ukratko opisani delovi koji čine razvijeno softversko rešenje kao i njihova uloga u procesu generisanja video igre. Na slici 1 prikazana je arhitektura softverskog rešenja, kao i tok aktivnosti koji dovodi do samog generisanja video igre.



Slika 1. Proces generisanja igre upotrebom rešenja i softverskih alata opisanih u ovom radu

Prvi korak (uokviren žutim kvadratom) predstavlja specifikacija video igre upotrebom namenskog jezika *RogueDSL*, o kojem će biti više reči u kasnijem poglavlju. Korisnik ovo može uraditi na dva načina, ispisujući .rogue datoteku u tekstualnom editoru ili upotrebom *RogueUI* modula za generisanje .rogue datoteke na osnovu ispunjenih formi.

Namenski jezik *RogueDSL* je razvijen u okviru razvojnog okruženja *Eclipse*. Napravljeni su i alati za sintaksnu i semantičku proveru modela, kao i mogućnost automatske dopune modela kreiranih pomoću namenskog jezika. Meta-model koji predstavlja apstraktnu sintaksu jezika *RogueDSL* implementiran je u okruženju EMF (*Eclipse Modeling Framework*). Konkretna sintaksa jezika kao i prateći alati razvijeni su u okruženju *Xtext*. Korisnik može upotrebom konkretne sintakse specificirati model tekstualno. Alternativan pristup je upotreba *RogueUI* modula. *RogueUI* modul predstavlja skup formi napisanih u *Java Swing* biblioteci, čijim ispunjavanjem se generiše .rogue datoteka.

Za jedan namenski jezik moguće je razviti više različitih generatora kôda. U ovom trenutku razvijena su dva: za programske jezike *Java* i *C#* (plavi i zeleni pravougaonik, redom). Kôd za pomenute programske jezike generatori proizvode na osnovu .rogue datoteke. Generator *Java* kôda vrši generisanje skupa klasa koje zajedno formiraju igru u dvodimenzionalnom prostoru napisanu u *Java* programskom jeziku. Generator *C#*-a vrši generisanje .cs

datoteka koje mogu biti korištene kao script datoteka u *Unity* softverskom alatu.

Nakon generisanja kôda za programske jezike, .java datoteka se dalje može koristiti sa *Rogue Jar* modulom. Neophodno je smestiti datoteku u folder koji sadrži slike koje će biti korištene unutar igre (na primer, .png datoteka koja sadrži crtež koji predstavlja glavnog junaka igre, neprijatelje ili predmete koje je moguće pokupiti). *Rogue Jar* nakon pokretanja na osnovu slika i programskog kôda vrši generisanje izvršne .jar datoteke, pokretanjem koje se otvara igra.

Kada su generisane datoteke u *C#* programskom jeziku, prvo je neophodno ručno kreirati novi projekat unutar softverskog alata *Unity*, uključiti datoteke u projekat i onda, koristeći korisnički interfejs softverskog alata *Unity*, dodati ih kao *script* datoteke na neophodna mesta.

3. NAMENSKI JEZIK ZA GENERISANJE VIDEO IGARA

Razvoj namenskog jezika podrazumeva prethodni razvoj apstraktne i konkretne sintakse. Apstraktna sintaksa se razvija modelovanjem koncepata iz domena pomoću koncepata meta-meta-modela. Za razvoj apstraktne sintakse jezika *RogueDSL* korišten je radni okvir *EMF* unutar *Eclipse* alata i *Ecore* jezik. Koncepti apstraktne sintakse jezika su:

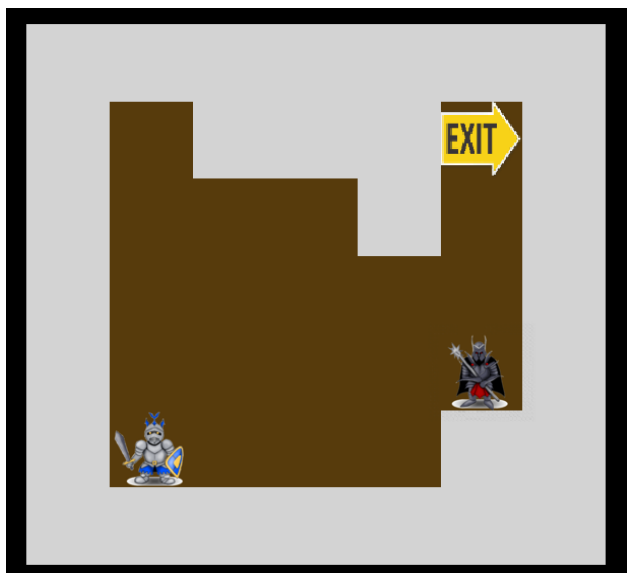
- Igra – odgovara igri koja se kreira i predstavlja korenski element,
- Heroj – odgovara karakteru kojeg igrač kontroliše unutar igre,
- Priča – odgovara pozadinskoj priči igri,
- Nivo – odgovara skupu soba kroz koje se igrač kreće, a koje odgovaraju jednoj logičkoj celini unutar igre,
- Soba – odgovara prostoru unutar kojeg se korisnik kreće,
- Karakteri koje igrač ne kontroliše – odgovara karakterima koji ne napadaju igrača, ali igrač pri tome nema kontrolu nad njima,
- Inventar – odgovara prostoru u koji igrač može da skladišti predmete koje koristi unutar igre,
- Raspored Opreme – odgovara prostoru u koji igrač može da smesti opremu koju će heroj nositi na sebi,
- Prostorna jedinica za smeštanje opreme – odgovara prostor u koji je moguće smestiti jedan predmet opreme, a koji će heroj zatim nositi,
- Zadatak – odgovara zadatku koji heroj treba da ispuni da bi dobio nagradu ili prešao nivo,
- Dijalog – odgovara tekstu koji će se voditi između heroja i nekog drugog karaktera,
- Dijalog opcija – odgovara jednom od tekstova koje je moguće izabrati kao odgovor na dijalog vođen sa nekim od karaktera,
- Podloga – odgovara podlozi po kojoj se karakteri kreću, a koja sačinjava sobu,
- Boja – odgovara nekoj od mogućih boja podloge,
- Neprijatelj – odgovara karakteru koji može naneti štetu heroju,
- Događaj – odgovara nekom od ključnih događaja u igri koji dovode do završenja zadataka ili izmene u sobi,

- [illegible]

4. PRIMER GENERISANIH IGARA

U ovom kratkom poglavlju biće prikazan izgled generisanih video igara. Na slici 3 prikazano je kako izgleda igra kada se pokrene kao Java program, dok se na slici 4 prikazuje igra koja je pokrenuta upotrebom Unity alata.

Pored načina na koji se piše sama datoteka, treba voditi računa da se zadovolji niz ostalih ograničenja koje su definisane preko *OCL*-a (*Object Constraint Language*) i *Xtend*-a. Na primer, nivo ne sme imati naziv koji već postoji za neki drugi nivo, niti bilo koji naziv koncepta ne sme biti prazan. Već je pomenuto u ranijem potpoglavlju da za heksadecimalni kôd boje mora biti tačno sedam cifri, gde prva mora biti # a ostalih 6 predstavljaju samu vrednost boje. Još neka od ograničenja uključuju to da se jedna ključna reč za atribut unutar koncepta ne može navesti više puta, da svi celi brojevi moraju biti veći ili jednaki nuli, da soba ne sme imati više od jednog neprijatelja sa *isBoss* atributom postavljenim na *Yes*. Svaka otvorena zagrada unutar .rogu datoteke mora imati odgovarajuću zatvorenu zagradu.



Slika 4. Igra pokrenuta preko Unity alata

Iako izgenerisane igre liče jedna na drugu, mogu se primetiti razlike u izgledu mape, a postoje i razlike u tome kako se igra izvodi. Pre svega, da bi se sprečio izlazak heroja van granica mape, *Unity* igra zahteva zidove oko same mape. Ovo doprinosi tome da mapa deluje manje nego ona realizovana kao *Java* program i da prostor za kretanje deluje manji. Druga razlika je u tome što *Unity* zahteva sprajtove posebnih dimenzija, dok se u *Java* kôdu slika automatski prilagođava veličini mape. Još jedna bitna razlika jeste to što se protivnici u *Unity* programu kreću ka heroju, dodajući na težini same igre.

5. ZAKLJUČAK

Zbog rasprostranjenosti i sve veće konkurentnosti na tržištu, pojavljuje se potreba za brzim razvojem video igara. Primenom pristupa razvoju softvera koji se oslanja na modele (eng. *Model Driven Engineering-MDE*), vreme razvoja video igara može se prekratiti i omogućuje se dizajnerima da jasnije specificiraju pravila igre, bez programiranja.

Namenski jezik *RogueDSL*, koji je opisan u ovome radu, nudi mogućnost specificiranja i generisanja video igara sa nasumično generisanim nivoima. Podrška za veliki broj različitih koncepata je obezbeđena da bi se mogao specificirati i pokriti što veći broj aspekata igre. Kako je konkretna sintaksa namenskog jezika tekstualna, obezbeđeno je i razvojno okruženje unutar kojeg je moguće pisati programski kod, a preko kojeg se vrši provera validnosti sintakse jezika. Dodatno, da bi jezik bio dostupan većem borju korisnika, omogućeno je i pisanje preko niza formi ispunjavanjem kojih se generiše datoteka u odgovarajućem formatu. Sam namenski jezik je implementiran na takav način da je model razdvojen od konkretne sintakse, čime se omogućuje mogućnost različite reprezentacije istog modela.

Kroz razvoj namenskog jezika ustanovljeno je da postoje kako prednosti, tako i mane ovoga pristupa. Iako namenski jezik olakšava proces generisanja video igre, njegova izrada zahteva dosta vremena. Pravi pristup bi bio razvoj više različitih namenskih jezika, gde bi se jedan

bavio specificiranjem dizajna igre, drugi animacijom a treći veštačkom inteligencijom protivnika. Ova tri namenska jezika, ukombinovani sa nekim postojećim *Game Engine*-om mogli bi dosta olakšati proces razvoja video igara. Ovime bi se obezbedila pokrivenost svakog aspekta ovog problema softverskim alatom koji je dizajniran za rešenje problema iz domena.

Rešenje razvijeno kroz ovaj rad u svome trenutnom stanju nije spremno da se koristi kao gotov proizvod za zadatke izrade složenih video igara, već samo za neke jednostavnije. Ono ipak može poslužiti kao dokaz da je ovaj pristupu razvoju igara moguć i uz nadogradnju bi moglo biti korišteno kako za složene dvodimenzionalne igre, tako potencijalno i za igre u trodimenzionalnom prostoru. U budućnosti, ima potrebe razmotriti mogućnost kreiranja alata koji bi na osnovu izvornog koda sličnih video igara dao model adekvatnog namenskog jezika preko kojeg bi one mogle biti specificirane.

Pored primene drugačijih pristupa slučajno generisanim igrama, a koji su spomenuti u drugom poglavlju, poboljšanje rešenja opisanog u ovom radu moglo bi se ostvariti tako što bi se ono kombinovalo sa nekim drugim namenskim jezicima. Neki od jezika koji bi se mogli razviti i kombinovati sa *RogueDSL* namenskim jezikom su:

- jezik za specificiranje i generisanje grafičkog korisničkog interfejsa. Ovim jezikom bi bilo moguće igraču predstaviti informacije bitne za igru, kao što su životi, predmeti u inventaru ili njegov nivo i
- jezik za kreiranje i generisanje fajlova za animaciju. Ovakav jezik bi pogotovo bilo korisno za *Unity* alat, koji pruža posebnu podršku za koncept animacije.

S obzirom na kompleksnost nekih igri, pogotovo smeštenih u trodimenzionalni prostor, postoji još nekoliko aspekata koji bi se mogli automatizovati primenom inženjeringa baziranog na modelima.

Na kraju, iako ovaj pristup u početku predstavlja kompleksniji zadatak od *ad hoc* razvoja igri, prednosti razvoja igara na ovaj način leže u tome što je specifikacija jasnija, a izmena određenih parametara olakšana. Ovaj pristup bi dodatno umanjio troškove razvoja samih video igara i tim7 omogućio manjim, ne toliko novčano imućnim, razvojnim timovima da svoje ideje uspešno plasiraju na tržište.

6. LITERATURA

- [1] <https://www.statista.com/statistics/246892/value-of-the-video-game-market-in-the-us/> (pristup.: jul 2018.)
- [2] <http://www.roguebasin.com> (pristupljeno: jul 2018.)

Kratka biografija:

Dušan Radisavljević rođen je 18. Septembra 1994. godine u Rumi. Školske 2013/2014. Godine upisuje Fakultet tehničkih nauka u Novom Sadu, studentski program Računarstvo i automatika. Osnovne akademske studije završio je 2017. godine.

**AGREGACIJA VREMENSKIH SERIJA U OBLASTI ELEKTROENERGETSKIH
SISTEMA NA RAZLIČITIM TIPOVIMA OLAP SERVERA****TIME SERIES AGGREGATION IN POWER SYSTEMS BASED ON VARIOUS OLAP
SERVER TYPES**

Andrijana Stefanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je primenjen metod agregacije vremenskih serija u oblasti elektroenergetskih sistema. Potom su predstavljeni različiti načini skladištenja agregiranih vrednosti, od kojih su neki primenjeni na uskladištene vremenske serije i njihove agregacije. Definisane su odgovarajuće metrike i izvršena je uporedna analiza performansi dobijenih skladišta u cilju odabira pogodnog tipa servera za primenu u okviru elektroenergetskih sistema.

Ključne reči: Vremenske serije, Agregacija podataka, Elektroenergetski sistemi, Hiperkocka, OLAP.

Abstract – In this paper we present an application of a method for time series aggregation in power systems. Different storage methods for aggregated values were presented and some of them were applied on stored time series and their aggregates. Appropriate measures were defined and comparative analysis of applied storage methods was conducted in order to determine the most appropriate OLAP server type for use in power systems.

Keywords: Time Series, Aggregation, Power System, Data Cube, OLAP

1. UVOD

Sistemi za podršku odlučivanju koji se oslanjaju na skladište podataka (engl. *Data Warehouse*) predstavljaju vredne alate organizacijama jer omogućavaju generisanje znanja, neophodnog za upravljanje i odlučivanje [1].

Moderni sistemi za nadzor, upravljanje i praćenje (engl. *Supervisory Control and Data Acquisition, SCADA*) našli su primenu u mnogim granama industrije. SCADA sistemi prikupljaju velik broj rezultata merenja u obliku vremenskih serija sa merača u kritičnim infrastrukturama, poput infrastruktura za prenos i distribuciju električne struje. Takve infrastrukture su kompleksne i podrazumevaju „pametne“ sisteme koji rade u realnom vremenu i kontinuirano prikupljaju merenja sa merača. S obzirom na broj prikupljenih rezultata merenja, njihova individualna analiza je retko moguća.

Jedan od mogućih pristupa prevazilaženju ovog problema podrazumeva upotrebu *business intelligence* rešenja koje je sposobno za agregaciju vremenskih serija.

Na taj način, omogućava se pristup i analiza znatno manje količine podataka.

Sistemi za nadzor i upravljanje elektrodistributivnom mrežom su veliki, distribuirani sistemi u kojima broj merača može da pređe i nekoliko miliona, te prikupljaju znatnu količinu podataka u obliku vremenskih serija. Vremenska agregacija predstavlja teži problem od konvencionalne agregacije jer za svaku vrednost vremenske serije postoji vremenski period u toku kog je validna. Stoga je neophodno obezbediti algoritam koji vrši efikasnu vremensku agregaciju velike količine podataka. Pored toga, potrebno je obezbediti adekvatan način skladištenja agregiranih vrednosti, koji dozvoljava veliku brzinu izvršenja upita. Karakteristični upiti u posmatranom sistemu uključuju izdvajanje agregiranih podataka o velikom broju vremenskih serija, nastalih u toku određenog perioda vremena. Sistemi skladišta podataka mogu uključivati različite tipove servera u svojoj arhitekturi, te mogu skladištiti podatke u različitim strukturama, koje mogu imati različit uticaj na performanse. Potrebno je odabrati tip servera čije karakteristike ga čine najpogodnijim za skladištenje agregiranih vrednosti vremenskih serija.

Autori rada [2], kroz svoje istraživanje, razvijaju sistem za podršku odlučivanju koji je zasnovan na skladištu podataka i prilagođen preduzećima manjeg i srednjeg obima. U radu su opisani razni pristupi, dat je pregled osobenosti tipova OLAP (engl. *On-Line Analytical Processing*) servera i predlog adekvatnog tipa, najbolje prilagođenog potrebama razmatranih preduzeća.

Glavni cilj ovog rada je odabir pogodnog tipa OLAP servera za primenu u okviru sistema skladišta podataka u elektroenergetskom sistemu. Prvi zadatak je upoznavanje sa osnovama agregacije vremenskih serija sa ciljem njihove primene nad rezultatima merenja u okviru elektroenergetskih sistema. Zatim, potrebno je generisati skup podataka koji opisuje dovoljno velik broj vremenskih serija, kako bi se simuliralo visoko opterećenje sistema. Sledi projektovanje šeme operativne baze podataka i baze podataka skladišta podataka. Zatim, definisanje metrika za merenje performansi servera u skladu sa potrebama posmatranog sistema. Poslednji zadatak je vršenje uporedne analize tipova servera u skladu sa definisanim metrikama i odabir pogodnog tipa na osnovu dobijenih rezultata.

Pored uvoda i zaključka, rad sadrži šest poglavlja. U poglavlju 2, diskutuje se pojam vremenske serije i osnovni principi vremenske agregacije i daje opis skupa

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Ivan Luković, red.prof.

podataka. U poglavlju 3, dat je opis šeme operativne baze podataka. U poglavlju 4, definisani su pojmovi skladišta podataka i hiperkocke i opisani različiti tipovi OLAP servera i zvezdasta šema baze podataka. U poglavlju 5, opisani su ECTL (eng. ECTL – Extraction, Cleaning and Transformation, and Loading) zadaci za različite tipove servera. U poglavlju 6, definisane su metrike po kojima se mere performanse servera. Opisan je tok izvršenja ECTL procesa i karakterističnih upita na različitim tipovima OLAP servera i izvršeno poređenje dobijenih rezultata. U poglavlju 7, izvršena je analiza rezultata i opisane su prednosti i mane OLAP servera i načini prevazilaženja određenih nedostataka. Konačno, predložen je najpogodniji tip OLAP servera za posmatrani sistem.

2. VREMENSKE SERIJE

Vremenska serija se može definisati kao uređeni niz opservacija, pri čemu se uređivanje vrši s obzirom na vreme. U upitnim jezicima, agregacija je proces računanja jedne vrednosti koja sumiruje skup vrednosti atributa upotrebom agregacione funkcije. Agregaciona funkcija prima skup torki i rezultuje vrednošću koja sumiruje informacije sadržane u ulaznom skupu. Vremenska agregacija predstavlja proces zasnovan na vremenskom grupisanju, gde se vremenska linija particionira, torke u particijama se grupišu i agregirane vrednosti se računaju za svaku grupu. Postoje dve vrste vremenskog grupisanja – po opsegu (engl. *span grouping*) i po trenutku (engl. *instant grouping*). Grupisanje po opsegu se zasniva na predefinisanim periodima vremena, dok se grupisanje po trenutku zasniva na pronalaženju torki čiji intervali važenja sadrže zadati vremenski trenutak. Agregacije zasnovane na prethodnim tipovima grupisanja nazivaju se vremenske agregacije po opsegu (engl. *Span Temporal Aggregation, STA*) i trenutku (engl. *Instant Temporal Aggregation, ITA*), respektivno [3].

Svaka vremenska serija koju prikuplja SCADA potiče sa različitog senzora, koji se naziva tačka. Vremenske agregacije se računaju za individualne tačke, kako bi se svaka tačka mogla analizirati zasebno, pri čemu je cilj smanjenje količine podataka koja je predmet analize. Vremenska agregacija po trenutku nije pogodan izbor jer, kada se primeni na individualne tačke, rezultuje istom količinom podataka [4]. Stoga, odabrana je vremenska agregacija po opsegu.

Ukoliko se vremenska serija posmatra kao funkcija promene vrednosti u vremenu, prosečna vrednost vremenske serije za određeni period vremena se može računati kao prosečna vrednost funkcije. Prosečna vrednost funkcije $f(x)$ nad intervalom $[a, b]$ je data izrazom (1):

$$f_{avg} = \frac{1}{b-a} \int_a^b f(x) dx \quad (1)$$

Skup podataka koji je korišćen generisan je sa ciljem simuliranja scenarija visokog opterećenja elektro-energetskog sistema. Sadrži podatke o milion vremenskih serija koje se ažuriraju hiljadu puta dnevno u toku 5 dana. Broj merenja je ravnomerno raspoređen po danima i satima. Skladištenje generisanih podataka u odgovarajuću tabelu rezultovalo je tabelom veličine 180 GB.

3. OPERATIVNA BAZA PODATAKA

Namena operativne baze podataka je skladištenje atomičkih podataka o vremenskim serijama. U tabeli „Tačka“ („Point“) nalaze se podaci o svim meračima u sistemu. U tabeli „TS“ su uskladišteni atomički podaci o vremenskim serijama, koji uključuju identifikator merača sa kog je vremenska serija potekla, vremenski trenutak merenja i vrednost vremenske serije.

4. SKLADIŠTE PODATAKA

Skladište podataka podržava neposrednu analitičku obradu podataka (OLAP) čiji su zahtevi bitno drugačiji od zahteva neposredne transakcione obrade podataka (engl. *On-Line Transaction Processing, OLTP*) koja je podržana od strane operativnih baza podataka. U cilju olakšanja kompleksne analize i vizualizacije, podaci u skladištu se često modeluju multidimenzionalno. Većina skladišta podataka koriste zvezdastu šemu baze podataka kako bi predstavili multidimenzionalni model podataka.

Zvezdasta šema sastoji se od jedne činjenične tabele i više dimenzionih tabela. Dimenzija „Tačka“ („DIM_POINT“) predstavlja merač sa kojeg potiče vrednost. Vremenska dimenzija je u šemi baze podataka iskazana šemom relacije „DIM_TIME“. Vremenska hijerarhija modelovana je kroz kompletno denormalizovanu strukturu, što znači da se u istoj šemi relacije nalaze svi hijerarhijski nivoi: „Sat“, „Dan“ i „Mesec“. Hijerarhija vremenske dimenzije definisana je funkcionalnim zavisnostima: „Sat“ funkcionalno određuje „Dan“, „Dan“ funkcionalno određuje „Mesec“. Činjenična tabela, pored primarnih ključeva dimenzija, sadrži i attribute mera, koji se odnose na minimalnu, maksimalnu i prosečnu vrednost vremenske serije u toku jednog sata.

Hiperkocka je kocka u n -dimenzionalnom prostoru. Svaka dimenzija podataka pridružuje se jednoj hiperivici, a mere činjenica se beleže kao vrednosti u hiperprostoru. Mere činjenica se agregiraju s obzirom na dimenzije.

Skladišta podataka mogu biti implementirana kroz standardne ili proširene relacione sisteme za upravljanje bazama podataka, koji se nazivaju „Relacioni OLAP serveri“ (ROLAP). Takvi serveri pretpostavljaju da su podaci uskladišteni u relacionoj bazi podataka i podržavaju proširenja jezika SQL i posebne metode pristupa kako bi efikasno podržali multidimenzionalni model podataka i operacije nad njim. Multidimenzionalni model i operacije nad njim preslikavaju se na relacije i SQL upite. Predstavljaju međusloj između relacione baze podataka i OLAP klijenta. Transformišu pogled na strukturu zvezdaste šeme u pogled na strukturu hiperkocke. Multidimenzionalni OLAP serveri (MOLAP) skladište podatke u specijalnim strukturama podataka i implementiraju OLAP operacije nad njima. Pružaju direktnu podršku multidimenzionalnih struktura tipa hiperkocke. Hiperkocka se implementira putem višedimenzionalnih nizova. MOLAP serveri obavljaju predizračunavanje kako bi ubrzali obradu upita. To rezultuje dužim vremenom punjenja i često zahteva značajnu količinu memorijskog prostora. Hibridni OLAP (HOLAP) serveri predstavljaju pokušaj da se prevaziđe jaz između MOLAP i ROLAP tehnologija, obezbeđenjem

pristupa i relacionoj bazi podataka i višedimenzionalnim nizovima. Dele upite na više podupita, izvršavaju ih nad odgovarajućim delovima podataka i kombinuju rezultate [1].

5. ECTL PROCES

ECTL proces je implementiran uz oslonac na softversko rešenje za podršku zadatka izdvajanja, prečišćavanja, transformacije i punjenja podataka, *SQL Server Integration Services (SSIS)*. Koraci *ECTL* procesa variraju u zavisnosti od odabranog tipa servera. Prvi korak, bez obzira na tip servera, je punjenje dimenzionih tabela. Podaci o tačkama se izdvajaju iz izvorne tabele u okviru operativne baze podataka („*Point*“) i njima se puni tabela dimenzije tačke. U tabelu vremenske dimenzije se upisuje 240 torki, po jedna za svaki sat u periodu od 10 dana.

U slučaju korišćenja *MOLAP* servera, potrebno je obaviti dodatne korake u vezi sa upravljanjem hiperkockom. Prvi od njih je kreiranje strukture hiperkocke, uz oslonac na softversko rešenje *SQL Server Analysis Services (SSAS)*. Hiperkocka se kreira nad zvezdastom šemom baze podataka. Formiranje višedimenzionalnih nizova koji čine hiperkocku se vrši u koraku procesiranja hiperkocke. Prilikom procesiranja, preuzimaju se podaci iz činjenične tabeli i dodaju se u odgovarajuće nizove. Kao jedna od mana *MOLAP* servera navodi se redundansa podataka, usled kopiranja podataka iz činjenične tabele. Kako bi se izbeglo postojanje kopije podataka, uveden je iterativan proces formiranja i procesiranja hiperkocke. Hiperkocka se particionira po satima. Svaka particija hiperkocke će sadržati podatke o merenjima koja su nastala u toku određenog sata. U svakoj iteraciji *ECTL* procesa se obrađuju podaci koji se odnose na jedan sat – vrši se njihova ekstrakcija, agregacija i punjenje činjenične tabele agregiranim vrednostima. Za svaki obrađen deo podataka formira se posebna particija hiperkocke i particija se odmah procesira. Nakon procesiranja, moguće je obrisati preuzete podatke iz činjenične tabele, te hiperkocka postaje jedini izvor agregiranih podataka. Ovakav pristup smanjuje zahteve po pitanju prostora na disku. Agregacija vremenskih serija se obavlja pomoću uskladištene procedure, pri čemu se za svaku vrednost vremenske serije traži vrednost koja joj je prvi sledbenik, i na osnovu vremenskih trenutaka njihovih nastanaka, računa se dužina trajanja vrednosti. Kako bi se pronašla vrednost vremenske serije koja je bila validna u trenutku početka sata, potrebno je izdvojiti poslednju od svih vrednosti koje prethode prvoj vrednosti vremenske serije u okviru tog sata. To znači da se u svakoj iteraciji vrši uređivanje svih prethodnih vrednosti vremenskih serija po trenutku početka važenja kako bi se pronašla poslednja vrednost. Sa rastom rednog broja particije, raste i redni broj sata od interesa, a broj vrednosti koje prethode se povećava. Zbog loših performansi koje je takvo rešenje pokazalo prilikom testiranja, uvedena je optimizacija procedure koja vrši agregiranje. Ideja je da se izbegne traženje direktnog prethodnika prve vrednosti u odabranom satu za svaku vremensku seriju. Uvedena je tabela „*SNAPSHOT*“ čija je namena skladištenje poslednjih vrednosti svih vremenskih serija u satu koji prethodi satu od interesa. Kako bi ti podaci bili spremni na početku izvršavanja uskladištene procedure, dodaje se

još jedan korak na njen kraj, u kojem se pripremaju podaci za njeno sledeće izvršavanje: briše se sadržaj tabele „*SNAPSHOT*“ i u nju se unose poslednje vrednosti vremenskih serija u toku tekućeg sata. Na taj način, traži se maksimalni vremenski trenutak samo u okviru tekućeg sata, te količina podataka koja se uređuje ne raste sa rednim brojem particije.

MOLAP hiperkocka procesiranjem postaje fizički nezavisna od relacionog izvora podataka, dok kod *ROLAP* hiperkocke to nije slučaj. U toku procesiranja *ROLAP* hiperkocke, procesira se samo šema pozadinske baze podataka, ali ne i podaci, što je bitna razlika u odnosu na *MOLAP* pristup. Agregiranje podataka radi se na zahtev upita, te nije potrebno obaviti predizračunavanje agregiranih vrednosti. Kako bi, na zahtev upita, bili dobijeni agregirani podaci o vremenskim serijama, potrebno je definisati pogled koji hiperkocka ima na podatke. Kreira se imenovani upit koji agregira podatke o vremenskim serijama, na sličan način kao uskladištena procedura kod *MOLAP* servera.

6. REZULTATI

Definisane su sledeće metrike po kojima se mere performanse *OLAP* servera: vreme izvršenja *ECTL* procesa, i vreme izvršenja karakterističnih upita u okviru elektroenergetskih sistema. Sva merenja su vršena na računaru sa procesorom i3-4130 3,40GHz i 24GB radne memorije.

Vreme potrebno za izvršenje zajedničkog dela oba *ECTL* procesa, pripremu dimenzionih tabela, iznosi 4,7 s. Kod neoptimizovanog *MOLAP ECTL* procesa postoji rastući trend dužine trajanja izvršenja iteracija *ECTL* procesa. Dužina trajanja izvršenja *ECTL* zadatka za prvu particiju iznosi 29 minuta i 40 sekundi, a za poslednju 58 minuta i 51 sekundi. Dužina trajanja celokupnog procesa iznosi 90 sati, 9 minuta i 25 sekundi. Kod optimizovanog pristupa ne postoji rastući trend, već je dužina trajanja izvršenja iteracija konstantna, i u proseku iznosi 28 minuta i 57 sekundi. Dužina trajanja celokupnog procesa iznosi 57 sati, 54 minuta i 11 sekundi. U slučaju *ROLAP* servera, *ECTL* proces obuhvata samo procesiranje hiperkocke, bez ekstrakcije i agregacije podataka, te traje samo 2 s.

Na postavljenom primeru, bez obzira na tip servera, šablon karakterističnog upita je sledeći: „Izdvoji podatke o X vremenskih serija za period od Y dana“, $X = 10^i, i \in \{0 \dots 5\}$, $Y = \{1 \dots 5\}$. U elektroenergetskom sistemu se najčešće vrši čitanje vrednosti vremenskih serija u okviru neprekidnih vremenskih intervala, dok identifikatori odabranih tačaka najčešće nisu susedni brojevi. Stoga, prilikom tesiranja su tačke birane slučajno, dok su intervali vremena bili kontinualni. U upitima nad *MOLAP* i *ROLAP* hiperkockama su birane iste tačke. Vršene su dve grupe upita: u prvoj grupi je vršena selekcija na nivou granularnosti dana, a u drugoj na nivou granularnosti sata. Brzina izvršenja je veća na *MOLAP* serveru za sve posmatrane upite. Prilikom izdvajanja manje količine podataka, razlike u brzini na *MOLAP* i *ROLAP* serverima su manje drastične nego prilikom izdvajanja velike količine podataka. Uočeno je da kod *ROLAP* servera vreme izvršenja značajno raste sa povećanjem broja dana od interesa, dok je kod *MOLAP* servera rast gotovo

neprimetan. Razlike u performansama upita na *ROLAP* i *MOLAP* serverima postaju sve uočljivije sa porastom izdvojene količine podataka. *ROLAP* server nije imao prihvatljive performanse čitanja podataka o hiljadu vremenskih serija. Upit koji izdvaja podatke o hiljadu vremenskih serija za period od jednog dana je prekinut nakon sat vremena jer je ta dužina trajanja upita ocenjena kao neprihvatljiva za taj broj vremenskih serija.

7. DISKUSIJA

Redundansu podataka koju uvodi *MOLAP* server je moguće izbeći iterativnim izvršavanjem *ECTL* procesa. *MOLAP ECTL* proces uključuje velik broj koraka, a pokazalo se da obrada i agregiranje podataka nastalih u toku jednog sata može trajati duže od sat vremena. Uvedena optimizacija je značajno smanjila vreme potrebno za izvršenje *ECTL* procesa. *ROLAP* server ne uvodi redundansu podataka. Agregacija podataka se vrši na zahtev upita, pa *ECTL* proces traje vrlo kratko i ne obuhvata korak predizračunavanja. Jednostavnost i brzina izvršenja *ECTL* procesa predstavljaju prednost *ROLAP* servera. Još jedna prednost je pristup podacima u realnom vremenu. Kod *MOLAP* servera postoji korak predizračunavanja, usled čega *MOLAP* server ne može da pruži pristup „najsvežijim“ podacima. Kod *ROLAP* server upiti se izračunavaju nad operativnim podacima.

Predizračunavanje omogućava *MOLAP* serverima odlične performanse upita. Na *MOLAP* serveru uspešno su izvršeni svi upiti, dok su kod *ROLAP* servera uspešno izvršeni samo upiti koji zahtevaju obradu manje količine podataka. Poređenjem dužine trajanja uspešno izvršenih upita na oba servera, uočeno je da *MOLAP* server izvršava sve upite za kraće vreme. Razlika se povećava sa rastom količine podataka.

Pošto većina alata u okviru elektroenergetskog sistema podrazumeva izdvajanje podataka o velikom broju vremenskih serija, *ROLAP* server je odbačen kao moguće rešenje. Pored prednosti koje *ROLAP* server ima, nemogućnost izvršenja određenih upita je značajna mana. *ROLAP* server nije proizveo rezultate upita nad 1000 vremenskih serija u prihvatljivom roku, što ga čini nepodobnim za upotrebu u sistemu u kom se upiti često vrše nad znatno većim brojem vremenskih serija. *ROLAP* server može biti prihvatljivo rešenje kod alata čija je namena vizuelna inspekcija manjeg broja vremenskih serija od strane domenskog stručnjaka.

Neprilagođenost radu sa podacima koji imaju velik broj dimenzija se navodi kao mana *MOLAP* servera [3]. Podaci o vremenskim serijama imaju samo dve dimenzije, pa ova mana ne predstavlja realan problem u ovom slučaju. *MOLAP* serveri uvode ograničenje na količinu podataka jer se sva predizračunavanja obavljaju u trenutku izgradnje hiperkocke. Mogućnost particionisanja kocke po satima pomaže u prevazilaženju ovog problema jer je moguća postepena izgradnja hiperkocke i vršenje predizračunavanja nad delom podataka.

HOLAP server nije uzet u razmatranje jer, kod *SSAS* softverskog rešenja, podrazumeva podelu podataka tako da se u višedimenzionalnim nizovima skladište samo agregirani podaci, a atomički podaci u relacionoj bazi

podataka. Zbog toga je *HOLAP* server ekvivalentan *MOLAP* serveru za upite koji pristupaju samo agregiranim podacima, a to je jedina vrsta upita koja je uzeta u obzir.

Na osnovu navedenog, *MOLAP* server je odabran kao najpogodniji tip *OLAP* servera.

8. ZAKLJUČAK

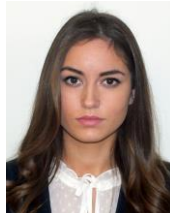
U ovom radu opisane su mogućnosti primene različitih tipova *OLAP* servera u okviru elektroenergetskog sistema. Sprovedena je uporedna analiza njihovih relevantnih karakteristika kako bi se odabrao pogodan tip. Posmatrano je vreme potrebno za izvršenje *ECTL* procesa i vreme potrebno za izvršenje karakterističnih upita. Karakteristični upiti podrazumevaju izdvajanje agregiranih podataka o različitom broju vremenskih serija u toku neprekidnih intervala vremena, koji variraju od jednog do pet dana. Za potrebe testiranja, generisan je skup podataka koji simulira scenario visokog opterećenja elektroenergetskog sistema. Dat je predlog izvršenja *ECTL* procesa za svaki od razmatranih tipova servera. Prikazani su koraci formiranja strukture hiperkocki. Dat je predlog optimizacije agregacije, u cilju omogućavanja skalabilnosti *ECTL* procesa kod *MOLAP* servera.

Većina alata u okviru elektroenergetskog sistema podrazumeva izdvajanje agregiranih podataka o velikom broju vremenskih serija za protekle periode vremena, te je *ROLAP* server odbačen kao moguće rešenje usled nemogućnosti izvršenja takvih upita. *HOLAP* server nije uzet u razmatranje jer je ekvivalentan *MOLAP* serveru za upite koji pristupaju samo agregiranim podacima, a to je jedina vrsta upita koja je uzeta u obzir. Stoga, *MOLAP* server je izdvojen kao potencijalno pogodno rešenje za agregiranje i skladištenje podataka u elektroenergetskom sistemu.

9. LITERATURA

- [1] Tam, Y. J. (1998). *Datacube: Its implementation and application in olap mining*. Simon Fraser University.
- [2] Grabova, O., Darmont, J., Chauchat, J. H., & Zolotaryova, I. (2010). Business intelligence for small and middle-sized enterprises. *ACM SIGMOD Record*, 39(2), 39-50.
- [3] Kline, N., & Snodgrass, R. T. (1995, March). Computing temporal aggregates. In *Data Engineering, 1995. Proceedings of the Eleventh International Conference on* (pp 222-231). IEEE.
- [4] Manojlović, I. & Erdeljan, A. (2017) Efficient aggregation of time series data. In *Zdravković, M., Konjović, Z., & Trajanović, M. (Eds.) ICIST 2017 Proceedings Vol.1* (pp 102-207).

Kratka biografija:



Andrijana Stefanović rođena je u Zrenjaninu 1994. god. Fakultet tehničkih nauka upisuje 2013. god. Bečelor rad iz oblasti Elektrotehnike i računarstva – Računarske nauke i informatika odbranila je 2017. god. Master rad iz iste oblasti odbranila je 2018. god.

1POL PLOTTER – VIZUELIZATOR JEDNOPOLNIH ŠEMA ELEKTRODISTRIBUTIVNE MREŽE**1POLE PLOTTER – SINGLE POLE SCHEME VISUALIZATOR**

Filip Đorđević, *Fakultet tehničkih nauka, Novi Sad*

Oblast: RAČUNARSKA GRAFIKA – Primenjeno
Softversko Inženjerstvo

Kratak sadržaj – U okviru ovog rada razvijana je aplikacija koja omogućava vizuelizaciju jednopolne šeme elektrodistributivne mreže i nudi korisniku mnoge opcije za modifikaciju prikaza.

Ključne reči: 2D vizuelizacija, grafovi, jednopolna šema

Abstract – The present work describes the development of an application that enables distribution power grid single pole scheme visualization and presents many options for modifying the display to the user.

Keywords: 2D visualization, graphs, single pole scheme

1. UVOD

Tema rada je aplikacija za vizuelizaciju jednopolnih šema elektrodistributivne mreže, pod nazivom 1Pol-ploter (slika 1).

Vizuelizacija se može okarakterisati kao tehnika kreiranja slika, dijagrama ili animacija koje prenose neku poruku. Vizuelizacija podataka je podkategorija vizuelizacije koja se, pored ostalih primena, bavi geografskim i prostornim podacima koji se abstrakuju šematskim formama.

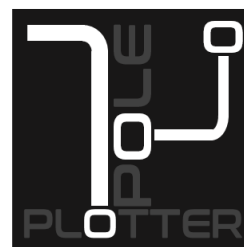
Računarska 2D grafika predstavlja generisanje digitalnih prikaza na osnovu 2D modela, kao što su npr. geometrijski oblici. Da bi se elementi elektrodistributivne mreže vizuelizovali, potrebno je da postoje podaci o njihovom položaju.

Koordinate položaja elemenata su zadate u UTM formatu, u okviru kojeg se koordinate zadaju po horizontalnoj reprezentaciji. Tokom rada aplikacije, ove vrednosti se konvertuju u decimalni format, koji čini par koordinata geografske širine i dužine i koristi se u mnogim geografsko-informacionim sistemima, kao i veb aplikacijama za prikaz mapa.

Cilj rada aplikacije je prikaz 2D jednopolne šeme elektrodistributivne mreže, koji ima strukturu grafa. Strukturu grafa čini skup objekata (čvorovi grafa) i veze između objekata (grane).

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivetić, red. prof.



Slika 1. Logo aplikacije

2. JEDNOPOLNE ŠEME

Na osnovu generalnih zahteva koji se postavljaju pred svaki energetska i tehnički sistem, strukturu elektroenergetskog sistema čine četiri podсистema: proizvodnja, prenos, distribucija i neposredna potrošnja.

Distribucija električne energije se zasniva na koncepciji izvoda („fidera“). Oni predstavljaju dovoljno jake elektroenergetske veze kojima se mogu pridružiti veći delovi jednog potrošačkog područja. Izvodi sa svojim grananjem, sve do mesta mesta priključka individualnih potrošača, čine distributivnu mrežu elektroenergetskog sistema.

Električne šeme nalaze svoju primenu u prikazu spajanja pojedinih aparata i postrojenja; prikazu brojnog stanja aparata, kablova, opreme, dalekovoda itd.

Jednopolna šema, kao vrsta električnih šema služi kao osnova za projektovanje ili jednopolni prikaz glavnih strujnih krugova (razvođenje električne energije).

2.1. Vizuelizacija jednopolne šeme

Iz perspektive računarske grafike i vizuelizacije, jednopolna šema aproksimira mrežu na ortogonalni prikaz. Ima strukturu rešetke i elementi mreže se projektuju na njene čvorove.

Za potrebe grafičkog prikaza jednopolne šeme, elementi mreže su definisani u XML fajlu. Takozvanu XML šemu podataka čini kolekcija entiteta *NetworkModel* koja sadrži modele elemenata mreže. Oni su grupisani u kolekcije svakog tipa elementa pojedinačno:

- kolekcija *Substations* sadrži podatke o trafostanicama,
- kolekcija *Nodes* sadrži podatke o čvorovima mreže,
- kolekcija *Switches* sadrži podatke o prekidačima, i
- kolekcija *Lines* sadrži podatke o vodovima.

Rešetka koja se formira na površini za crtanje u okviru aplikacije se modeluje matricom. Svaki entitet se na osnovu koordinata svog geografskog položaja pozicionira u matrici, odnosno samim tim i na prikazu. Modelovanje te matrice dovodi do toga da rešetka predstavlja svoj konkretni koordinatni sistem u okviru prostora za crtanje u aplikaciji.

Prilikom iščitavanja podataka iz XML fajla, elementi se iscrstavaju redom kojim su iščitani jer se pomoću funkcija za aproksimaciju pozicije, koje su detaljnije opisane u poglavlju 3.2, njihove koordinate položaja odmah mapiraju na položaj u matrici.

BFS algoritam koji traži najkraće putanje između zadatih čvorova kako bi iscrtao vodove se poziva tako da pronade što više putanja između čvorova tako da se linije vodova ne presecaju, da bi se potom iscrtili vodovi koji će se presecati, sa označenim tačkama presecanja.

3. IMPLEMENTACIJA REŠENJA

U okviru ovog poglavlja predstavljen je način implementacije rešenja kroz tri osnovne celine: model, logika iscrtavanja, i korisnički interfejs.

3.1. Model

Model čine klase koje sadrže osnove funkcionisanja aplikacije – entiteti prisutni na jednopolnoj šemi, klase sa logikom za formiranje rešetke jednopolne šeme i klasa koja implementira BFS algoritam.

Bazna klasa svih entiteta prisutnih na jednopolnoj šemi je klasa *SinglePoleEntity.cs* koja sadrži osnovne atribute za prikaz entiteta:

- jedinstveni brojevni identifikator entiteta,
- ime entiteta,
- decimalna vrednost X koordinate položaja entiteta, i
- decimalna vrednost Y koordinate položaja entiteta.

Ovu klasu nasleđuju klase: *SubstationEntity.cs*, koja modeluje trafostanicu; *NodeEntity.cs*, koja modeluje čvor u sistemu; i *SwitchEntity.cs*, koja modeluje prekidač i uvodi novi atribut – status, koji predstavlja status prekidača (otvoren/zatvoren). Vodove modeluje klasa *LineEntity.cs*. Na slici 2 je prikazan UML dijagram osnovnih klasa koje modeluju elektrodistributivnu mrežu i na osnovu kojih se formira jednopolna šema.

Za formiranje rešetke jednopolne šeme implementirane su tri klase: *GridSetup.cs*, *GridNode.cs* i *GridMatrix.cs*.

Klasa *GridSetup.cs* služi za pripremu podataka koji se prikazuju u okviru rešetke jednopolne šeme. Metoda *GetDataFromXML* će isparsirati XML fajl sa podacima o entitetima u mreži koji je na početku rada aplikacije predefinisani.

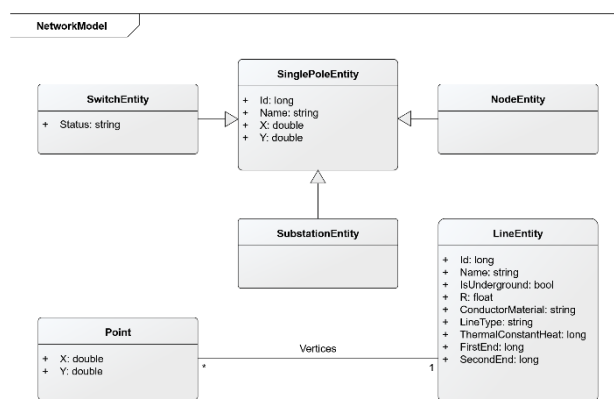
Za svaku vrstu entiteta će biti kreiran novi objekat sa vrednostima atributa iščitanim iz fajla. U XML fajlu koordinate su zapisane u UTM formatu. Konvertuju se u decimalne vrednosti pre dodele atributima. Za to se koristi metoda *ToLatLon* koja sadrži logiku za konvertovanje.

Nakon svake konverzije, tekući par koordinata se upoređuje sa trenutnim vrednostima najmanjih i najvećih vrednosti kako bi se oformila “površina” na osnovu koje će pozicije entiteta biti aproksimirane u okviru metode *GetMinAndMaxValues*.

Klasa *GridNode.cs* modeluje jedan element u okviru rešetke jednopolne šeme. Element je modelovan svojom pozicijom (brojeve vrednosti - red i kolona), koji element rešetke je roditelj, i da li je ranije pretraživan.

Klasa *GridMatrix.cs* modeluje prikaz elementa u okviru rešetke jednopolne šeme čuvajući njegov identifikator, X i Y koordinatu položaja, boju kojom se identifikuje, kao i tip entiteta (trafostanica, čvor ili prekidač).

Funkcionalnost BFS algoritma je implementirana u klasi *BFSAlgorithm.cs*. BFS algoritmu se može zadati matrica za pretragu pomoću metode *AddMatrix*, gde se dvodimenzionalni niz prosleđen kao parametar metodi dodeljuje u matricu za pretragu. Metoda *AddAdjacents* omogućava pregled zauzetih mesta u matrici oko zadatog čvora kako bi se pronašli njegovi susedni elementi, dok metoda *ShortestPathSearch* implementira pretragu prema BFS algoritmu.



Slika 2. UML dijagram klasa koje modeluju elemente mreže

3.2. Logika iscrtavanja

Logika iscrtavanja je implementirana u okviru logike glavnog prozora WPF aplikacije rešenja. Za iscrtavanje jednopolne šeme, pripremljena je rešetka na osnovu koje će biti aproksimirane pozicije entiteta.

Ovo omogućava funkcija *InitializeGridMatrix* gde se računaju dimenzije matrice, tako što se konkretna dimenzija rešetke deli sa dimenzijom markera koji predstavlja entitet. Nakon toga se kreira dvodimenzionalni niz objekta *GridMatrix* koji će sadržati sve iscrtane entitete. Poslednji korak u ovoj funkciji je inicijalizacija novog objekta klase *BFSAlgorithm.cs* čime se omogućava funkcionalnost algoritma.

Funkcija *LoadSinglePoleData* služi za učitavanje svih entiteta u pojedinačne liste za svaki tip i određuje se površina za iscrtavanje pretragom najmanjih odnosno najvećih vrednosti X i Y koordinata.

Popunjavanje matrice elementima se vrši pomoću funkcije *PlotMatrixWithData*, gde se iterira kroz svaku od lista elemenata i parametri potrebni za iscrtavanje se prosleđuju funkciji *InsertDataIntoMatrix*, u okviru koje se formiraju pojedinačni objekti klase *GridMatrix* sa svojim atributima.

Aproksimaciju pozicije na kojoj se iscrtava entitet obavlja funkcija *ApproximatePointPosition* u okviru koje se poziva konverzija vrednosti koordinata iz UTM u decimalni format. Položaj markera u odnosu na površinu na kojoj se iscrtavaju markeri se množi odgovarajućom dimenzijom rešetke kako bi se dobila pozicija u odnosu na rešetku određena celobrojn timer vrednošću koordinata X i Y.

U slučaju da ovakvim pozicioniranjem entiteta jednodolne šeme tekući entitet bude iscrtan preko nekog već postojećeg, rekurzivna funkcija *ApproximateClosestFreePosition* će se pozivati i pretraživati slobodan prostor od po jednog mesta sa svih strana postojećeg entiteta kako bi pronašla slobodno mesto za novi entitet.

Funkcija *PlotMarkers* iscrtava svaki entitet koji postoji u matrici kao krug prečnika 5 piksela, boji ga u skladu sa bojom zadatom za taj entitet, dodaje mu se *ToolTip* sa informacijama o njegovom identifikatoru i tipu. Takav objekat tipa *Ellipse* se dodaje na prikaz zadavanjem pozicije u odnosu na levu i gornju ivicu.

Kako bi vodovi bili iscrtani, iterira se kroz listu objekata koji predstavljaju vodove i za svaki se uzimaju vrednosti identifikatora entiteta na krajevima voda. Oni se potom pronalaze u matrici i sa vrednostima atributa tih objekata se kreiraju objekti potrebni za funkcionalnost BFS algoritma. Algoritmu se dodeljuje matrica, a potom se traži najkraća putanja pozivanjem metode *ShortestPathSearch* u okviru BFS algoritma.

Kada se pronađe najkraća putanja između dva entiteta, u nastavku se poziva metoda *PlotPath* koja iscrtava liniju između dva entiteta. Formira se lista tačaka koja služi za iscrtavanje izlomljene linije. Ona ima svoj *ToolTip* u okviru kojeg se ispisuje jedinstveni identifikator voda koji konkretna linija predstavlja. Pored toga, svakoj liniji se dodeljuje *ContextMenu* sa opcijom za označavanje čvorova koje ta linija spaja.

Prilikom iscrtavanja jednodolne šeme izvršena su merenja koliko je vremena potrebno da se iscrtava određen broj entiteta zavisno od dimenzija rešetke. U Tabeli 1 su prikazani rezultati ovih merenja gde je uračunato vreme i iscrtavanja vodova.

Kada se iscrtava svih 4392 entiteta vreme iscrtavanja je brže u odnosu na to kada se, u ovom primeru ne crtaju čvorovi mreže, jer kada su svi entiteti prisutni, funkcija za iscrtavanje vodova proverava postojanje oba kraja voda u matrici, što u slučaju kada oni nisu iscrtani, iziskuje dodatno vreme. Dodatno, veća rešetka predstavlja veću matricu za popunjavanje što isto zahteva dodatno vreme, ali je vizuelizacija korektnija.

Tabela 1. Vremena potrebna za iscrtavanje određenog broja entiteta mreže

Broj iscrtanih entiteta	Vreme iscrtavanja (s)	Dimenzije rešetke
4392	23.01	200x200
4392	41.25	300x300
4392	56.26	400x400
2349	31.36	200x200
2349	54.29	300x300
2349	98.74	400x400

3.3. Korisnički interfejs

Veliku ulogu u aplikacijama koje nude različite izbore predstavlja način na koji su ti izbori predstavljeni korisniku.

Prilikom razvoja korisničkog interfejsa aplikacije u programu Microsoft Visual Studio u okviru *Designer* dela WPF aplikacije, formiran je raspored kontrola koje nude različite opcije korisniku. Aplikaciju čine tri dela:

- glavni meni (prostor pri vrhu prozora),
- platno (Canvas) za vizuelizaciju jednodolnih šema, i
- legenda (pri dnu prozora).

Dizajn korisničkog interfejsa je u skladu sa bojama logoa aplikacije (slika 1). Sve kontrole su iste boje kao podloga menija, a konturne linije i tekst su bele boje.

Glavni meni sadrži opcije za učitavanje podataka i prikaz jednodolne šeme. Korisnik ima mogućnost izbora lokacije foldera za izbor XML fajlova sa podacima o mreži. Izbor se vrši klikom na dugme „Browse“, čimer direktno pronalazi putanja do foldera. Ovo se postiže korišćenjem klase *OpenFileDialog* koja poziva dijalog za izbor fajla na određenoj putanji, a naziv fajla se uklanja iz putanje kako bi bio zapamćen folder.

Korisnik može da unese putanju u okviru *ComboBox*-a, ili može da vidi sve prethodno korišćene putanje kada proširi *ComboBox*. Klikom na dugme „Apply Path“, korisnik dobija listu svih XML fajlova u okviru izabranog foldera kao opcije u okviru drugog *ComboBox*-a, za izbor fajlova. Pomoću metode *GetFiles* klase *Directory* učitana su imena svih fajlova sa ekstenzijom .xml na zadatoj lokaciji.

Izborom fajla i klikom na dugme „Load“ koje se nalazi pored *ComboBox*-a za izbor fajlova nakon nekoliko trenutaka će biti iscrtana jednodolna. Ukoliko korisnik kreira novi XML fajl u okviru trenutnog foldera izabranog u aplikaciji, klikom na dugme „Refresh“ će ažurirati fajlove u *ComboBox*-u za izbor ponovnim iščitavanjem svih fajlova u okviru foldera.

Treća opcija je mogućnost uvećanja ili umanjenja prikaza (*zoom*) jednodolne šeme. Ona je implementirana pomoću *Slider* kontrole, kojom se zadaje uvećanje, odnosno umanjenje prikaza. Pri zadavanju osnovnih atributa *Slider* kontrole, definišu se minimalne i maksimalne vrednosti, kao i korak za pomeranje klizača.

Akcija zumiranja je implementirana tako što se nad platnom (*Canvas* kontrola) na kojem je iscrtana jednopolna šema, primenjuje transformacija skaliranja, gde se kao vrednost skaliranja uzima trenutna vrednost pozicije klizača *Slider* kontrole. Dodatno je implementirana mogućnost zumiranja pritiskom na tastere „+“ i „-“ na tastaturi, zadavanjem *KeyDown* događaja, koji reaguje na pritisnut taster tastature kada god je prozor aktivan.

U okviru glavnog menija postoji dugme za poziv novog prozora sa dodatnim opcijama za izmenu prikaza jednopolne šeme. Korisnik može da promeni dimenzije rešetke na osnovu koje će se aproksimirati položaj entiteta. To se postiže unošenjem vrednosti u *TextBox* polja na vrhu prozora. Klikom na dugme „Apply“ poziva se metoda *ApplyDimensions*, unešene vrednosti se parsiraju i dodeljuju kao nove dimenzije rešetke za iscrtavanje.

Prilikom izbora boja kojom će biti obojene linije i pozadina rešetke za iscrtavanje se bira od tri opcije (SCADA, Blueprint ili Regular) koje su označene kontrolama *RadioButton*. Jedna od njih uvek mora biti označena. Nakon što korisnik klikne na dugme „OK“ odabrane boje će biti dodeljene.

Izbor boja kojima se iscrtavaju entiteti ponuđen je za sva tri tipa entiteta. Ponuđena su tri *ComboBox*-a, sa svim bojama. Nakon izbora, menja se prikaz u okviru prozora, a tek nakon klika na dugme „OK“ boje se menjaju prilikom samom iscrtavanju jednopolne šeme. Opcija „Cancel“ poništava sve promene.

Pri dnu ekrana glavnog prozora se nalazi legenda sa prikazom boja markera svakog tipa entiteta. U okviru ovih natpisa tipova elemenata je implementirana *ContextMenu* kontrola koja se poziva desnim klikom miša na tekst.

Izborom opcije da se neki tip entiteta sakrije, prilikom poziva metode za iscrtavanje taj tip entiteta se preskače. Dostupnost opcija u okviru *ContextMenu*-a se takođe menja u skladu sa time da ne bude moguće dva puta zaredom odabrati opciju sakrivanja.

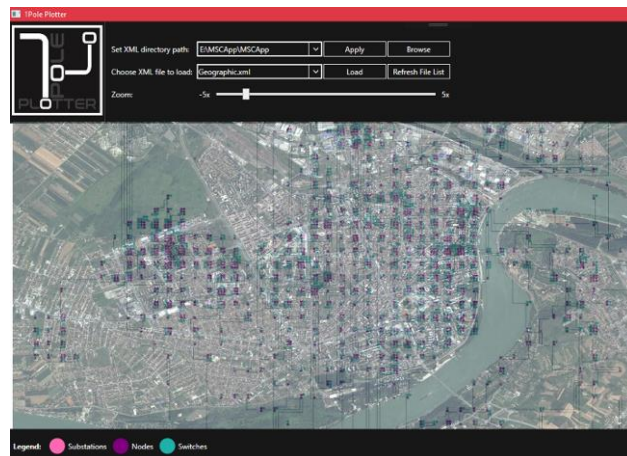
Funkcija *showNodesConnectedByLine* za prosleđeni identifikator voda pronalazi čvorove koje spaja i boji njihove markere u žutu boju. To se postiže tako što se u podacima o vodu nalaze identifikatori elemenata koje spaja. Tim konkretnim markerima (objekat klase *Ellipse*) se kao boja kojom su obojeni zadaje žuta.

4. ZAKLJUČAK

Rezultat rada jeste WPF aplikacija implementirana u programskom jeziku C# koja radi kao vizuelizator jednopolne šeme elektrodistributivne mreže i sa sobom nosi mnogo opcija za upravljanje iscrtavanjem i izmenu stila iscrtavanja jednopolne šeme, sa mogućnošću da se u daljem radu broj ovih opcija poveća.

Zadavanjem dimenzija rešetke se određuje preciznost aproksimacije, izborom fajlova se biraju podaci na osnovu kojih će biti formirana jednopolna šema, uklanjanjem pojedinih entiteta se može dobiti veća preglednost, a promenom boja se bira kojim bojama će prikazani entiteti biti kodovani.

Prilikom učitavanja podataka koriste se koordinate stvarnih elemenata elektrodistributivne mreže i oni su prikazani u skladu sa svojim geografskim rasporedom (slika 3).



Slika 3. Poređenje iscrtane jednopolne šeme na primeru Novog Sada

Prilikom razvoja aplikacije najveći izazovi su bili implementacija aproksimacije slobodne pozicije na koju će biti postavljen marker za novododati element jednopolne šeme i prilikom potrage za najkraćom putanjom između dva čvora pretragom matrice prikazanih entiteta, nakon implementacije aproksimacije pozicija.

Opcije koje bi dodatno poboljšale vizuelizator bi bile mogućnost izdvajanja određenog dela jednopolne šeme u posebni prikaz i dobiti taj deo izdvojen sa simbolima elemenata umesto markera; inkluzija digitalnih geoprostornih kartografskih podataka ispod crteža jednopolne šeme; implementacija opcije „dovrši iscrtavanje linije“ za nepotpuno iscrtane linije.

5. LITERATURA

- [1] A. Nathan, *WPF 4 Unleashed*, 1st ed., Indianapolis: Sams Publishing, 2010
- [2] Vladimir C. Strezoski, *Osnovi elektroenergetike*, Novi Sad: Fakultet Tehničkih nauka, 2014
- [3] Ivan Mihajlov, *Geovizuelizacija autogenerisane jednopolne šeme elektroenergetske distributivne mreže*, Novi Sad: Fakultet Tehničkih nauka, 2015
- [4] Michael Friendly, *Milestones in the History of thematic Cartography, statistical Graphics and Data Visualization*, 2009
- [5] 3.geosrbija.rs – kartografski podaci nacionalne infrastrukture geoprostornih podataka

Kratka biografija:



Filip Đorđević rođen je u Zrenjaninu 1994 god. Master rad na Fakultetu tehničkih nauka iz oblasti Računarske grafike – 1Pol ploter - vizuelizator jednopolnih šema elektro distributivne mreže odbranio je 2018.god.

АРХИТЕКТУРА ГЕНЕРИЧКОГ СЕРВИСА ЗА МАНИПУЛАЦИЈУ МОДЕЛИМА ПОДАТАКА**ARCHITECTURE OF GENERIC SERVICE FOR MANIPULATING DATA MODELS**

Немања Жерајић, Немања Недић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – Овај рад се бави улогом и начином функционисања сервиса за руковање подацима модела реалног система којим се софтвер бави и нуди решење које доводи до његове потпуне генеричке природе. Решење представљено у овом раду прелаже архитектуру сервиса који независно од модела података обезбеђује приступ, додавање, брисање и ажурирање података, као и њихову валидацију на потпуно генерички начин. У оквиру решења, као подршка је уведен нови сервис за рад са метаподацима. Коришћењем његових услуга подржан је начин аутоматизације рада сервиса за руковање подацима у ситуацијама када дође до измјене модела којим рукује.

Кључне речи: моделовање, метамодел, апстракција, генератор кода, UML, MOF

Abstract – This paper presents work methods of a service which is used for manipulating data models of the real system and offers a solution that leads to its complete generic nature. The solution presented in this paper identifies the architecture of the service that, independently of the data model, provides data access, addition, deletion and update of data as well as data validation in a completely generic way. As part of the solution, a new metadata exchange service has been introduced as support, and using its services provides a way to automate operation of the service for manipulating data in situations when there are modifications of the model it handles.

Keywords: modeling, metamodel, abstraction, code generator, UML, MOF

1. УВОД

Један од основних задатака сваке озбиљне софтверске архитектуре јесте омогућити максималну флексибилност и проширивост. Флексибилност и проширивост знатно поједностављују животни циклус софтверских система и представљају велику олакшицу за све актере укључене у развој и експлоатацију софтвера.

То поједностављење животног циклуса огледа се у могућности брзе реакције на промене у захтевима. Позитиван утицај флексибилности и проширивости на животни циклус софтверских система позната је ствар, али начин постизања као и ниво постизања наведених

особина, питање је на које софтверска индустрија кроз своју еволуцију перманентно тражи одговор. С обзиром да сваки информациони систем посједује сервис податка за преставу модела реалног система [5, 6] којим се информациони систем бави, архитектура овог сервиса која подржава проширивост и модуларност је предмет проучавања у овом раду.

2. ОПИС ЗАДАТКА ИСТРАЖИВАЊА

Мотивација за бављењем овом проблематиком налази се у чињеници да генеричност и проширивост сервиса за манипулацију моделом реалног система (*Model Service* даљем тексту) умногоме повећава проширивост читавог система (свих софтверских компоненти које се на тај модел ослањају). Међутим, ниво генеричке имплементације није могуће постићи у тој мјери да на *Model Service* не утичу промјене модела података. Из тога разлога у овом раду се архитектура развија у два правца.

Први правац је креирање генеричког интерфејса за остатак система, да приликом промјене модела података интерфејси остају исти и да не утичу на остале компоненте. Други правац је аутоматизација рада *Model Service*-а након измјене модела којим рукује. Из тога разлога, задатак који се поставља пред рад јесте да се одговори на питање архитектуре посматраног сервиса кроз двије цјелине. Једна цјелина би била статичке природе, непромјенива, једном имплементирана и стално кориштена, док би друга цјелина била промјенива природе и била би производ генератора извршног кода [4].

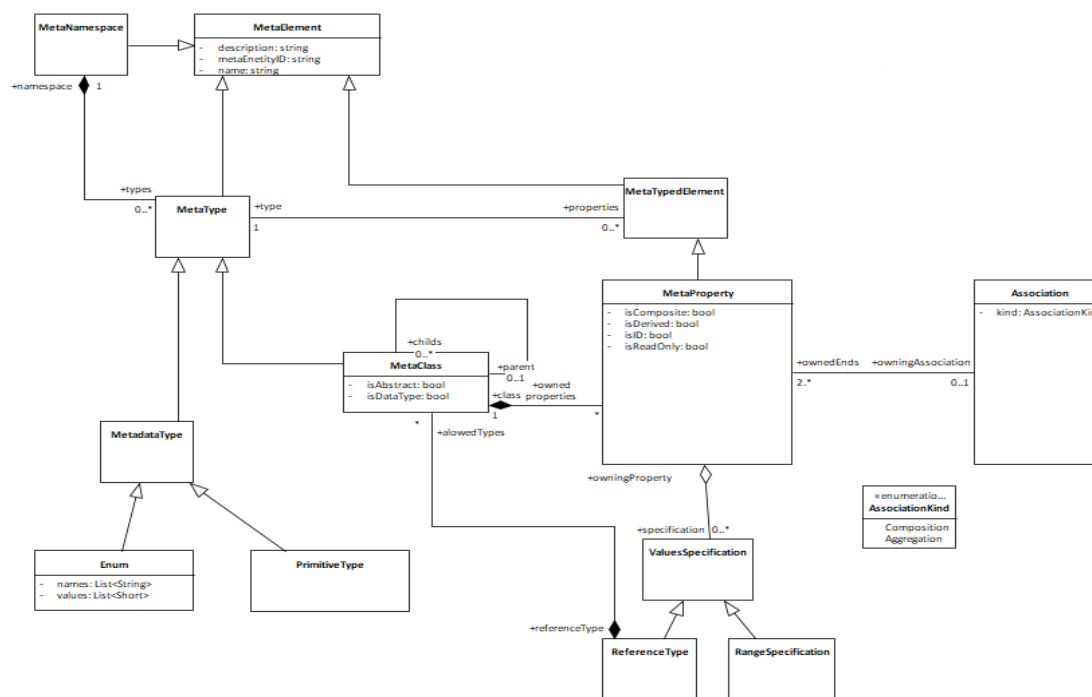
Као подршку имплементацији уведен је додатни сервис, који има функцију читавања метаподатака који описују модел *Model Service*-а, а кроз своје интерфејсе чини их доступним остатку система (у даљем тексту *Metamodel Service*). Његове услуге користи поменути генератор извршног кода, али и сам *Model Service*.

За функционисање *Metamodel Service*-а потребно је имплементирати читавање метаподатака на основу модела дефинисаног UML дијаграмом. С обзиром на то да данас постоје бројни алати за моделовање, тешко би било направити универзални адаптер за рад са дијаграмима модела. За потребе овог рада користиће се *Enterprise Architect*, а за потребе интеграције са датим алатом неопходно ће бити имплементирати адаптер, који преко одговарајућег интерфејса комуницира са *Metamodel Service*-ом и шаље му метаподатке на основу UML дијаграма.

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је био др Немања Недић, доцент.

доступним осталим чиниоцима система.



Слика 3. Metamodel

У склопу овог сервиса налази се компонента *MetamodelStore*, која служи за складиштење метаподатака, као и компоненте *ValidationEngine* и *CodeGenerator*.

Основу сервиса чини метамодел приказан на слици 3. који представља апстракцију свих модела података дефинисаних UML дијаграмом. Метамодел се у великој мјери ослања на MOF [3]. Он на врху хијерархије наслеђивања дефинише *MetaElement*, који садржи заједничке атрибуте за сваки елемент модела, а то су име (*name*), опис (*description*) и идентификатор који једнозначно одређује сваки елемент модела.

Класе су моделоване помоћу *MetaClass* елемента у метамоделу, док атрибуте класе представља *MetaProperty*. Атрибути могу да имају тип (*type*) који може бити примитивни тип, изведени тип (класа) или енумерација (описано *MetaType* елементом). Метамодел предвиђа да сваки тип припада неком именском простору (*namespace*). Класе међусобно могу имати везу генерализације, у којој могу имати улогу надкласе (*parent*) или улогу подкласе (*child*). Осим генерализације, класе могу бити повезане везом асоцијације преко одговарајућих атрибута типа референце, што је моделовано елементом *Association* у метамоделу. Асоцијација има два типа: агрегација (*Aggregation*) и композиција (*Composition*).

На нивоу метамодела је дефинисано да атрибутима у моделу могу бити додијељене одговарајуће валидационе спецификације (*ValuesSpecification*) које у зависности од типа атрибута могу представљати дозвољени опсег вриједности у случају простих типова, или скуп типова референцираних објеката у случају атрибута типа референца (показивач) на неку класу.

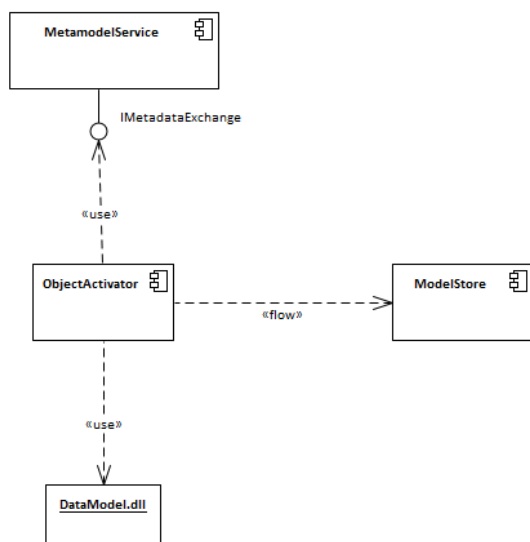
За добављање метаподатака на основу модела дефинисаног UML дијаграмом, имплементиран је *EA Adapter*, који врши учитавање одговарајућег

дијаграма, на основу њега извучи метаподатке и путем *IMetadataImport* интерфејса просљеђује их *Metamodel Service*-у.

CodeGenerator је компонента задужена за креирање дијела кода *Model Service*-а, који је зависан од модела података којим сервис рукује. Увођењем ове компоненте ријешени су неки од уочених проблема на које ово решење треба да да одговор. Као прво генерише се библиотека која садржи структуре података (*ResourceDescription*, *Property*, *PropertyValue*) неопходне за употребу *GDA* интерфејса. Поред тога, генерише се и библиотека *DataModel* која садржи све класе модела са структуром која пружа подршку за *GDA* интерфејс.

Бећ је објашњено да се *ModelStore* састоји од колекције инстанци класа типа *Container*. Сваки *Container* је намјењен за објекте одговарајуће конкретне класе модела. Да би се постигла генеричка манипулација објектима модела и избјегло да свака промјена модела захтијева рефакторисање класе *Container*, уводи се компонента *ObjectActivator* која је задужена за инстанцирање објеката модела (слика 4).

ObjectActivator своју функцију обавља користећи се техникама рефлексije и уз помоћ метаподатака које добавља са *Metamodel Service*-а. Улазна информација за ову компоненту је јединствени идентификатор класе чији објекат треба инстанцирати. Све остале метаподатке, тј. именски простор и сам назив класе, *ObjectActivator* добија позивањем одговарајуће метода *IMetadataExchange* интерфејса *Metamodel Service*-а. Након добављања потребних метаподатака, учитава се одговарајући *assembly* (у конкретном случају *DataModel.dll*), рефлексijом проналази тражена класа међу класама које *assembly* садржи, инстанцира њен објекат и смјешта у одговарајућу колекцију унутар *ModelStore*-а.



Слика 4. *ObjectActivator* у оквиру *ModelService*-а

Да би се било каква измјена модела могла сматрати успјешном, потребно је да се изврши процес валидације података модела који су захваћени том измјеном. На нивоу модела података предвиђене су двије врсте валидација: *RangeSpecification* (опсег дозвољених вриједности атрибута примитивног типа) и *ReferenceType* (скуп могућих типова референцираних објеката). Током измјене модела, *Model Service* прикупља идентификаторе објеката на које су измјене утицале. Након што је цјелокупан скуп измјена примјењен на модел, прибављају се објекти на основу прикупљених идентификатора и шаљу на валидацију. С обзиром да *Metamodel Service* чува податке који описују коректност атрибута (*RangeTypeSpecification* и *ReferenceType* објекти), *Model Service* му се обраћа преко *IValidation* интерфејса и као одговор очекује индикацију да ли су сви подаци валидни. Уколико неки од измењених објеката нису валидни, *Model Service* добија списак тих објеката и атрибута у оквиру њих који нису коректни. Уколико дође до појаве невалидности модела, читава акција измјене модела се поништава, односно задржава се стање модела прије захтјева за измјеном.

4. ЗАКЉУЧАК

Предмет истраживања у овом раду био је сервис за манипулацију подацима реалног система којим се софтвер бави. Акценат је био на његовој архитектури која предвиђа имплементацију на што флексибилнији начин. Мотивација за бављење овом проблематиком налази се у уоченим проблемима који прате промјену модела података којима сервис рукује. Представљено решење у великој мјери доприноси проширивости сервиса за манипулацију подацима, а с тим у вези и читав систем који се на њега ослања чини флексибилнијим.

Потпуном аутоматизацијом измјена модела података постигнута је и велика уштеда времена потребног да се промјене имплементирају писањем програмског кода.

Укључивање сервиса за манипулацију метподацима отвара нове могућности и обезбјеђује ефикасан начин размјене метаподатака унутар система. То је основа за даљи развој компоненти чије функције се заснивају на метаподацима, као што би био генератор корисничког интерфејса или адаптера за интеграције са другим системима. Основни допринос овог рада огледа се у потпуној аутоматизацији промјена сервиса за манипулацију подацима које настају као последица промјене модела података којима рукује, као и у постављању полазних оквира и давању смјерница за даљи развој заснован на метаподацима чија је доступност обезбјеђена у оквиру представљеног решења.

4. LITERATURA

- [1] IEC: 61970-403 Ed.1: Energy management system application program interface (EMS-API) - Part 403: Generic data access, 2008.
- [2] *Unified Modeling Language (UML), Version 2.5.1*, <http://www.omg.org/spec/UML/2.5.1/>, datum pristupa: 21.08. 2018.
- [3] Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification, Final Adopted Specification, ptc/051101, dosupno na <http://www.omg.org/>, datum pristupa: 21.08. 2018.
- [4] J.Bezivin, O.Gerbe: Towards a Precise Definition of the OMG/MDA Framework, proceedings of the 16th Annual International Conference on Automated Software Engineering (ASE), 2001.
- [5] K.Hazzard, J.Bock: Metaprogramming in .NET, Manning Publications, ISBN 978-1617290268, 2013.
- [6] Jean Bézivin, Olivier Gerbé: Towards a Precise Definition of the OMG/MDA Framework, Proceedings of the 16th Annual International Conference on Automated Software Engineering (ASE), 2001.

Кратка биографија:



Немања Жерајић рођен је 30.4.1994. у Требињу. Завршио је гимназију у СШЦ „Перо Слијепчевић“ у Гацку 2013. године, а дипломирао на основним студијама Електротехнике и рачунарства на Факултету Техничких Наука у Новом Саду 2017. године.



Др Немања Недић рођен је 7.10.1981. у Сомбору. Докторирао 2016. године на Факултету техничких наука у Новом Саду у исте године изабран у звање доцента. Ужа научна област му је Примењено софтверско инжењерство.

**РАЗВОЈ АПЛИКАЦИЈЕ ЗА ДЕТЕКЦИЈУ И АНАЛИЗУ ДУПЛИРАНОГ КОДА
УПОТРЕБОМ ЈСЦПД АЛАТА****DETECTION AND ANALYSIS OF DUPLICATED CODE APPLICATION
DEVELOPMENT USING JSCPD TOOL**

Бранко Савић, Факултет техничких наука, Нови Сад

Oblast – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – У раду су анализирани неки од постојећих алгоритама за детекцију дуплираног кода. Изабран је и анализиран ЈСЦПД алат који је искоришћен за развој веб апликације. Веб апликација пружа визуелан и информативан приказ дуплираног кода. На тај начин су кориснику је доступна информација о проблематичним деловима неког програмског кода.

Кључне речи: Дуплирани код, анализа дуплираног кода, jscpd

Abstract – The paper analyzes some of the existing algorithms for detection of duplicate code. The JSCPD tool was analyzed and used for web application development. The web application provides a visual and informative duplicate code display. In this way, users are provided with information about the problematic parts of a program code.

Keywords: Duplicated code, duplicated code analysis, jscpd

1. УВОД

Задатак овог рада јесте анализа алгоритама за детектовање дуплираног кода, као и анализа алгоритама за претрагу речи у тексту, који се могу модификовати и искористити као алгоритам за претрагу и анализу дуплираног програмског кода.

За потребе бољег увида у постојеће техничке недостатке одређеног програмског кода, потребно је развити алат у виду веб апликације, који ће бити у стању да прикаже места на којима се налази дуплирани код. Приказ дуплираног кода би требало да буде визуелан, информативан и претражив, како би се могао добити увид у места која су најпроблематичнија за одређени програмски код.

Апликација треба да понуди могућност прегледа свих фајлова у којима се налази дуплирани код, као и приказ садржаја фајла са означеним дуплираним деловима кода.

Такође алат би требало да подржи могућност конфигурације, односно избора параметара на основу којих ће се одредити да ли је неки део кода дуплиран.

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је био др Дарко Чапко ванр. проф.

2. ТЕОРИЈСКЕ ОСНОВЕ

Дуплирани код је део програмског кода који се појављује више од једном унутар једног или више програма које поседује и одржава одређени ентитет. Сегменти кода који су дуплирани називају се клоновима. Дуплирани код може настати на више начина. Најчешћи узрок настајања дуплираног кода јесте *copy-paste* програмирање.

Други, такође чест узрок појаве дуплираног кода јесте ситуација у којој више програмера ради на различитим деловима исте програмске компоненте у исто време. Пошто раде на различитим задацима, нису свесни да је неко већ написао код који они могу искористити за сличан проблем.

Дуплирање дела програмског кода који представља одређену слабост система или има неку грешку, доводи до појаве слабости, односно грешке система на два различита места. Отклањањем слабости на једном делу кода, она и даље постоји у дуплираном делу.

Неколико студија показује да око 5% до 20% софтверских система може садржавати дуплирани код, што је у основи резултат копирања постојећих фрагмената кода и његова употреба без или са мањим модификацијама [1].

Најпримитивнији начин за детектовање дуплираног кода јесте поређење једнакости стрингова. Алат имплементиран на тај начин да пореди стрингове одређене дужине би препознао дупликат уколико би дужина стринга која се пореди била мања од броја карактера који се понављају. Овај начин не врши никакву обраду, не препознаје белине(размак, таб и нови ред), не садржи у себи никакву семантику, па самим тим не представља применљиво решење. Описани приступ потребно је модификовати, тако да врши уклањање белина и коментара. Различити програмски језици подржавају различите начине за писање коментара, па самим тим, и програмски језик који се посматра игра важну улогу у томе да ли неки ред програмског кода представља коментар. Мана оваквог приступа огледа се у томе да алат имплементиран на овакав начин такође не поседује никакву семантику. Он само једноставно пореди стрингове из кога су одстрањене непотребне белине и коментари. Ако се у разматрање узму две функције које извршавају исту програмску логику, простим поређењем стрингова не може се доћи до закључка да

оне раде исти посао. Разлог је тај што су функције и променљиве које се користе могу бити назване различитим именима.

Још један недостатак оваквог приступа су перформансе. За само један документ било би потребно за сваки карактер направити стринг који обухвата одређени број узастопних карактера, и проћи кроз све остале карактере, креирати стрингове исте дужине и поредити их са претходним. На тај начин добили бисмо квадратну сложеност, где број елемената од кога зависи сложеност представља број карактера који се налазе у оквиру посматраног фајла. Ако узмемо у обзир и то да је потребно вршити проверу на нивоу неког пројекта, долазимо до тога да је потребно то исто извршити за све фајлове у којима се налази програмски код. Из тог разлога потребно је пронаћи бољи и ефикаснији начин од простог поређења стрингова. Један од начина јесте помоћу лексичке анализе програмског кода.

Лексичка анализа је процес претварања низа карактера у низ **токена**. Програм задужен за лексичку анализу назива се скенер. Скенер препознаје лексичке јединице из улазног текста. Лексичка јединица назива се **симбол**. Скенер преузима знак по знак улазног текста и покушава да препозна симбол [2].

Различити програмски језици садрже различите кључне речи, као и могућност писања идентификатора са различитим знаковима као што је доња црта. Из тога се долази до закључка да је приликом реализације скенера потребно обратити пажњу и на то који програмски језик је у питању, односно који се скуп симбола посматра.

3. АЛГОРИТМИ ЗА ДЕТЕКТОВАЊЕ ДУПЛИРАНОГ КОДА

Најпознатији алгоритми који се могу употребити за детекцију дуплираног кода су:

- Рабин-Карп алгоритам,
- Коришћење апстрактног синтаксног стабла,
- Детекција визуелног клона,
- Бејкерин алгоритам, итд.

3.1. Рабин-Карп алгоритам

Рабин-Карп алгоритам је *string-searching* алгоритам осмишљен од стране Ричард Карпа и Михаел Рабина. *String-searching* или *string-matching* алгоритми представља класу алгоритама који покушавају да пронађу место где се један или више стрингова налазе у већем стрингу или тексту. Овај алгоритам користи хеширање да пронађе било коју комбинацију стрингова у тексту [3].

Алгоритам ради на следећи начин: Основни елемент овог алгоритма јесте хеш функција. Хеш функција сваком стрингу додељује хеш вредност. Хеш вредност представља неки цео број. Рабин-Карп алгоритам проверава да ли су хеш вредности сваког стринга

исте. Уколико се хеш вредност разликује, онда се и стрингови разликују, а ако је хеш вредност једнака, онда се сматра да су и стрингови једнаки. Иако тражење хеш вредности за сваки стринг може трошити доста времена, поређење целобројних бројева је знатно брже у односу на поређење стрингова.

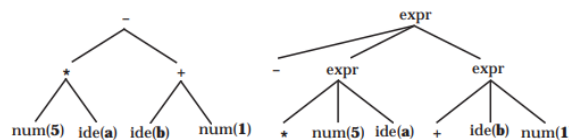
3.2. Апстрактно синтаксно стабло

Апстрактно синтаксно стабло је коначно стабло које представља апстрактну синтаксну структуру кода написаног у неком програмском језику. [4]

Уколико имамо следећи израз:

$$5 * a - (b + 1)$$

Пример синтаксног стабла за наведени израз дат је на слици 3.1. [4]



Слика 3.1. Пример апстрактног синтаксног стабла

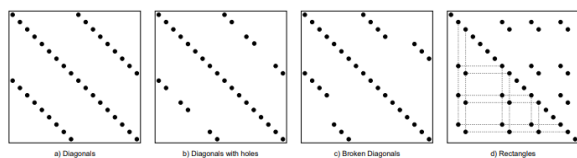
Са слике 3.1. може се приметити да су оператори представљени унутрашњим чворовима, док су листовима представљене променљиве и константне вредности. Постоје разни начини за коришћење апстрактног синтаксног стабла приликом детекције дуплираног кода. Један од начина да се искористи апстрактно стабло за детекцију дуплираног кода јесте поређење различитих делова стабла и провера да ли се у стаблу налазе два или више једнаких, тј. „сличних“ подстабала.

3.3. Детекција визуелног клона

Детекција визуелног клона заснива се на теорији да се поређење програмског кода врши поређењем сваке две линије кода. То поређење производи огромну количину димензионалних података, тј. матрицу где свака ћелија представља поређење и садржи вредност, осим ако се две линије не поклапају. Следећи корак је онда пронаћи зоне од интереса матрице. Матрица се може испитати путем аутоматског или истраживачког приступа:

Аутоматско испитивање: алат тражи неколико познатих конфигурација тачака у матрици и доставља извештај о инстанцама које су пронађене. Овај приступ је ефикасан зато што аутоматски проналази занимљива места у коду, што је погодно за велике количине података.

Међутим, тачне конфигурације за тражење морају бити претходно програмиране. Објекти аутоматског претраживања могу се кретати од једноставних дијагонала са рупицама као у примеру на слици 3.2. до тачкастих кластера пронађених помоћу статистичких метода.



Слика 3.2. Различите конфигурације тачака

Визуелна истраживања: Користећи алат који графички приказује матрицу за упоређивање, инжењер прегледа матрицу, зумира на занимљивим местима и кликом на тачке, прегледа изворни код који припада одређеном подудару.

Неке интересантне конфигурације формиране тачкицама у матрицама су следеће:

- дијагонале тачака означавају копиране секвенце изворног кода,
- секвенце које имају рупе у њима указују на то да је део копиране секвенце промењен,
- изломљене секвенце с помереним доњим деловима указују на то да је нови део кода убачен, и
- правоугаоне конфигурације указују на периодичне појаве истог кода [5].

Овакав приступ претраге дуплираног кода може детектовати делове кода који су најчешће добијени *copy-paste* програмирањем. Овај метод препознаје дуплирани код и када се он разликује само у називима варијабли, јер може препознати дуплирани код по кључним речима и на тај начин формирати тачке у том делу. Предност оваквог приступа огледа се у томе што је могуће урадити и анализу дупликата, зашто се десило дуплирање, да ли је нешто измењено, избрисано, додато итд.

4. ОПИС КОРИШЋЕНИХ АЛАТА И ТЕХНОЛОГИЈА

Пошто је одлучено да цео код буде писан у *JavaScript* програмском језику, као серверско окружење за извршавање *JavaScript* програмског кода коришћен је *Node.js*.

Npm је менаџер пакета за *JavaScript* програмски језик који је коришћен за инсталирање потребних *JavaScript* пакета. Он је такође подразумевани менаџер пакета за *JavaScript* радно окружење *Node.js* [6].

Да би се омогућило писање кода у *ECMAScript* 2015 стандарду, коришћен је *Babel* како би се писани код превео на *JavaScript* који је разумљив данашњим претраживачима.

Express представља веб *framework* за *Node.js*. Користи се за израду серверског дела веб апликација. За развој клијентског дела веб апликације коришћена је библиотека *React*. *ESLint* је *open-source JavaScript* линтинг програм. Код линтинг је врста статичке анализе која се често користи за проналажење проблематичних образаца или кодова који се не придржавају одређених смерница за стил програмирања. За потребе тестирања коришћен је

framework Jasmine. Приликом дизајнирања корисничког интерфејса коришћен је *Sass*. *Sass* је стилски језик који се преводи у *CSS*.

Webpack представља отворени *JavaScript* модул *bundler*. Он на основу свих инсталираних модула и њихових зависних модула генерише статичке фајлове који представљају те модуле. Поред наведеног још су коришћени *HTML*, *CORS*, *jQuery* и *Bootstrap*.

5. JCPCD АЛАТ

JSCPD (JavaScript Copy-Paste Detector) је *open-source* алат одржаван на *github*-у. За писање алата коришћен је *CoffeeScript* који се приликом извршавања преводи у *JavaScript*. Пакет се инсталира помоћу *npm*-а. Може се користити преко конзоле, где се уз одговарајуће параметре може добити резултат на конзоли. Исто тако, алат може бити искоришћен и приликом развоја друге апликације као компонента која поседује свој API.

Иако је настао као алат за детекцију дууплираног *JavaScript* кода, *JSCPD* сада подржава различите програмске језике као што су C++, C#, Java, PHP итд. *JSCPD* алат ради на принципу најсличнијем Рабин-Карповом алгоритму. На основу токена креира хеш вредности. Прво препозна све фајлове који би требало да буду прочитани. Затим пролази кроз сваки фајл, препознаје токене. Након што су препознати токени, у сваком кораку итератор се помера за један токен. Узима се онолико токена, у зависности од параметра који је пристигао као минималан број токена који се препознаје као дупликат. Затим се од тих токена креира хеш вредност, која се пореди са мапом која садржи све дотадашње хеш вредности.

6. ОПИС РЕШЕЊА ПРОБЛЕМА

Задатак овог рада јесте развој веб апликације која се ослања на *JSCPD* алат. Апликација кориснику обезбеђује кориснички интерфејс који кориснику нуди могућност избора фајлова, и параметара за које је потребно урадити анализу дуплираног кода, после чега се кориснику приказују резултати и датотеке са приказаним дуплираним програмским кодом.

Први корак приликом имплементације оваквог система јесте избор архитектуре. Помоћу *Express framework*-а који се ослања на *Node.js* креиран је *Web API*.

Како се улога сервера заснива на томе да прими захтев са одређеним параметрима, врши обраду захтева и враћа одговор, није потребно никакво привремено чување или смештање података у базу података.

Следећи корак у имплементацији сервера јесте избор модела података.

Модел података на серверу треба да покрије облик у којем параметри стижу до сервера и да пошаље одговор такође у виду *JSON* објекта са свим клоновима и потребним статистичким подацима.

Након што је креиран модел потребно је обезбедити флексибилан сервер тако да је могуће подржати и друге алате за претрагу. Иако је модел података

сличан формату података **JSCPD** изабраног алата, постоје одређене разлике. За потребе мапирања модела података **JSCPD** алата на модел података сервера, формиран је **JSCPD** адаптер.

Серверски део је имплементиран тако да поседује 3 контролера. Један контролер је задужен за враћање резултата обраде захтева за дуплираним кодом. Као што је већ објашњено до сада, контролер преко сервиса **InspectorService** позива изабрани инсталирани алат који врши обраду захтева.

На клијентској страни кориснику је потребно обезбедити преглед структуре одабраног фолдера у облику стабла. Задатак апликације је да за изабрани директоријум прође кроз све фајлове који одговарају задатим условима и изврши обраду.

За враћање структуре стабла кориснику задужен је **folderStructureController**. Такође, клијенту је потребно обезбедити преглед фајлова у којима се налазе дуплирати.

Потребно је да постоји могућност да клијент отвори фајл у коме се налази дупликат и да може да га прегледа. Стога је на серверу имплементиран контролер **getFilesController**. Задатак овог контролера је да на захтев корисника пошаље читав садржај тражене датотеке.

6.1. Развој клијентског дела веб апликације

Након што је завршен сервер, потребно је имплементирати и клијентски део апликације. Коренску компоненту чини **CIVIEW** компонента, која садржи и управља са 6 компоненти:

- **TopNavBar**,
- **TreeView**,
- **DuplicateList**,
- **DuplicateFile**,
- **Form**, и
- **Statistics**.

TopNavBar је компонента задужена за управљање заглављем, док је **TreeView** је компонента задужена за управљање структуром стабла које представља изабрани директоријум.

DuplicateList је компонента задужена за приказ парова назива датотека у којима се налази детектовани дуплицирани код, а након што сервер врати одговор, садржај датотеке се приказује помоћу компоненте

DuplicateFile. **Form** је компонента која рендерује и управља формом за унос параметара. Проверава избор алата који се користи након чега мења садржај форме. Приликом потврде слања форме серверу, врши се

валидација форме и уколико је валидација успешна, о томе се обавештава **CIVIEW** компонента.

Statistics је компонента задужена за приказ статистичких података.

7. ЗАКЉУЧАК

Задатак рада био је анализа постојећих алгоритама за детектовање дуплираног кода, као и развој веб апликације која, ослањајући се на **JSCPD** алат може да визуализује и информисе корисника о критичним местима у одређеном програмском коду. Анализирани су разни алгоритми, наведене њихове предности и недостаци. Развијена веб апликација нуди увид у проблематична места у коду. Правци у којима је потребно наставити развој ове веб апликације јесте подршка за још постојећих алата развијених за детекцију дуплираног кода.

8. ЛИТЕРАТУРА

- [1] Chanchal Kumar Roy, James R. Cordy, „A Survey on Software Clone Detection Research”, Queen's School of Computing, Ontario, 2007.
- [2] Зорица Сувајдин Ракић, „Програмски преводаици“, слајдови са предавања, Факултет техничких наука, Нови Сад, 2018.
- [3] Richard M. Karp, Michael O. Rabin, „Efficient randomized pattern-matching algorithms“, IBM Journal of Research and Development - Mathematics and computing, 1987.
- [4] Kenneth Slonneger, Barry L. Kurtz, „Formal Syntax and Semantics of Programming Languages“, Addison-Wesley Publishing Company, 1995.
- [5] Matthias Rieger, Stephane Ducasse, „Visual Detection of Duplicated Code“, Object-Oriented Technology (ECOOP'98 Workshop Reader), 1998.
- [6] <https://docs.npmjs.com/getting-started/what-is-npm>, датум приступа: август 2018.

Кратка биографија



Бранко Савић рођен је 09.02.1994. године у Власеници, БиХ. Завршио је Основну школу "Јован Дучић" у Шековићима, после чега уписује Средњу школу "Петар II Петровић Његош", смер гимназија - општи смер. Факултет техничких наука, смер Електроенергетски софтверски инжењеринг, уписује школске 2013/2014. године. Основне студије је завршио 2017. године након чега је уписао мастер студије, смер Примењено софтверско инжењерство. Положио је све испите прописане планом и програмом.

**MEHANIZMI PRAĆENJA POTROŠNJE RESURSA PO ORGANIZACIJI PRI
MIGRACIJI NA MULTI-TENANT SAAS****TENANT RESOURCE MONITORING MECHANISMS FOR MIGRATION TO MULTI-
TENANT SAAS**

Radislav Zekanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je predstavljen problem praćenja korišćenosti resursa i izolacije podataka u multi-tenant arhitekturi. Opisana je multi-tenant arhitektura i navedene su prednosti i nedostaci u odnosu na multi-instance arhitekturu. Nakon toga, dat je opis predloženog rješenja. Na kraju rada predstavljeni su rezultati poređenja različitih pristupa izolacije podataka i dat je primjer generisanog izvještaja o korišćenosti resursa po korisniku.

Abstract – This paper covers the problem of resource usage monitoring and data isolation in multitenant architecture. Also, in this paper will be described multitenant architecture, and their advantages and disadvantages relative to multi-instance architecture. After that, architecture of the proposed solution is given. At the end of the paper, the results of the comparison are presented and given an example of the generated report of the resource usage per tenant.

Cljučne reči: Cloud, multi-tenant, izolacija podataka, praćenje korišćenosti resursa

1. UVOD

Način na koji se softver i aplikacije isporučuju korisnicima i organizacijama, mnogo se promijenio u posljednjih nekoliko godina. Tradicionalni model isporuke, poznat kao *on-premise*, sve se manje koristi i sve je veći broj proizvođača koji svoje usluge i rješenja isporučuju koristeći neki od tri modela Cloud servisa: IaaS, PaaS i SaaS. Mnogo je razloga zbog kojih sve veći broj proizvođača softvera koristi Cloud tehnologije prilikom isporuke softverskih rješenja, ovo su neki od njih: nagli rast podataka ograničio je sposobnost tradicionalnih tehnologija za skladištenje i upravljanje podacima [1], niži početni i operativni troškovi, manje potrebno tehničko znanje, skalabilnost i lakša isporuka proizvoda. Međutim, da bi se postigle navedene prednosti, sam prelazak na Cloud tehnologije nije dovoljan. U velikom broju slučajeva, prelazak može koštati mnogo više od korišćenja *on-premise* modela isporuke. Prelazak na Cloud tehnologije sa sobom nosi dodatne probleme i izazove sa kojima se proizvođači softverskih usluga moraju boriti. U cilju ostvarivanja maksimalnih prihoda mnogi proizvođači posežu za redizajnom arhitektura svojih aplikacija. Kako bi se izabrala odgovarajuća arhitektura i kako bi prelazak na Cloud tehnologije zaista bio isplativ, u ovom radu biće opisane i analizirane postojeće softverske arhitekture.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Srđan Vukmirović, vanr.prof.

2. TEORIJSKE OSNOVE

Izbor arhitekture sistema najviše zavisi od problema koji se tim sistemom rješava i tehnologija koje se koriste prilikom njegove implementacije. Rezultat izbora određene arhitekture krajnjem korisniku ne znači mnogo. Međutim, izbor odgovarajuće arhitekture proizvođaču softvera može donijeti određene benefite u pogledu rada i efikasnosti aplikacije, kao i uštede prilikom njenog isporučivanja. U nastavku će biti opisana dva arhitekturna šablona, *multi-tenant* i *multi-instance*, i biće razmotrene prednosti i mane ovih pristupa.

2.1. Multi-instance arhitekturni šablon

U *multi-instance* arhitekturnom šablonu svaki korisnik koji koristi aplikaciju posjeduje svoju vlastitu odvojenu instancu aplikacije, sa svojom vlastitom bazom podataka [3]. To znači da nije moguće da se desi da podaci jednog korisnika budu pomiješani sa podacima ostalih korisnika. Korišćenjem ove arhitekture i modela isporuke ostvaruju se sljedeće prednosti: istinska izolacija podataka, visoka dostupnost, *customer-driven* raspored nadograđivanja aplikacije i nezavisnost u slučaju ispada. Pored navedenih prednosti ova arhitektura posjeduje i nedostatke zbog kojih se ona sve manje koristi za aplikacije koje se isporučuju koristeći neki od prethodno navedenih Cloud modela. Glavni nedostaci ove arhitekture su: duže vrijeme potrebno za instaliranje aplikacije i veći troškovi prilikom instaliranja i rada aplikacije.

2.2. Multi-tenant arhitekturni šablon

Razvoj i usvajanje *multi-tenant* arhitekture u velikoj mjeri promoviše računarstvo u oblaku (*Cloud Computing*) [4], koje ima za cilj bolje iskorištavanje resursa Cloud snabdjevača. Da bi se smanjili ukupni troškovi vlasništva, neke od SaaS ponuda dodjeljuju korisnike različitih kupaca istoj instanci aplikacije. Aplikacije dizajnirane da uslužuju korisnike različitih kupaca sa jednom instancom aplikacije, nazivaju se *multi-tenant* aplikacije. Za razliku od *multi-instance* pristupa, u *multi-tenant* arhitekturnom šablonu različiti korisnici pristupaju istoj instanci aplikacije, izvršavaju na istom operativnom sistemu i na istom hardveru. Pored boljeg iskorištavanja resursa, ovaj pristup pojednostavljuje dodavanje novih funkcionalnosti aplikacije i njeno lakše održavanje (*maintenance*). Najveći nedostaci ovog pristupa su: izolacija podataka, zavisnost u slučaju ispada i otežano prilagođavanje korisniku (*customization*). Međutim, sve češće korišćenje *multi-tenant* arhitekture uslovljeno je njenim prednostima u odnosu na druge arhitekture. Osnovne prednosti korišćenja *multi-tenant* arhitekture su: niži troškovi rada i

vremenska efikasnost. Ukoliko je potrebno uraditi nadogradnju postojećeg sistema, posao nadogradnje je potrebno odraditi na samo jednoj instanci aplikacije, čime se ostvaruje značajna vremenska efikasnost.

Iz navedenih činjenica može se uočiti da *multi-tenant* arhitektura predstavlja ekonomičnije i vremenski efikasnije rješenje. Upravo su ova dva parametra najbitnija za proizvođače softverskih usluga, te je zbog toga *multi-tenant* arhitektura sve više zastupljena u novijim softverskim rješenjima.

3. ZADATAK RADA

Zadatak rada bio je da se implementira softversko rješenje za praćenje korišćenosti dijeljenih resursa u cilju omogućavanja naplate njihovog korišćenja od strane ponuđača softvera. Kako bi se to postiglo, bilo je neophodno implementirati sistem koji omogućava korišćenje dijeljenih resursa između više različitih korisnika.

Takav sistem implementiran je korišćenjem *multi-tenant* arhitekture. Navedeni sistem predstavlja aplikativni program čiji je zadatak da analizira i prikazuje incidente nastale u elektroenergetskoj mreži u vremenu njihovog nastanka. Zbog postojanja višekorisničkog režima rada, bilo je neophodno obezbijediti i potrebnu izolaciju podataka. Problem izolacije podataka rješavan je na dva načina, pri čemu su testirana oba pristupa i na osnovu čega je izabrano bolje rješenje.

Nakon implementacije prethodno opisanog aplikativnog rješenja, bilo je potrebno implementirati mehanizam za praćenje korišćenosti resursa. Dobijene rezultate bilo je potrebno sačuvati u datoteku u kojoj će biti omogućen prikaz rezultata po korisniku, za svaki mjesec tokom godine. Takođe, kreirana datoteka je trebalo da sadrži i prikaz korišćenosti resursa po korisniku, na nivou cijele godine. S obzirom na činjenicu da podaci elektroenergetske mreže predstavljaju kritične podatke i s obzirom da je mali broj kompanija koje se usuđuju da svoje kritične podatke čuvaju u *Cloud* skladištima podataka, bilo je neophodno implementirati bezbjednosni mehanizam kojim će se ovaj problem riješiti.

4. AKTUELNO STANJE U OBLASTI

Obezbjedjivanje višekorisničkog režima rada predstavlja osnovni zahtjev skoro svake aplikacije. Kako bi ostvarili što veći profit, proizvođači aplikacija sve češće svoja rješenja baziraju na nekom od pristupa za dijeljenje resursa između korisnika. U radu [3] razmatrane su prednosti i nedostaci korišćenja *multi-tenant* arhitekture za prethodno opisana rješenja. Kao osnovne prednosti ovog pristupa, navode se lakša isporuku proizvoda i poboljšanje iskorišćenosti hardvera koji aplikacija koristi. Takođe, u ovom radu se tvrdi da ova dva faktora značajno smanjuju troškove instaliranja i održavanja aplikacije. Iako je ušteda resursa neminovna, potrošnja resursa po različitim korisnicima postaje otežana, što je fokus ovog rada. Kao osnovne probleme za prelazak na *multi-tenant* arhitekturu, autor rada navodi strah kompanija zbog visokih početnih troškova i njihovu zabrinutost da *multi-tenant* arhitektura može donijeti nove probleme u procesu održavanja softvera.

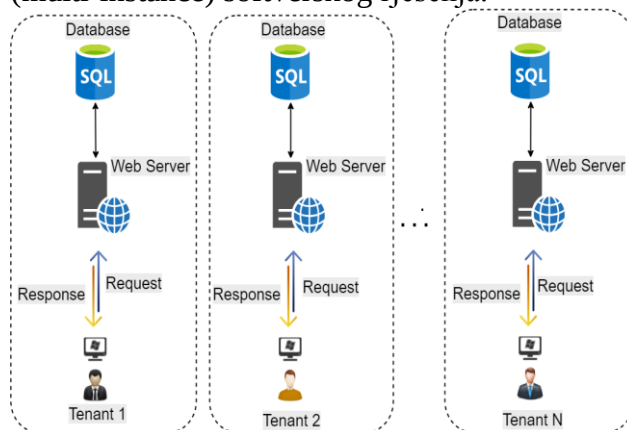
U radu [5] opisan je problem izolacije podataka u *multi-tenant* SaaS okruženju i dati su prijedlozi rješenja. Kao pristupi za izolaciju podataka, navode se: *separate database*, *shared database separate schema* i *shared database shared schema*. Slučaj koji nije razmatran, a koji je obuhvaćen u ovom radu, jeste pristup poznat pod imenom *row isolation*. Rezultati testiranja ovog pristupa dati su u poglavlju 6.1, pri čemu su dobijeni rezultati upoređeni sa *shared database shared schema* (dijeljena baza dijeljena šema) pristupom.

U radu [6] opisani su modeli naplate usluga za SaaS aplikacije (*pricing models*). U ovom radu se navodi da postoji mnogo različitih načina za naplatu SaaS usluga i da su modeli naplate veoma raznovrsni i složeni. Kao neki od najčešćih tokova naplate, navode se: prihodi od naplaćivanja mjesečne ili godišnje pretplate (*subscription*), prihodi zasnovani na reklamiranju proizvoda, prihodi zasnovani na transakcijama, prihodi od implementacije i održavanja softvera i prihodi od licenciranja softverskog proizvoda. Prihodi koji su od najvećeg značaja za ovaj rad su prihodi zasnovani na transakcijama. Implementirani mehanizam za praćenje korišćenosti resursa funkcioniše na osnovu praćenja broja pristupa i ukupnog vremena korišćenja resursa, pri čemu se svaki pristup može smatrati jednom transakcijom.

Za razliku od rada [6] koji se bavi analizom svih mogućih prihoda, u radu [7] analizirane su konkretne metode za naplatu *Cloud* usluga na osnovu određenih metrika. Tako se npr. kao metrika za korišćenje servera koristi sat, metrika za sistemsko skladište koristi sat po GB. Pored navedenih metrika, analiziran je i sistem naplate poznat kao *on-demand*. Za ovaj pristup navode se vrste naplate na osnovu tipa instance. Tipovi koje instance može posjedovati su: *Standard Instance*, *Micro Instance*, *High-Memory Instance* i *High-CPU Instance*.

5. OPIS APLIKATIVNOG RJEŠENJA

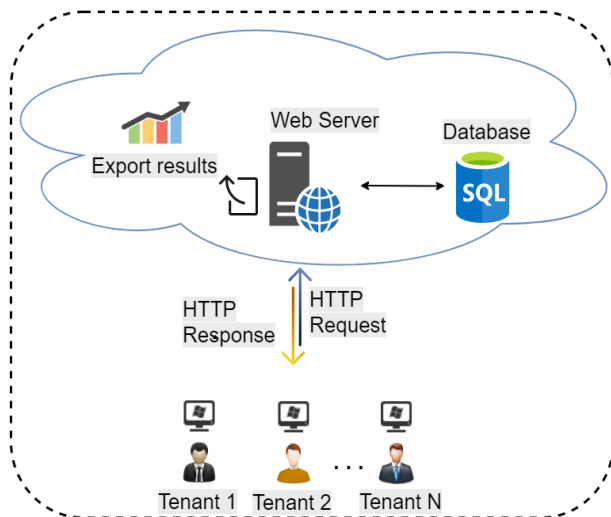
Na slici 1. prikazana je arhitektura trenutnog (multi-instance) softverskog rješenja.



Slika 1. Arhitektura trenutnog rješenja (*multi-instance*)

Sa slike se može vidjeti da postojanje N korisnika zahtjeva postojanje N instanci *web* servera kao i N instanci baze podataka. Migracija ovakvog rješenja na *Cloud* okruženje zahtjevala bi 2xN virtuelnih mašina uzimajući u obzir visoku dostupnost koju *Cloud* snabdjevači nude. Ovakav pristup predstavlja veoma skupo rješenje za ponuđače softverskih usluga. Iz tog

razloga, na slici 2. dat je prijedlog rješenja koji omogućava značajne uštede prilikom migracije na *Cloud*.



Slika 2. Arhitektura predloženog rješenja (multi-tenancy)

Upoređujući slike 1. i 2. može se uočiti da je broj instanci aplikacije redukovano sa N na samo jednu instancu koju će koristiti svi korisnici. Međutim, uvođenjem ovakvog pristupa javljaju se problemi koji nisu bili poznati do tada. Jedan takav problem jeste otežana funkcija naplate korišćenja usluga. Ovaj problem razmatran je u toku ovog rada i prijedlog njegovog rješenja biće dat u nastavku.

Sa slike 2. se vidi da se implementirani sistem sastoji od sljedećih komponenti: baza podataka, web server i klijentska web aplikacija. U bazi podataka nalaze se podaci o incidentima koji su se desili u elektroenergetskoj mreži. Kako bi se postojeća baza podataka prilagodila *multi-tenant* arhitekturi, bilo ju je neophodno proširiti sa informacijom o identifikatoru korisnika (*TenantId*). U cilju omogućavanja testiranja različitih pristupa izolacije podataka, prilikom implementacije baze korišćena su sljedeća dva pristupa: dijeljena baza - dijeljena šema i *row isolation* pristup. U prvom slučaju, prilikom korišćenja dijeljene baze sa dijeljenom šemom, za svakog korisnika kreirana je zasebna tabela, čiji naziv sadrži identifikator korisnika. Na taj način se za svakog korisnika, na osnovu njegovog identifikatora, određuje iz koje tabele će se čitati podaci. U drugom slučaju, prilikom korišćenja *row isolation* pristupa, svaku tabelu u postojećoj bazi bilo je potrebno proširiti sa novom kolonom, čije su vrijednosti sadržale identifikatore korisnika. Na osnovu identifikatora korisnika i na osnovu podataka iz novokreirane kolone, određuje se koji će se podaci čitati za koje korisnike.

Zadatak implementiranog web servera je isporučivanje informacija o incidentima, koje su smještene u bazi podataka. Prilikom svakog zahtjeva, web server vodi računa o identifikatoru korisnika, koji je dati zahtjev uputio. Kako bi web server imao informacije o svojim korisnicima, spisak svih registrovanih korisnika se čuva u bazi podataka, koju web server iščitava prilikom svoje inicijalizacije. Na osnovu tih informacija web server zna koji su registrovani korisnici, a koji ne. Ukoliko se neki korisnik ne nalazi u bazi registrovanih korisnika, a želi da dobije informacije o incidentima, to neće biti omogućeno. U suprotnom, na osnovu identifikatora korisnika, web

server dobavlja podatke iz baze specifične za datog korisnika, obrađuje ih u skladu sa dobijenim zahtjevom i vraća rezultat funkcije kao odgovor.

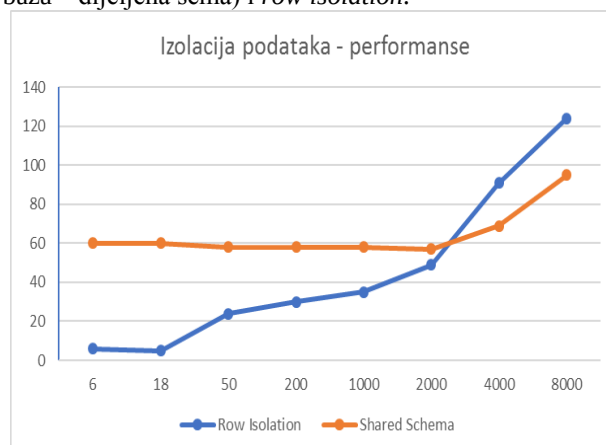
Da bi se omogućilo praćenje korišćenosti dijeljenih resursa, implementirano je softversko rješenje koje na web serveru prati aktivnosti korisnika. Prilikom implementacije ovog mehanizma korišćeno je aspektno orijentisano programiranje, u cilju brže i jednostavnije integracije sa postojećim softverskim rješenjima baziranim na *multi-tenant* arhitekturi. Sve što je potrebno da uradi proizvođač softvera koji želi da prati korišćenost resursa svoje aplikacije jeste, da na željenu metodu stavi atribut *TenantMonitoring*. Dodavanjem *TenantMonitoring* atributa u zaglavlje metode omogućava se praćenje korišćenosti resursa te metode. Za svakog registrovanog korisnika se prilikom pristupa sistemu bilježe informacije o njegovoj aktivnosti. Parametri koji se uzimaju u obzir su broj pristupa resursu i ukupno vrijeme njegove korišćenosti. Na osnovu ovih informacija i na zahtjev vlasnika aplikacije, generiše se izvještaj o korišćenosti resursa, čiji se rezultati čuvaju u datoteku u excel formatu.

U toku ovog rada napomenuto je da je jedan od glavnih problema migracije na SaaS model bezbjednost kritičnih podataka. Podaci o incidentima elektroenergetske mreže svakako predstavljaju kritične podatke, pri čemu bi migracijom na SaaS model bila ugrožena njihova bezbjednost. Metode zaštite softvera pomoću obfuskacije predstavljaju jedan od najvažnijih i najpouzdanijih načina fizičke zaštite softvera [8]. Zbog te činjenice, upravo je obfuskacija izabrana kao mehanizam za obezbjeđivanje bezbjednosti kritičnih podataka u *Cloud* okruženju.

6. EKSPERIMENTI

6.1. Tehnike izolacije podataka

U cilju odabira boljeg rješenja za izolaciju podataka u *multi-tenant* arhitekturi, u toku ovog poglavlja biće predstavljeni rezultati testiranja dva pristupa izolacije podataka. Testirani pristupi su: *shared schema* (dijeljena baza – dijeljena šema) i *row isolation*.



Slika 2. Poređenje performansi tehnika izolacije podataka

Na slici 3. dat je dijagram koji prikazuje rezultate korišćenja navedena dva pristupa. X-osa data na dijagramu predstavlja broj redova u tabeli, za koje je vršeno testiranje, dok Y-osa predstavlja ukupno vrijeme trajanja SQL upita izraženo u milisekundama. Sa grafikom se jasno može uočiti da *row isolation* predstavlja brži

pristup sve dok je broj redova u tabeli manji od 2000. S obzirom na činjenicu da se sve veća količina podataka skladišti u bazama, kao i činjenicu da *shared schema* pristup obezbijedi bolju izolaciju podataka, može se konstatovati da ovaj pristup predstavlja brže i sigurnije rješenje za proizvođače softverskih rješenja.

6.2. Mjerenje korišćenosti resursa

U cilju omogućavanja naplate korišćenja resursa, u nastavku će biti priložena dva pristupa određivanja njihove korišćenosti. Prvi pristup se bazira na mjerenju ukupnog broja pristupa resursima sistema, dok se drugim pristupom mjeri ukupno vrijeme korišćenosti resursa po korisniku. Prilikom testiranja sistema, u bazi registrovanih korisnika nalazili su se korisnici sa identifikatorima: *Tenant1*, *Tenant2*, *Tenant3* i *Tenant4*.

6.2.1. Mjerenje broja pristupa

Za sve registrovane korisnike uvećavana je vrijednost broja pristupa prilikom svakog njihovog pristupa sistemu. Rezultati ukupnog broja pristupa dati su u tabeli 1.

TENANT NAME	JAN	FEB	MAR
Tenant1	8,548.00	9,455.00	1,588.00
Tenant2	5,423.00	4,525.00	5,854.00
Tenant3	3,321.00	15,084.00	8,524.00
Tenant4	3,521.00	1,215.00	2,542.00

Tabela 1. Ukupan broj pristupa resursima sistema

Iz tabele se može vidjeti da se za svakog korisnika koji je pristupao resursima sistema, generiše poseban red u tabeli sa informacijama o ukupnom broju pristupa sistemu, za svaki mjesec tokom godine. Zbog bolje čitljivosti, u tabeli su prikazani samo rezultati za prva 3 mjeseca.

6.2.2. Mjerenje ukupnog vremena korišćenja

Kako bi se omogućilo što preciznije određivanje korišćenosti resursa, pored praćenja broja pristupa, implementirano softversko rješenje podržava i praćenje ukupnog vremena, za koje je sistem bio korišćen od strane pojedinačnih korisnika.

TENANT NAME	JAN	FEB	MAR
Tenant1	543.00	1,245.00	9,558.00
Tenant2	545.00	8,421.00	9,654.00
Tenant3	8,745.00	4,521.00	695.00
Tenant4	8,452.00	2,542.00	8,543.00

Tabela 2. Ukupno vrijeme korišćenja resursa sistema

U tabeli 2. prikazani su rezultati ukupnog vremena korišćenja resursa tokom godine, izraženi u minutima. Upoređujući rezultate iz tabele 1 i 2, može se uočiti da broj pristupa i ukupno vrijeme korišćenosti resursa nisu proporcionalni. To znači da bi vršenje naplate po samo jednom parametru bilo neadekvatno. Uvođenjem većeg broja parametara stvara se mogućnost za što preciznije određivanje korišćenosti resursa.

7. ZAKLJUČAK

U ovom radu predstavljeno je softversko rješenje za praćenje korišćenosti resursa u *multi-tenant* arhitekturi. Kao model isporuke izabran je SaaS model. Takođe, u radu su predstavljena dva pristupa izolacije podataka, dijeljena baza - dijeljena šema i *row isolation* pristup, i dato je obrazloženje izbora jednog od njih. Implementirano softversko rješenje za praćenje

korišćenosti resursa omogućava generisanje dva tipa izvještaja za vlasnike aplikacija. Prvi tip odnosi se na generisanje izvještaja o ukupnom broju pristupa resursu sistema, dok se drugi tip odnosi na ukupno vrijeme trajanja korišćenosti resursa.

Implementirano softversko rješenje moglo bi se iskoristiti za *trotting* mehanizam, što predstavlja prvi mogući pravac daljeg razvoja. Proširivanje logike za praćenje korišćenosti resursa po dodatnim parametrima, takođe predstavlja mogući pravac daljeg razvoja.

8. LITERATURA

- [1] Hanpin Wang, Zhao Jin, Lei Zhang, Yuxin Jing, Jiang Xu, Yongzhi Cao, "Reasoning about Cloud Storage Systems", IEEE Third International Conference on Data Science in Cyberspace, 2018.
- [2] Anthony Ayodele, Jia Rao, Terrance E. Bolt, "Towards Application-centric Fairness in Multi-Tenant Clouds with Adaptive CPU Sharing Model", IEEE 9th International Conference on Cloud Computing, 2016.
- [3] Cor-Paul Bezemer, Andy Zaidman, "Multi-tenant SaaS applications: maintenance dream or nightmare?", ACM 4th Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE), pp. 88–92, 2010.
- [4] Peter Mell, Timothy Grance, "The NIST definition of cloud computing", NIST special publication 800.145 7, pp. 1–3, 2011.
- [5] Keshav Gupta, Sandeep Kumar, Ojaswi Agnihotri, "Data isolation in multi-tenant SaaS environment", IEEE International Conference on Computing, Communication and Automation (ICCCA2016), 2016.
- [6] Gabriella Laatikainen, Arto Ojala, "SaaS architecture and pricing models", IEEE International Conference on Services Computing, 2014.
- [7] Xing Wu, Wu Zhang, Dou Wanchun, "Pricing as a Service: Personalized Pricing Strategy in Cloud Computing", IEEE 12th International Conference on Computer and Information Technology, 2012.
- [8] O.O. Shevtsova, D.N. Buintsev, "Methods and software for the program obfuscation", IEEE International Siberian Conference on Control and Communications, 2009.

Kratka biografija:



Radislav Zekanović je rođen 06.02.1994. godine u Loznici. Osnovnu školu "Vuk Karadžić" u Ročeviću završio je 2009. godine, poslije čega upisuje Srednju tehničku školu u Loznici. 2013. godine završava srednju školu i iste godine upisuje Fakultet tehničkih nauka u Novom Sadu, smjer Primijenjeno softversko inženjerstvo. Osnovne studije je završio 2017. godine, nakon čega je upisao master studije, smjer Primijenjeno softversko inženjerstvo.

РАЗВОЈ ПЕРСОНАЛНОГ СИСТЕМА ЗА ПРАЋЕЊЕ ЖИВОТНИХ ПАРАМЕТАРА PERSONAL HEALTHCARE MONITOR SYSTEM DEVELOPMENT

Бојан Торбица, *Факултет Техничких Наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – Рад се бави реализацијом хардверског прототипа уграђеног система који служи за мониторинг медицинског стања пацијената на кућном и клиничком лечењу. Уређај врши мониторинг откуцаја срца и квалитета ваздуха просторије у којој се пацијент тренутно налази. Реализована је и пратећа андроид апликација за прикупљање података генерисаних са уређаја.

Кључне речи: Уграђени системи, Ардуино, Интернет Ствари, Сензори, Андроид

Abstract – This paper presents a developed embedded system prototype used for health monitoring for patients that undergo home, or clinical treatment. The device measures heart rate and determines the air quality of the room the patient is currently in. An accompanying android application is developed which is used to gather generated data from the device.

Keywords: Embedded systems, Arduino, Internet of Things, Sensors, Android

1. УВОД

У последњих неколико година, развојем нових микропроцесорских технологија које у стопу прати развој нових архитектура тзв. System on chip (скраћено SoC) и PC (енгл. Personal Computer) решења, способности и процесна моћ микрочипова расте смањењем њихове саме величине као и минимализација уређаја на којима се исти налазе (овде се пре свега мисли на SoC решења).

За свега неколико деценија, уређаји који су заузимали читаве просторије, данас стају у длан шаке и имају неколико реда величина већу процесну моћ.

Велика већина микропроцесора се производи као интегрална компонента неког од тзв. уграђених система (енгл. Embedded Systems) као самосталан чип, или део тзв. микроконтролера који представља интегрисану процесорску јединицу са меморијским интерфејсом и другим потребним периферијама.

Микроконтролери су специфично дизајнирани за коришћење у уграђеним системима и веома често су део SoC система.

Уграђени системи су природни продужетак развоја микропроцесора представљајући минималистичке уређаје, чија својства осликавају намену: минимални захтеви напајања, мали габарити, а уз све то и мини-

мална цена потребна како за производњу тако и за пласирање на тржиште.

Као део техничко-технолошког напретка друштва, фокус развоја се помера према интегрисању и међуповезивању људи и уређаја путем интернета. Ова промена названа је интернет ствари (енгл. Internet of Things, скраћено IoT).

Циљ овог рада је анализа могућности и представа израде прототипа уграђеног система чија је главна намена мониторинг пацијената на кућном, или клиничком лечењу. Овај уређај мери откуцаје срца и даје процену квалитета ваздуха просторије у којој се пацијент налази: температура, влажност ваздуха, атмосферски притисак и сл.

2. УГРАЂЕНИ СИСТЕМИ И ЕКОСИСТЕМ ИНТЕРНЕТА СТВАРИ

Уграђени систем, по дефиницији је хардверско-софтверска комбинација уз евентуалне додатне механичке, или друге делове, који је дизајниран да изврши предодређену функцију. У неким случајевима уграђени уређаји су део већег система, као што је то на пример ABS у возилима. Разлог из ког се овакав систем зове уграђен је проста чињеница да је, као што се да закључити из понуђене дефиниције, уграђен као део комплетног уређаја заједно са другим чиниоцима у виду механичких, или других делова.

Обзиром на то да уграђени системи извршавају специфичну функцију, односно дизајнирају се како би извршавали ту функцију, инжењери могу да дизајнирају целокупни систем тако да оптимизују разне параметре у производњи система. Ти параметри могу бити везани за саму величину уређаја, имплементационе детаље управљачког софтвера, цена израде, или комбинација ових параметара.

Након 60-их година 20. века и појаве првих уграђених система, процесна моћ ових система се развила експоненцијално. Готово обрнуто пропорционално процесној моћи, цена уграђених система се смањила па је самим тим и њихова приступачност постала већа. Као што је споменуто у уводу овог рада, појавом микроконтролера крајем 70-их и почетком 80-их година 20. века настаје највећи помак у развоју уграђених система, услед чињенице да се сада меморијски и улазно-излазни модули налазе на једном чипу, далеко олакшавајући и чинећи бржим поступак прототиповања и развоја електронских уређаја.

Прва особа којој се приписује креирање термина: интернет ствари, је Кевин Ештон [1] 1999. године. Ештон аргументује да је аспект интеракције људи са стварима унутар физичког света потребно суштински

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био др Драган Иветић, ред. проф.

променити услед напретка у рачунарским, комуникационим и технологијама везаним за генерисање и прикупљање велике количине података који се могу наћи у паметним уређајима. Ештонов рад на тзв. Auto-ID пројекту у лабораторији института за технологију у Масачусетсу је допринео распрострањивању и широј примени RFID технологије, што је изградило солидну основу за тренутну визију IoT екосистема.

Једна од градивних карактеристика интернета ствари је тзв. памет (енгл. Smartness). Приписивањем ове карактеристике екосистему интернета ствари, прави се јасна дистинкција између сличних концепата као што је на пример мрежа сензора. Овој карактеристици се могу припојити додатни термини те створити још ближе њену одредницу као тзв. мрежна памет, или боље речено паметна мрежа. Паметна мрежа представља комуникациону инфраструктуру коју карактеришу следеће функционалности:

- Стандардизација и отвореност коришћених комуникационих стандарда и протокола. Ово укључује слојеве интеракције уређаја са физичким светом као и слојеве комуникације са интернетом.
- Способност адресирања уређаја, односно додељивање директне IP (енгл. Internet Protocol) адресе и мултифункционалности, односно могућности да уређај једне намене буде употребљен за другу намену са минимално, или без преправки.

Стопа генерисања дигиталних података расте свакодневно, а са њима и разлози за бригу о приватности који се спрежу са безбедношћу тих података. По својој природи и намени, сензори као уређаји су способни за прикупљање података о нпр. корисничкој локацији, откуцајима срца и покретима. Приватност тако генерисаних података мора бити загарантована и у потпуности у контроли корисника. У дистрибуираним окружењима као што је то IoT, приватност се може остварити или путем централизованог приступа, или тако што сваки ентитет у систему управља својим подацима, односно тзв. приватност-по-дизајну [2] (енгл. Privacy-by-design) приступ.

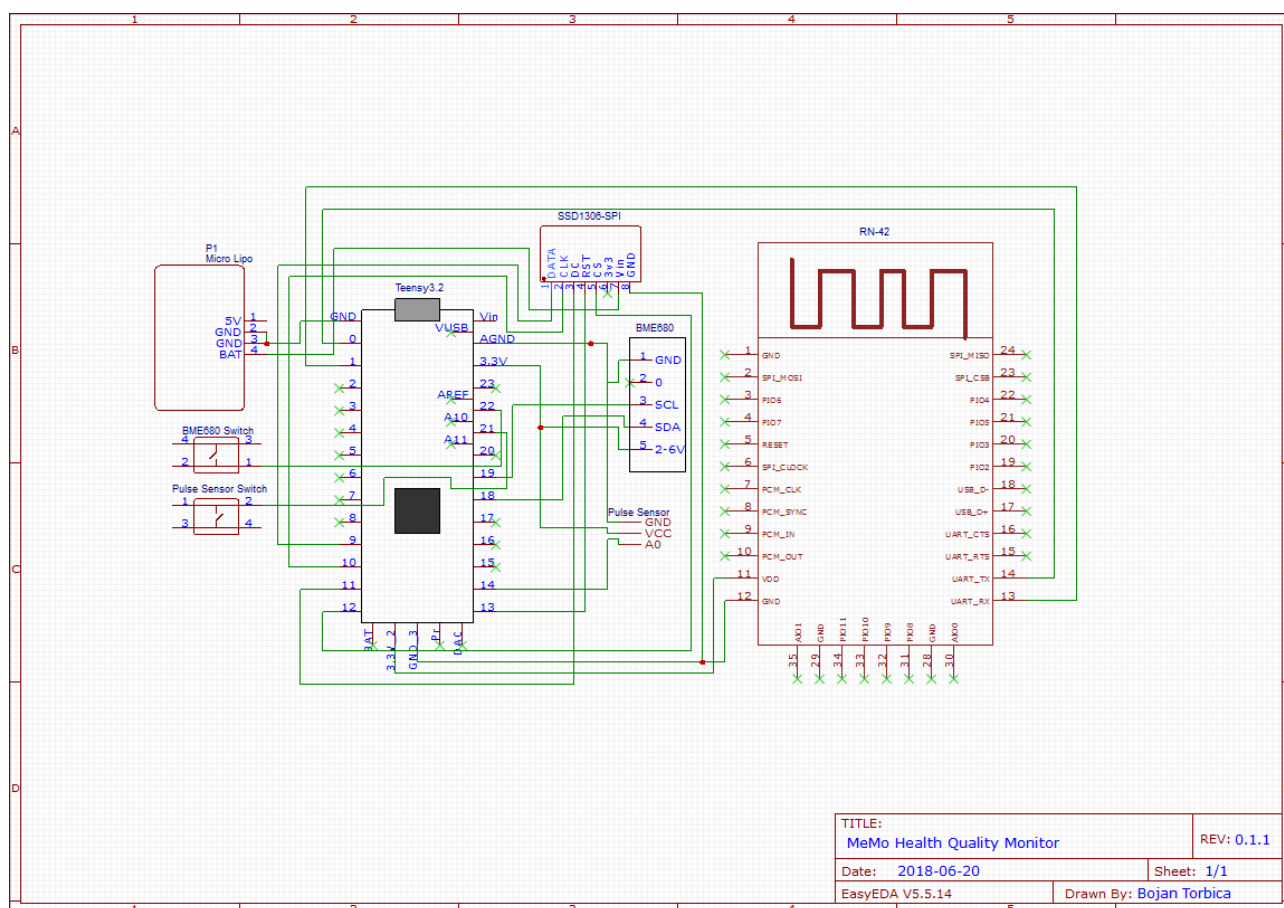
3. РЕАЛИЗАЦИЈА ПРОТОТИПА УГРАЂЕНОГ УРЕЂАЈА

Главна мотивација која стоји иза овог рада је анализа могућности реализације једноставног физичког, носивог уграђеног система који би служио као медицинско помагало особама на кућном лечењу којима је потребно одржавање одређеног степена квалитета ваздуха и при том пратећи здравствено стање пацијента који носи уређај. Систем би био повезан на мобилни телефон пацијента, или било који други паметни уређај са способношћу блутут комуникације. Тако повезан, уређај би слао податке генерисане својим сензорима на паметни уређај. Дужност паметног телефона, односно апликације која комуницира са уграђеним системом за мониторинг, би у овом односу била једноставно прикупљање генерисаних информација са система и реаговање по потреби, а по задатим параметрима. Реаговање

апликације би се састојало у провери прикупљених параметара и њиховом поређењу са постављеном референтном вредношћу. У случају одступања од референтних вредности, систем би био задужен да, путем интернета обавести о томе надлежно особље медицинске установе која брине о пацијенту. Овај пројекат је у складу са наменом назван „MeMo“ што је скраћено за Медицински Монитор.

Као мозак целокупног система употребљен је тинси микроконтролер, односно хардверска платформа. Тинси је ардуино-компатибилна развојна плоча. Обзиром на природу и намену система, употребљена су два типа сензора. Први, сензор квалитета ваздуха који може да мери температуру просторије, влажност ваздуха, атмосферски притисак, детектује присуство тзв. испарљивих органских једињења и даје процену надморске висине на којој се тренутно налази. Други тип сензора употребљен у овом раду је сензор пулса. Поред сензора, систем се састоји још и од дисплеја типа OLED (енгл. Organic Light-Emitting Diode), блутут модула, литијум-полимер батерије и пуњача за исту који је интегрисан у цео систем. Пулс сензор има три прикључка које је потребно повезати: напајање, уземљење и аналогни излаз. Обзиром на то да тинси микроконтролер има предодређени улаз за уземљење аналогних прикључака, природно је повезати одговарајући прикључак са пулс сензора на њега. Напајање се може прикључити на било који од тинси 3.3V прикључака. Преко аналогног излаза пулс сензора пристижу подаци на тинси микроконтролер па је исти потребно повезати на један од аналогних улаза тинси плоче. У овом пројекту, повезан је A0 аналогни прикључак на тинси микроконтролеру који је уједно и дигитални прикључак 14. BME680 сензор квалитета ваздуха коришћен у овом систему, има четири обавезна прикључка и један који се не повезује. Своју комуникацију овај сензор одвија преко I²C магистрале и као такав, два његова прикључка, SDA и SCL је потребно спојити на одговарајуће прикључке на тинси микроконтролеру. За тинси микроконтролер ови прикључци су дигитални улаз 18 за SDA и дигитални улаз 19 за SCL. У општем случају, ове дигиталне прикључке је потребно поставити у режим рада за I²C комуникацију. SSD1306 OLED екран комуницира преко SPI, а развојна плоча открива неколико прикључака који морају бити повезани: линија података (DATA), линија синхронизације (CLK), линија за разлучивање података од команди (DC), линија ресетовања (RST), линија за селекцију чипа (CS), линија напајања (Vin) и линија уземљења (GND). Линија назначена 3.3 остаје неповезана. Све линије имају одговарајуће улазе на тинси микроконтролеру:

- DATA – дигитални улаз 9.
- CLK – дигитални улаз 10.
- DC – дигитални улаз 11.
- RST – дигитални улаз 13 који је уједно и уграђена светлећа диода.
- CS – дигитални улаз 12 на тинси контролеру



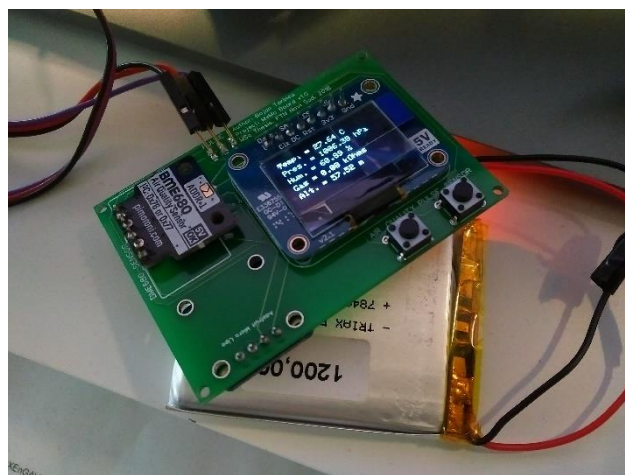
Слика 1. Шематски приказ повезивања компоненти

- Vin – Vin прикључак на тинсију, односно напајање које тинси добија са батерије у случају да је повезан на батерију, а у случају повезивања на прото-плату, потребно је повезати SSD1306 Vin прикључак на један од два 3.3V прикључка на тинсију.
- GND – истоимени прикључак на тинсију, GND.

Блутут модул коришћен у овом раду, RN-42, није имао никакву развојну плату већ је коришћен као SMD компонента. Од мноштва изложених прикључака, за потребе овог пројекта од значаја су била само четири: напајање (VDD), уземљење (GND), предајник (UART_TX) и пријемник (UART_RX). Предајник и пријемник морају се повезати на одговарајуће улазе на тинси микроконтролеру и то: UART_TX на дигитални улаз 0, односно RX и UART_RX на дигитални улаз 1, односно TX на тинси развојном систему. Као део финалног дизајна, за извор напајања МеМо уграђеног система одабрана је литијум-полимер батерија, а самим тим је било потребно одабрати и пуњач за исту. Пуњач који је одабран за овај пројекат на свом излазу даје напон, или од батерије, или од микро USB-а који служи за пуњење батерије. На слици 1 је приказана финална шема уређаја.

Након завршетка прото-дизајна и тестирања компоненти, било је потребно на неки начин „спаковати“ све компоненте у једну зарад лакшег преноса и коришћења. Као природно решење овог проблема дизајнирана је штампана плоча која обједињује све елементе овог пројекта у један и чини прави уграђени систем. За потребе дизајна штампаних плоча постоји мноштво алата, бесплатних и лиценцираних. За

потребе овог пројекта коришћен је бесплатан онлајн алат за дизајн штампаних плоча и интегрисаних кола, EasyEDA [3]. На слици 2 је приказан завршен уређај.



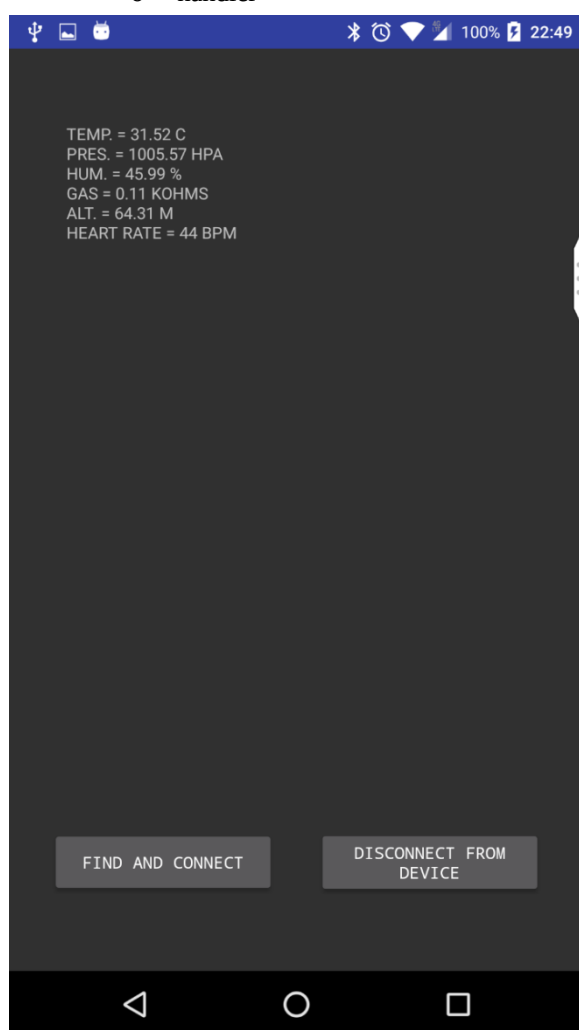
Слика 2. Приказ састављеног уређаја у режиму рада

Штампана плоча се састоји из два слоја. На првом слоју се налазе сензор квалитета ваздуха, дисплеј, конектори за сензор пулса као и два дугмета која углавном служе за дебаговање система. На другом, доњем слоју налази се тинси, блутут модул и пуњач за литијум-полимер батерију на чијем излазу се преко батерије добија напајање за тинси, а последично и за цео систем. На крају, све компоненте су постављене на своје место и залемљене за плочу.

Како би уређај суштински могао да има општу употребу, потребно је тинси микроконтролер заменити у потпуности, тако да се искористи само микропроцесор и линије које се извуку из њега. Наравно, ово свакако важи и за остале делове система. Предност овога би била мања цена израде и смањење самог уређаја. Друга предност, што је свакако и један од главних фактора за унапређење, би било лакше додавање нових компоненти, а примарно других, прецизнијих сензора.

Пратећа андроид апликација сама по себи је веома једноставна: структура је подељена у два модула и један подмодул. Хијерархијска структура ових компоненти је такође веома једноставна и организована је на следећи начин:

- app
 - handler



Слика 3. Приказ рада андроид апликације која кориснику на приказ даје податке генерисане од стране МеМо уређаја

Ова апликација, као што је већ било напоменуто, а за потребе овог рада служи једноставној сврси: да се повеже на реализован уређај и прикаже податке генерисане са њега. Наравно, у реалним условима у којима је циљано да се ова апликација и уређај користе, једноставан приказ података, суштински није довољан. Поред приказа података, за сценарио општег

коришћења, потребно је реализовати сервер који би служио за прикупљање и обраду података у реалном времену. Овај сервер би садржао логику која би била јединствена за сваког пацијента, у смислу потребе за мониторингом стања, а одговарајуће медицинско особље би имало приступ овим подацима у било ком тренутку како би благовремено могли да реагују на било какве промене које су сумњивог порекла. На слици 3 је приказан главни екран андроид апликације.

4. ЗАКЉУЧАК

Уграђен систем реализован у овом раду представља један од уређаја са потенцијалном реалном применом у медицинском свету. Пацијенти на кућном, или клиничком лечењу, или рехабилитацији могу имати користи од уређаја који би им давао константни приказ квалитета њиховог здравственог стања током опоравка, или лечења. Пацијенти би сами могли да прате дневну рутину у рехабилитационом смислу и сами реагују на евентуалне промене у околини, обзиром на то да ће уређај те промене приказати. Такође, додељено медицинско особље може да прати развој и процес оправка својих пацијената.

На крају, аутор овог рада се нада да ће сам рад као и презентовани уређај и андроид апликација, макар на нивоу идеје и концепта видети неку форму примене.

5. ЛИТЕРАТУРА

- [1] Ashton Kevin. „That ‘internet of things’ thing.“ RFID journal, pp. 97-114, Јул 2009.
- [2] [22] Tuhin Borgohain, Uday Kumar and Sugata Sanyal. „Survey of security and privacy issues of Internet of Things.“, arXiv Prepr. arXiv1501.02211; 2015.
- [3] <https://easyseda.com> (приступљено у августу 2018)

Кратка биографија:



Бојан Торбица рођен је у Горњем Милановцу, 1992. год. Основне студије завршио је на Факултету Техничких Наука 2015. године из области Рачунарска Графика. Мастер рад на Факултету техничких наука из области Електротехника и Рачунарство – Рачунарство и Аутоматика одбранио је 2018. год.

RELEJNA ZAŠTITA I PREDIKCIJA INCIDENATA U SISTEMU ZA UPRAVLJANJE OTKAZIMA**RELAY PROTECTION AND INCIDENT PREDICTION IN OUTAGE MANAGEMENT SYSTEM**

Miloš Marković, *Fakultet tehničkih nauka, Novi Sad*

Oblast - ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U okviru ovog rada objašnjen je uloga relejne zaštite u predikciji incidenata u sistemu za upravljanje otkazima kao i njihov uticaj na rad distributivnog sistema. Relejom zaštitom štite se elementi elektroenergetskog sistema, ali njihovim reagovanjem deo potrošača ostaje bez napajanja. Mehanizam predikcije otkaza omogućava brzo i precizno detektovanje elemenata koji su odreagovali usled kvara u mreži, pa samim tim omogućava i brže otklanjanje samog kvara.

Ključne reči: *Relejna zaštita, upravljanje otkazima*

Abstract – This document provides explanation of relay protection and its role for incident prediction in outage management system and their influence on functionality of distribution system. Relay protection provides protection of distribution system elements, but their protection actions causes outages for some customers. Prediction mechanism provides fast and precise detection of protection elements that reacted due to fault in network, thus providing faster elimination of that fault.

Keywords: *Relay protection, outage management*

1. UVOD

Jedan od glavnih zadataka distributivnih preduzeća jeste pružanje kvalitetne usluge potrošačima. Pod kvalitetnom uslugom se podrazumeva pouzdano, sigurno i efikasno snabdevanje potrošača električnom energijom. Pošto su distributive mreže jako složene i sadrže veliki broj elemenata, javljaju se otkazi, odnosno kvarovi tih elemenata.

Otkazi elemenata u velikoj meri utiču na indekse pouzdanosti distributivnih mreža. Da bi se smanjilo vreme trajanja prekida napajanja, ublažio uticaj otkaza, odnosno da bi se povećala pouzdanost, potrebno je što brže i efikasnije rešiti probleme koje uzrokuju otkazi. S tim u vezi, kreiran je sistem za upravljanje otkazima, odnosno OMS (engl. Outage management system).

Dispečeri se u distributivnim preduzećima susreću sa ne tako malim brojem otkaza, koji moraju biti analizirani i otklonjeni u najkraćem mogućem roku, a u analizi, detektovanju kao i u brzini otklanjanja otkaza veliku ulogu ima mehanizam za predikciju otkaza, tačnije njegova funkcija za predikciju incidenata.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Duško Bekut, red.prof.

Nakon uvoda, u drugoj glavi rada date su teorijske osnove sistema za upravljanje otkazima. Objašnjena uloga OMS-a i uloga njegovih osnovnih jedinica.

U trećoj glavi rada, obrađen je mehanizam za predikciju incidenata i relejna zaštita. U prvom delu je predstavljena relejna zaštita, njena funkcija kao i osnovne podele. U drugom delu je predstavljena funkcija predikcije kao osnovni deo mehanizma predikcije, kao i njeni ulazni podaci i izlazni rezultati, a zatim je objašnjen pojam i tok analize otkaza.

U četvrtoj glavi je predstavljena simulacija predikcije. Nakon zaključka iznetog u šestoj glavi, navedena je literatura, korišćena za realizaciju ovog rada.

2. TEORIJSKE OSNOVE

OMS omogućavaju efikasnu identifikaciju i rešavanje otkaza kao i kreiranje korisnih izveštaja. Takođe, ova komponenta omogućava distributivnim preduzećima da informišu potrošače o trenutnom stanju otkaza, problemima koji su uzrokovali otkaz kao i da informišu potrošače o restauraciji napajanja.

OMS se koristi za dispečovanje svih planiranih i ne planiranih radova u distributivnim mrežama u šta spadaju otklanjanje otkaza, prekidačke operacije, rešavanje rizika, održavanje kao i sve ostale poslove neophodne za normalno dnevno funkcionisanje distributivne mreže [1,2].

Sistem za upravljanje otkazima se može podeliti u nekoliko logičkih jedinica, a to su: jedinica za upravljanje incidentima, jedinica za upravljanje podacima potrošača i jedinica za upravljanje pozivima.

2.1. Jedinica za upravljanje incidentima

Jedinica za upravljanje incidentima (engl. Incident Management) je glavna jedinica komponente za upravljanje otkazima, odgovorna da pomaže sistemskim dispečerima u efikasnom dispečovanju otkaza (aktivnih i pasivnih), njihovih problema kao i svih ostalih planiranih radova. Poremećaji zajedno sa informacijama neophodnim za izvršavanje radova su grupisani u objekte nazvane Incidenti (incidents).

Otkazi elemenata ne uzrokuju uvek i prekid snabdevanja električnom energijom. S tim u vezi razlikuju se aktivni i pasivni otkazi elemenata. Pod aktivnim se podrazumevju oni otkazi koji pobuđuju relejnu zaštitu i koji deluje na prekidače, tako da njihovim delovanjem dođe do prekida snabdevanja. Prekidi napajanja mogu biti trajni ili trenutni (prolazni), pa se tako trenutni prekidi obično rešavaju delovanjem automatskog ponovnog uključenja (APU) ili

manuelnim isključenjem i ponovnim uključenjem nekog prekidača [3].

Na osnovu ove podele otkaza, tip Incident može biti:

- Ne planirani aktivni incident (Unplanned outage Incident).
- Pasivni incident (Non-outage incident).
- Planiran aktivni incident (Planned outage incident).
- Planirani rad (Planned Work).

Kao što je gore prethodno navedeno, otkaz može biti trenutni ili prolazan (momentary) i može biti trajan (sustained) pa tako pod tip incidenta može biti:

- Trajni (Sustained) – incident gde je potreban izlazak radnika na teren.
- Trenutni (Momentary) – incident gde nije potrebno dispečovati radnike.

2.2. Jedinica za upravljanje potrošačkim podacima

Ova jedinica je odgovorna za skladištenje podataka o svim potrošačima unutar sistema za upravljanje distribucijom (DMS), kao i za obezbeđivanje podataka na korišćenje drugim jedinicama sistema za upravljanje otkazima kome su ti podaci potrebni.

2.3. Jedinica za upravljanje pozivima

Ova jedinica je odgovorna za snimanje i upravljanje pozivima vezanim za probleme. Ona prikazuje stanje trenutno aktivnih poziva potrošača, kao i stanje trenutnih povratnih poziva ka potrošaču u realnom vremenu. Ova jedinica sistema za upravljanje otkazima je konstruisana tako da može da skladišti i obrađuje veliki broj poziva potrošača, a da pritom ne dođe do smanjenja sistemskih performansi sistema za upravljanje otkazima.

3. RELEJNA ZAŠTITA I PREDIKCIJA OTKAZA

Za postizanje sigurnog i pouzdanog funkcionisanja elektroenergetskog sistema, potrebno je obezbediti zaštitu svakog njegovog elementa (generatora, transformatora, voda, motora, itd.), kao i pojedinih celina u okviru sistema od kvarova. Upravo se ta zaštita celokupnog elektroenergetskog sistema od kvarova ostvaruje relejnom zaštitom [4].

Osnovni cilj primene relejne zaštite je najbrže moguće isključenje elementa, odnosno dela elektroenergetskog sistema prilikom nekog abnormalnog stanja, inicijalizacijom odgovarajućih upravljačkih akcija, uz očuvanje funkcionalnosti i održanje stabilnosti ostalog dela sistema. Releji svojim delovanjem moraju izolovati samo deo mreže pod kvarom, omogućavajući nastavak rada nad delom mreže bez kvara [5]. Delovanjem releja na zaštitne uređaje (reklozeri, prekidači) izoluju se mesta sa kvarom, ali samim tim i prekida snabdevanje potrošača energijom. Da bi dispečer najbolje procenio mesto kvara, jako je bitno prvo odrediti koji je uređaj delovao usled kvara. Zbog slabe pokrivenosti zaštitnih uređaja SCADA sistemom, potrošački pozivi su glavni vid obaveštenja u slučaju prekida snabdevanja električnom energijom. Da bi se te informacije najbolje iskoristile u cilju lociranja uređaja koji je delovao usled kvara, kreiran je mehanizam za predikciju otkaza.

Mehanizam za predikciju otkaza predstavlja funkcionalnost OMS-a koja je odgovorna za detektovanje zaštitnih elemenata u mreži koji su delovali usled kvara,

na osnovu ulaznih podataka i trenutne mrežne topologije. Ovaj mehanizam se posmatra kao alat koji pomaže operatoru da upravlja incidentima u mreži, posebno aktivnim incidentima.

Funkcija predikcije, kao osnovni deo mehanizma predikcije, izvršava pretragu (tracing) do uzvodnih elemenata mreže koji su najverovatniji uzročnici prekida napajanja. Tokom pretrage, funkcija razmatra različite podatke, a koji se mogu konfigurisati u zavisnosti od potreba distributivnih preduzeća.

3.1. Relejna zaštita u distributivnim mrežama

Relej je uređaj koji služi za detekciju abnormalnih pogonskih stanja elemenata i delova sistema i na osnovu toga se vrši inicijalizacija odgovarajućih upravljačkih akcija za obezbeđenje normalnog pogona.

Pod abnormalnim pogonskim stanjem se podrazumeva stanje sa kvarom ili opasno pogonsko stanje. To su stanja u kojima parametri (napon, struja, frekvencija, trajanje nekog režima, itd.) izlaze iz okvira stanja predviđenih projektima i konstrukcijom elektroenergetskog sistema [5].

Osnovni zadatak relejne zaštite je najbrže moguće isključenje elementa sa kvarom, bez narušavanja funkcionalnosti i stabilnosti ostatka sistema. Pri ispunjenju ovog cilja potrebno je ispuniti nekoliko opštih zahteva, a to su: selektivnost, brzina reagovanja, osetljivost, pouzdanost, sigurnost i ekonomičnost.

Releji se mogu podeliti na osnovu dva kriterijuma: po tehnologiji koja se koristi i po tipu ulazne veličine. Po tehnologiji koja se koristi, releji se dele na elektromehaničke, statičke i mikroprocesorske. Prema vrsti ulazne veličine releji se mogu podeliti na: strujne, naponske, frekventne, releje snage, impedantne, termičke, mehaničke.

3.2. Automatsko ponovno uključenje voda (APU)

Prilikom pojave kvara u mreži, iskustva su pokazala da isključenjem dela nadzemnih mreže sa kvarom i njegovim ponovnim uključenjem, šanse da dođe do spontane eliminacije kvara su velike. Naime, istraživanja su pokazala da čak 70% do 80% kvarova čine prolazni kvarovi odnosno trenutni kvarovi.

Reagovanjem zaštitnog uređaja, odnosno otvaranjem prekidača, kvarovi ovog tipa se otklanjaju, a ponovnim zatvaranjem napajanje potrošača se ponovo uspostavlja i to u najkraćem roku, pa samim tim gubitci usled neisporučene energije su minimalizovani.

Ukoliko se nakon ponovnog zatvaranja prekidača kvar ne otkloni, ponovljenim postupkom se otklanja još 3% do 5% kvarova, dok u trećem pokušaju svega 1% kvarova. Prednost ovakvog rada releja je u tome što potošači retko kad mogu da prime prekid u napajanju [5].

Kratkotrajni prekid napajanja, kao posledica otklanjanja trenutnog kvara, se kod potrošača može manifestovati kao iznenadno treptanje svetla (engl. flick light). Potrošački pozivi koji za svoj razlog imaju treptanje svetla, iako malobrojni, imaju veliki značaj prilikom predikcije otkaza, jer pokreću funkciju predikcije sa *flick light* semantikom koja omogućava prediktovanje trenutnih, aktivnih incidenata u OMS sistemu.

3.3. Osnovni zadatak, podaci i rezultati funkcije predikcije

Glavni zadatak predikcije jeste da prediktuje incident od problemskog poziva ili AMI događaja (AMI engl. Advanced Meter Infrastructure) do uzvodnog zaštitnog uređaja i da podrži logiku analize otkaza koja je implemetirana na jedinicu za upravljanje incidentima. Klasična predikcija je zasnovana na modelu pretrage (engl. tracing), ali napredna, pored pretrage uzima u obzir i status telemtrisanih zaštitnih uređaja i merenja. Uzimajući u obzir telemtrisanu opremu, predikcija dobija na brzini i preciznosti, pa će tako u mrežama sa velikim brojem telemtrisanih tačaka predikciona funkcija dati bolje i preciznije rezultate.

Ulazni, statički podaci neophodni za izvršavanje funkcije predikcije jesu model mreže i opcije za izvršavanje. Pored opcija i modela, glavni ulazni podaci jesu i pozivi potrošača, AMI događaji ili senzori otkaza, odnosno informacije sadržane u njima, prosledjene OMS-u.

Kao rezultat, funkcija predikcije vraća Incident sa setom rezultata. Rezultati incidenta su drugačiji za različite tipove otkaza odnosno incidenata, pa tako za aktivne incidente, rezultati su sledeći: prvi prediktovani zaštitni uređaj, drugi prediktovani zaštitni uređaj, lista i broj afektovanih potrošača ispod prvog zaštitnog uređaja, broj afektovanih potrošača ispod drugog zaštitnog uređaja, lista afektovane zaštitne opreme, lista odgovarajućih poziva, lista afektovanih fidera, broj zaštitnih uređaja ispod drugog zaštitnog uređaja.

Za pasivne incidente, tačnije za incidente nastale kao posledica problema sa naponom, rezultati su sledeći: lista naponski regulatora, lista afektovanih potrošača, lista afektovane opreme, lista poziva.

3.4. Logika predikcije

Kod predikcije, funkcijom se izvršava uzvodna pretraga ulaznih elemenata (koji mogu biti brojila potrošača (engl. SDP) koji su prijavili problem) sve do korena mreže. U svakoj iteraciji se proverava postojanje zaštitne opreme, sloj po sloj. Napredna predikcija proverava napon, struju i snagu temetrisanih merenja tokom uzvodne pretrage, da bi se pronašao izvor energizacije na korenu kojim se napaja potrošač sa kvarom. Ukoliko merenje na putanji uzvodne pretrage nema vrednosti u normalnom opsegu (npr. vrednost napona blizu nule), to znači da je put deenergizovan i da je kvar zapravo iznad potrošača, odnosno da je odreagovala neka uzvodna zaštitna oprema. Ukoliko postoji više merenja, čije vrednosti pokazuju deenergizaciju, uvažavaju se merenja najbliža korenu, a prvi uzvodni zaštitni uređaj od tog merenja je uzet kao kao kandidat predikcije. Ukoliko sva merenja iznad potrošača imaju dobre vrednosti, kvar se onda desio na upravo kod potrošača. Pored telemtrisanih merenja, funkcija proverava za svaki telemtrisani zaštitni uređaj kvalitet statusnog SCADA signala. Ako je kvalitet signala dobar, taj element se isključuje kao kandidat predikcije, a ako je kvalitet SCADA signala loš ili sumnjiv, taj element se uzima kao kandidat predikcije.

Ukoliko na distributivnoj mreži ne postoje telemtrisani uređaji, funkcija će kao kandidata predikcije predložiti zaštitni uređaj najbliži potrošaču kojim je prijavljen problem kao što je to slučaj kod klasične predikcije.

3.5. Analiza otkaza

Analiza otkaza predstavlja složen proces koji obuhvata prihvatanje ulaznih parametara, kreiranje odgovarajućih incidenata na osnovu ulaznih parametara (poziva ili dospelih AMI događaja), pokretanje funkcije predikcije kao i odlučivanje o tome koji će od prediktovanih kandidata biti uzet kao elemenat na kome se dogodio otkaz, u zavisnosti od lokacije, broja poziva i drugih incidenata na sistemu, kao i od niza konfigurabilnih opcija definisanih od strane distributivnih preduzeća.

Kod standardnih *outage* scenarija, gde broj potrošačkih poziva nije veliki, pozivi ili AMI događaji se prosleđuju jedan po jedan ka predikciji, tako da se za jedan potrošački poziv kreira jedan odgovarajući incident. U slučaju da je potrošač prijavio problem na lokaciji koja je već afektovana nekim ranije kreiranim incidentom, ukoliko je to moguće, obaviće se spajanje tog potrošačkog poziva sa već postojećim incidentom. Ukoliko nije moguće upariti poziv sa već postojećim incidentom, kreira se novi odgovarajući incident. Nakon što je potrošački poziv prošao obradu i odgovarajući incident kreiran, pokreće se funkcija predikcije otkaza i rezultati predikcije se predaju incidentu. Dalje se proverava mogućnost za podizanje (engl. roll up) incidenta, odnosno prvog prediktovanog uređaja na neki uzvodni zaštitni uređaj. Mehanizam koji to omogućava jeste *Roll Up* predikcija.

Roll Up predikcija predstavlja sastavni deo mehanizma predikcije. Njen zadatak je da na osnovu podešenja funkcije, a u zavisnosti od uslova postavljenih od strane distributivnih preduzeća, vrši ažuriranje tih incidenata, odnosno repozicioniranje incidenata na uzvodni zaštitni uređaj. Postoje dva osnovna tipa *roll up*-a, manualni i automatski. Manualno premeštanje incidenta na sistemu izvršava operator lično, u slučaju da nije zadovoljan rezultatima predikcije, ili ukoliko je nakon već izvršene analize otkaza dobio nove informacije koje se tiču incidenta. Automatski *roll up* predstavlja sastavni deo analize otkaza. On se pokreće ukoliko su uslovi, definisani od strane distributivnog preduzeća, za njegovo pokretanje ispunjeni. Nakon što je roll up izvršen, vrši se spajanje zaostalih nizvodnih incidenata sa uzvodnim, *roll up*-ovanim incidentom.

Automatska roll up predikcija se izvršava:

- Ukoliko je broj problemskih poziva ili AMI događaja pod istom servisnim transformatorom jednak ili veći broju definisanom u opcijama od strane dispečera.
- Ukoliko je procenat afektovanih zaštitnih uređaja ispod zajedničkog drugog zaštinog uređaja veći ili jednak procentu definisanom u opcijama od strane dispečera.

Distributivna preduzeća definišu gore pomenute opcije prema karakteristikama mreže koju pokrivaju. Pod karakteristikama mreže koje utiču na uslove predikcije se podrazumeva, konfiguracija, broj potrošača, broj zaštitnih uređaja, broj telemtrisanih uređaja kao i učestanost kvarova i problema sa naponom.

Provera prvog uslova *roll up* predikcije se izvršava tako što se prvo uzima broj potrošačkih poziva i AMI događaja ispod prvog zaštinog uređaja sa različitih servisnih

lokacija. Zatim se uzima broj potrošačkih poziva i AMI događaja iz opcija, potrebnih za podizanje incidenta na prvi uzvodni zaštitni uređaj, a na kraju se proverava da li je broj poziva ili AMI-ja jednak ili veći od onog definisanog u opcijama.

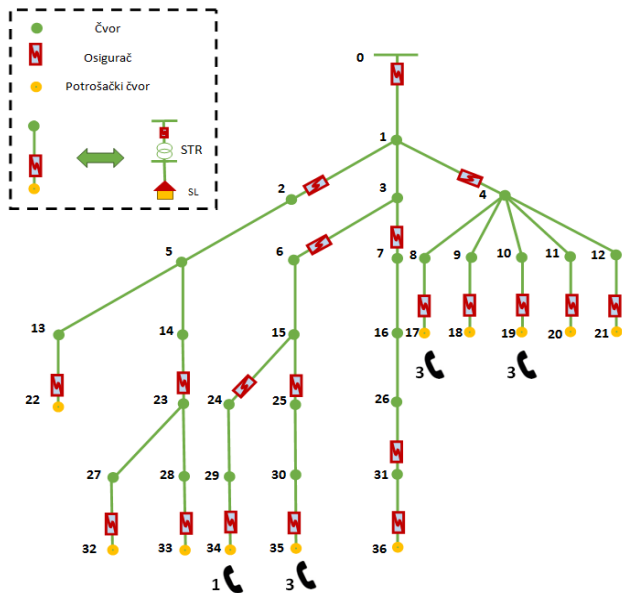
Ukoliko prvi uslov *roll up* predikcije nije ispunjen, incident će ostati na potrošačkom nivou. Kada je uslov ispunjen, odnosno, kada je dovoljan broj poziva ili AMI događaja prijavljen, tada će incident se biti prenet na prvi uzvodni zaštitni uređaj, a incidenti koji su ostali na potrošačkom nivou ispod *roll up*-ovanog incidenta biće spojeni u taj incident.

Provera drugog uslova *roll up* predikcije se izvršava tako što se prvo grupišu incidenti koji imaju zajednički drugi zaštitni uređaj. Zatim se za jednu grupu vrši prebrojavanje incidenata, odnosno, zaštitnih uređaja na kome je trenutno prediktovan incident i uzima se ukupan broj zaštitnih uređaja ispod drugog zaštitnog uređaja za tu grupu. Na kraju se izračunava procenat afektovanih uređaja ispod zajedničkog zaštitnog uređaja za tu grupu i poredi se sa procentom definisanom u opcijama.

Ukoliko je drugi uslov ispunjen, biće kreiran novi incident koji će biti prediktovan na tom drugom zajedničkom zaštitnom uređaju. Incidenti koji se nalaze ispod novokreiranog incidenta se spajaju u taj incident. Ceo postupak analize otkaza biće ponovljen svaki put pri prijemu novih ulaznih podataka.

4. SIMULACIJA PREDIKCIJE

Na slici 4.1 data je jednostavna distributivna mreža na kojoj će se simulirati rad predikcije.



Slika 4.1 – Jednostavna distributivna mreža

Broj prijavljenih poziva je 10, pozive su prijavili potrošači na čvorovima 17 (3 poziva), 19 (3 poziva), 35 (3 poziva) i 34 (1 poziv). Razlog svih poziva je *No Power*. Distributivno preduzeće je definisalo u opcijama da je broj problemskih poziva potrebnih za *roll up* na prvi zaštitni uređaj jednak 3, a procenat afektovanih uređaja ispod zajedničkog drugog zaštitnog uređaja je 30%. Prateći logiku analize otkaza, nakon izvršenja predikcije na mreži je aktivno 3 incidenta.

Prvi incident je *roll up*-ovan na osigurač 4 (osigurač iznad čvora 4). Taj *roll up* je omogućen zbog ispunjenja prvo prvog, a potom i drugog uslova *auto roll up*-a. Pošto je broj poziva na čvorovima 17 i 19 jednak 3, ispunjen je uslov za *auto roll up* od servisne lokacije do osigurača servisnog transformatora. Nakon toga, zajednički drugi zaštitni uređaj za oba incidenta je osigurač 4. Pošto je broj afektovanih zaštitnih uređaja ispod osigurača 4 jednak 2, a ukupan broj uređaja 5, to znači da je afektovano više od 30% uređaja, pa su ispunjeni uslovi za *roll up* na osigurač 4. Incident na čvoru 35 je *roll up*-ovan na osigurač 25. Pošto je broj poziva na čvoru 35 jednak 3, ispunjen je uslov da se uradi *roll up* na osigurač 6 servisnog transformatora. Pošto na mreži ne postoji incident čiji je drugi prediktovani uređaj isti kao i kod ovog incidenta, nije ispunjen drugi uslov za *auto roll up*. Sledeći incident je kreiran na servisnoj lokaciji 34. Pošto je broj poziva na čvoru 34 jednak 1, nije ispunjen prvi uslov za *roll up*, pa samim tim ni drugi. Ovde je najverovatnije problem kod potrošača.

5. ZAKLJUČAK

Funkcija za predikciju otkaza (incidenata) je oslonjena u značajnoj meri na primenu relejne zaštite i u mnogome olakšava posao distributivnim preduzećima, jer smanjuje vreme trajanja otkaza u mreži, povećava pouzdanost, pa se samim tim i troškovi usled neisporučene energije smanjuju. Operatori ne moraju dugo da analiziraju pozive dobijene od potrošača, već veći deo analize prepuštaju predikcionoj funkciji, koja sa velikom pouzdanošću i brzinom daje precizne rezultate.

Problemi sa funkcijom predikcije mogu se javiti u slučajevima lošeg modela mreže, kao i tokom scenarija sa olujom, gde se u vremenskom rasponu od samo nekoliko sati može prijaviti par hiljada i čak desetina hiljada potrošačkih poziva. U tom slučaju, performanse funkcije mogu opasti, a kao posledica toga, vreme izvršavanja predikcije će se povećati.

6. LITERATURA

- [1] <http://www.powersystem.org/outage-management-system-oms>
- [2] <http://www.puc.state.nh.us/2008IceStorm/ST&E%20Presentations/Unitil%20OMS%20Presentation%2006-09-09.pdf>
- [3] Miroslav D. Nimrihter, *Elektrodistributivni sistemi*, FTN Novi Sad 2009
- [4] Duško Bekut, *Relejna zaštita*, Fakultet Tehničkih Nauka Novi Sad, 2009.
- [5] Kristijan Huska, *Koordinacija relejne zaštite u radijalnim i slabo upetljenim distributivnim mrežama*, FTN Novi Sad 2017

Kratka biografija:



Miloš Marković rođen je u Šapcu 1992. god. Diplomirao je na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Elektroenergetski sistemi 2016. godine i nakon toga upisao master studije na istom fakultetu.

ХИДРАУЛИЧКА АНАЛИЗА ОДВОЂЕЊА УПОТРЕБЉЕНИХ ВОДА НАСЕЉА ПЕЋИНЦИ, СУБОТИШТЕ И ПРХОВО СА ТЕХНО-ЕКОНОМСКИМ ЕЛЕМЕНТИМА**HYDRAULIC ANALYSIS OF DRAINAGE OF USED WATERS OF SETTLEMENT PEĆINCI, SUBOTIŠTE AND PRHOVO WITH TECHNO-ECONOMIC ELEMENTS**

Данијел Малешевић, Матија Стипић, *Факултет техничких наука, Нови Сад*

Област - ГРАЂЕВИНАРСТВО

Кратак садржај – У оквиру Мастер рада дата је хидрауличка анализа канализационе мреже отпадних вода насеља Пећинци, Суботиште и Прхово. За моделовање канализационог система коришћен је програмски пакет EPA SWMM који омогућава брз, итеративни поступак пројектовања. Такође, техно-економском анализом узете су у обзир две варијанте за избор локације пречистача отпадних вода, од којих је након техно-економске анализе у самом раду усвојена и разрађена економски прихватљивија.

Кључне речи – хидрауличка анализа, канализациона мрежа, пречистач отпадних вода, техно-економска анализа

Abstract – Within Master thesis, it is given hydraulic analysis of the sewage network of the wastewater of the settlement of Pećinci, Subotičte and Prhovo. For the modeling of the sewer system, the EPAS WMM software package was used which enables a fast, iterative design process. Also, the techno-economic analysis took into account two variants for the selection of the location of wastewater treatment plants, of which, after the techno-economic analysis, the work itself was adopted and worked out more economically acceptable.

Keywords – hydraulic analysis, sewage network, WWTP(wastewater treatment plant), techno-economic analysis

1. УВОД

Свега 33% подручја АП Војводине покривено је са канализационом мрежом. На преосталих 67% отпадна вода домаћинства и индустрије се одводи директно у водоток, или у септичке јаме из којих се отпадна вода процеђује у подземље.

Велики део изграђене канализационе мреже нема пречистач отпадне воде, па се сва отпадна вода излива директно у водотокове и загађују их, док постоје и насеља у којима је изграђен пречистач отпадне воде али је ван употребе. Изградњом канализационе мреже, штити се животна средина (биљни и животињски свет), смањује потенцијална опасност од заразе и побољшава стање екосистема насеља.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био доц. др Матија Стипић.

Самим тим се на предметном подручју планира изградити фекална канализација за сва насеља општине Пећинци, а у самом раду биће обухваћена три насеља: Пећинци, Суботиште и Прхово.

Како се скупљене отпадне воде из предметних насеља не би директно упуштале у најближи ефлуент, неопходно је предвидети и постројење за пречишћавање отпадних вода.

2. ОПШТЕ КАРАКТЕРИСТИКЕ ПРЕДМЕТНОГ ПОДРУЧЈА

Општина Пећинци налази се у Доњем Срему, близу реке Саве, јужно од ауто-пута Београд-Загреб. Општина се граничи са општинама Рума, Стара Пазова, Земун, Обреновац и Владимировци и карактерише је периферан положај у односу на подручје АП Војводине.

Територија Општине Пећинци заузима површину од 488,65km², и по величини се налази на 18 месту у АП Војводини, а чине је 15 катастарских општина и 15 насељених места.

Насеља која се налазе у Општини су: Ашања, Брестач, Деч, Доњи Товарник, Карловчић, Купиново, Обреж, Огар, Пећинци, Попинци, Прхово, Сибач, Сремски Михаљевци, Суботиште и Шимановци.

Генерално, за сва насеља општине Пећинци, па самим тим у насељима Пећинци, Суботиште и Прхово уочен је пад броја становника. Како би се перманентан пад броја становника неутралисао, односно, до повећања броја становника могуће је доћи једино коренитим променама у свим насељима и подизањем њихових стандарда на виши ниво.

3. МОДЕЛИРАЊЕ ТЕЧЕЊА У КАНАЛИЗАЦИОНИМ СИСТЕМИМА

Код канализационих система за употребљене(отпадне) воде, промена дотока је спора па се они успешно могу третирали без узимања у обзир неусталених појава.

У раду се приказује једна могућност математичког моделирања односно рачунарске симулације течења воде у канализационим системима. Течење отпадне воде одвија се гравитационо, тако да једначине које ће описивати течење воде у канализационим цевима се обично описују једначинама одржања масе и количине кретања за линијско течење воде у отвореном каналу (тзв. Сен-Венанове једначине) [2]:

$$\frac{\partial A}{\partial t} + \frac{\partial Q}{\partial x} = 0 \quad (1)$$

$$\frac{\partial Q}{\partial t} + \frac{\partial}{\partial x} \left(\beta \frac{Q^2}{A} \right) + gA \frac{\partial z}{\partial x} + gAS_f = 0 \quad (2)$$

У мирном режиму течења потребни су по један гранични услов (ниво, протицај или крива протока) на узводном и низводном крају, а у бурном режиму задају се два гранична услова на узводном крају и један на низводном [2].

Горе поменуте једначине називају се и моделом динамичког таласа или комплетни динамички талас. Изостављање појединих чланова омогућава једноставније и ефикасније нумеричко решавање, али се тиме губи на тачности добијеног решења.

4. ПРОГРАМСКИ ПАКЕТ EPASWMM 5.1.

Примена софтверског пакета EPA SWMM у пројектовању канализације за атмосферску и употребљену воду омогућује комплексно сагледавање ситуације на терену узимајући у обзир распоред и карактеристике урбанизованих површина, распоред корисника, хидролошке податке о кишама и отицајима и течење воде по површини терена, као и у канализационој мрежи.

Осим пројектовања канализационих система, EPA SWMM омогућује проверу постојећих канализационих система и анализу њихових перформанси што може бити искоришћено при одржавању постојећих система или рехабилитацији и проширењу истих.

Течење у мрежи се програмом може симулирати помоћу модела устаљеног течења, кинематског и динамичког таласа, док за прорачун инфилтрације постоје три модела.

4.1. Могућности програма

Програм EPA SWMM користи се за хидрауличко моделирање које се примењује за прорачун течења у цевима и каналима, кроз резервоаре или објекте за расподелу протицаја.

SWMM програм садржи многобројне флексибилне хидрауличке могућности моделирања које се користе за усмеравање атмосферске воде, фекалне отпадне воде и спољашњих дотока кроз дренажну мрежу цеви, канала, погона за чување/третирање и обилазних структура.

Ту спадају и следеће могућности:

- рад са мрежама неограничене величине;
- коришћење разноврсних стандардних облика отворених и затворених канала као и природних канала;
- моделирање посебних елемената као што су јединице за чување/третман, разделници тока, пумпе, преливи и пригушнице;
- коришћење спољашњих протока и унетих података о квалитету воде из површинских вода, подземног међудотока, директног отицаја, санитарног

протока по сувом времену и дотока који одреди корисник;

- примена метода усмеравања протока помоћу кинематског или динамичког таласа;
- моделирање разних режима протока као што су повратна водена маса, преоптерећење, обрнут проток и висина круне прелива.

Приликом димензионисања канализационе мреже насеља Пећинци, Суботиште и Прхово, улазни параметри који одређују ток прорачуна морају бити унешени.

Уз помоћ унешених параметара, модел(канализациона мрежа) почиње анализу/моделирање и као излаз, односно, резултат добија се проток у мрежи, брзине по деоницама, квалитет отпадне воде,...

У улазне параметре спадају:

- број становника;
- анализирани(пројектни) период;
- потрошња, односно, количина употребљене воде насеља, индустрије, јавних објеката, инфилтрација стране воде;
- врста индустрије;
- коефицијенти неравномерности становништва, индустрије и јавних објеката;
- тип и пречник усвојене цеви у насељу, односно, храпавост цеви;
- дубине полагања цеви уз услов поштовања минималних и максималних ограничења;
- нагиб цеви.

5. ХИДРАУЛИЧКИ ПРОРАЧУН

Хидраулички прорачун рађен је за свако насеље понаособ.

У зависности од дијаграма часовне неравномерности који су усвојени за предметна насеља, представљају неравномерност потрошње отпадне воде, па самим тим и неравномерност отицања из насеља знатно се разликује у одређеним деловима дана [3].

Самим тим, дат је хидраулички прорачун потрошње воде, који је потом унешен у програмски пакет EPA SWMM које резултира брзинама, протицајима и испуњености цевима у сваком часу времена трајања симулације.

Будући да отпадна вода отиче канализационом мрежом гравитацијом, на појединим местима да не би дошло до превеликог укопавања канализационих цеви, раде се црпне станице.

Хидрауличком прорачуну претходила је обрада подлога, односно припрема улазних података за симулацију течења у систему колектора. Ова припрема је урађена коришћењем савремених поступака, при чему се имало у виду да тачност при уношењу података директно утиче на добијене резултате.

За пројектни период од 25 година, уз познавање броја становника, врсте и броја домаћих животиња, врсте индустрије, типа потрошача једним именом, коеф. неравномерности и специфичне потрошње појединих категорија потрошача, долази се до количине воде

која бива искориштена у насељу коју је потребно одвести до постројења за пречишћавање отпадне воде [1].

Усвојена специфична дневна потрошња (планирана потрошња) воде по становнику на дан $q_{spec} = 130$ l/st/dan [4].

$$Q_{sr,dn} = q_{spec} * N \quad (3)$$

N - укупан број потрошача

$$Q_{max,dn} = Q_{sr,dn} * k_d \quad (4)$$

k_d – коефицијент дневне неравномерности

Пошто коришћени софтверски пакет омогућава прорачун нестационарног течења у цевоводима и симулацију дневног протока (за 24 часа), израчунате количине средњег дневног протока деоница су се “упуштале” у колекторе умножене са коефицијентима часовних неравномерности k_h [4].

За насеља Пећинци, Прхово и Суботиште дијаграми часовне неравномерности убачени су у хидраулички прорачун програмском пакета EPA SWMM.

$$Q_{max,h} = Q_{sr,dn} * k_h \quad (5)$$

Специфична деонична количина отпадне воде:

$$Q_{spec,deon} = \frac{Q_{max,dn,ukup}}{\Sigma L} \quad (6)$$

ΣL – укупна дужина канализационе мреже

Чворно оптерећење:

$$Q_{cvor,i} = Q_{spec,deon} * \Sigma L_i \quad (7)$$

ΣL_i – низводна деоница од чвора

Чворно оптерећење задато је у сваком чвору хидрауличког модела као базна потрошња.

6. ТЕХНО-ЕКОНОМСКА АНАЛИЗА ВАРИЈАНТИ ПОСТРОЈЕЊА ПРЕЧИСТАЧА ОТПАДНЕ ВОДЕ

Приликом пројектовања канализационе мреже насеља Пећинци, Суботиште и Прхово, две варијанте биле су узете у обзир ради поређења.

Прва варијанта представља три пречишћавања отпадних вода, за свако насеље по једно у његовој непосредној близини.

Друга варијанта представља један заједнички пречишћач отпадних вода које се налази у близини насеља Пећинци, на које би била прикључена сва три насеља.

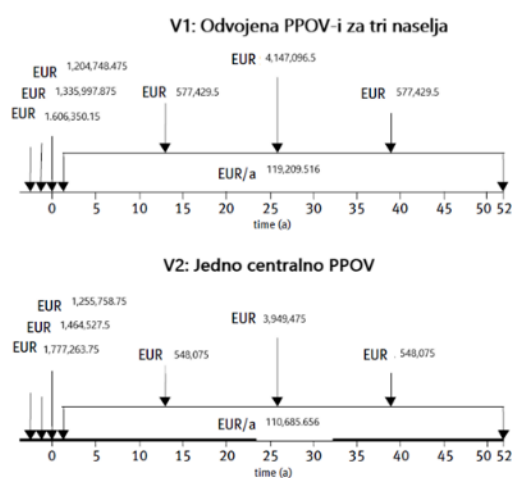
Упутства за рад за израду динамичких израчунавања поређења трошкова за пројектно планирање у управљању урбаним водама, рађено је по препоруци Немачког удружења за воду, отпадне воде и отпад (DWA-Deutsche Vereinigung für Wasserwirtschaft, Abwasser und Abfall) [5]. Вршена је анализа инвестиционих вредности опреме, као и експлоатационе вредности опреме током година рада. У обе варијанте постројења за пречишћавање, неопходно је дефинисати инвестиционе вредности.

Инвестициона вредност самог постројења за пречишћавање отпадне воде, пумпних станица, канализације износи 4147096,5eur за варијанту I, односно 4497550,0eur за варијанту II.

Експлоатациона вредност истих елемената износи 119209,5eur/a за варијанту I, односно 110685,6eur/a за варијанту II.

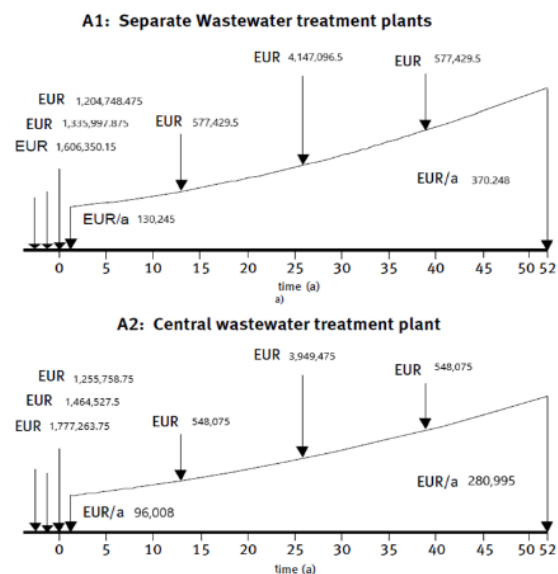
Период анализе од 52 године узет је као меродаван за анализу. Будући да инвестициони период траје 3 године, различите каматне стопе биће узете у обзир током трајања инвестиционог периода, као и експлоатационог периода.

На слици 1, приказани су инвестициони трошкови, реинвестициони трошкови, као и експлоатациони трошкови за обе варијанте, без повишица цена, а распоређени по годинама.



Слика 1. Трошкови пројекта без стварне повишице цена [5]

На основу годишње реалне стопе раста цене, узима се у обзир и стварно повећање цене експлоатационих вредности. Слика 2 представља график промене експлоатационих вредности током анализираног периода.



Слика 2. Серија трошкова, са разматрањем стварног повећања цене експлоатационих вредности [5]

Упоређујући трошкове пројекта садашњих вредности, као и трошкове вредности пројекта, долази се до закључка да је просечна годишња уштеда трошкова 2400eur, на страни варијанте II. Уколико се узме у обзир годишњи реални раст цена, долази се до просечне годишње уштеде од 3100eur, такође на страни варијанте II.

Овим је доказано да је варијанта II, односно решење са једним централним постројењем за пречишћавање отпадне воде исплативије.

7. ПОТИСНИ ВОД

Будући да се техно-економском анализом дошло до варијанте која представља решење са једним централизованим постројењем за пречишћавање отпадне воде у близини насеља Пећинци са потисним цевоводима од насеља Суботиште и Прхово, неопходно је отпадну воду насеља Суботиште и Прхово довести до постројења. То ће се учинити помоћу потисног вода од полиетиленских цеви, и изабраним врстама и капацитетима пумпи које се налазе у црпним станицама.

Код канализације са пумпним станицама потребан притисак се обезбеђује радом пумпи у пумпним станицама. Пумпа повећава енергију флуиду и омогућава да доспе до крајњег корисника(ППОВ). Канализациони системи са пумпањем се примењују често у равничарским пределима са изразито малом денивелацијом терена као што насеља Пећинци, Суботиште и Прхово. Пумпна станица добија енергију најчешће из електричних мрежа и ту енергију претвара у механичку посредством мотора, најчешће електромотора.

Сам одабир пумпи вршен је у програмском пакету Wilo EMU Select, у којем су такође изабране врсте и карактеристике пумпи које су коришћене за денивелације остварених цевовода у самим насељима.

8. ЗАКЉУЧАК

Циљ мастер рада јесте побољшање животног стандарда предметних насеља, квалитета подземних вода и животне средине драстичним смањењем броја септичких јама.

На територији Општине Пећинци за изградњу канализационе мреже фекалних вода, усвајају се цеви од тврдог поливинил хлорида (PVC) због низа предности и повољних карактеристика у односу на друге материјале. Пречници PVC цеви које су коришћене су од 250mm до 400mm.

Друга врста цеви које су коришћене јесу цеви од полиетилена високе густине(PEHD), чија намена је искоришћена за потискивање фекалне воде у самим насељима, од излаза црпне станице, па до највише тачке деонице која одводи гравитационо воду ка излазу из насеља.

Потисни вод од насеља Прхово ка ППОВ-у које се налази недалеко од насеља Пећинци, дугачак је 2500m, са усвојеном цеву HDPE100 – OD110 SDR11(S-5) PN16.

Потисни вод од насеља Суботиште ка ППОВ-у које се налази недалеко од насеља Пећинци, дугачак је 3300m, са усвојеном цеву HDPE80 – OD110 SDR22(S-10.5) PN10.

Након урађене хидрауличке анализе сваког насеља, приступило се анализи поређења варијанти које су представљале решење са једним централним пречистачем отпадних вода које се налази код насеља Пећинци, или три мања пречистача, за свако насеље посебно.

Техно-економска анализа показала је да је оптималније решење са једним централним пречистачем отпадних вода које се налази код насеља Пећинци.

Будући да се ради о једном заједничком пречистачу отпадних вода за сва три насеља, од насеља Прхово и Суботиште отпадна вода је потисним цевоводом одведена до пречистача. У том случају су такође коришћене цеви од полиетилена високе густине (PEHD) спољашњег пречника 110mm.

Доминантни пречник цеви у канализационој мрежи је Ø250mm у сва три насеља, и то је уједно и најмањи пречник. Овим је задовољен услов за минималним пречником у сепарационој канализационој мрежи. Проценат испуњености канализационих цеви је на свакој деоници испод 66%, како би се вршило оваздушење (проветравање) цевовода. У раду су димензионисани основни објекти канализационе мреже.

9. ЛИТЕРАТУРА

[1] Писана предавања: доц. др Матија Стипић, Комунална хидротехника, Факултет техничких наука у Новом Саду, 2016

[2] Хајдин Георгије: Механика флуида књига прва Увођење у хидраулику, Грађевински факултет, Београд, 2002

[3] Милојевић Милоје, Снабдевање водом и канализација насеља, Научна књига, Београд, 1987

[4] German rules and standards- STANDARD ATV-DVWK-A 134E

[5] DWA-Deutsche Vereinigung für Wasserwirtschaft, Abwasser und Abfall

Кратка биографија:



Данијел Малешевић рођен је у Новом Саду 1993. године. Дипломски рад из области грађевинарства – комунална хидротехника одбранио је 2017. године на Факултету техничких наука у Новом Саду. Мастер рад из области комуналне хидротехнике – хидрауличка анализа одвођења употребљених вода одбранио је на истом факултету 2018. године.



Матија Стипић рођен је у Сомбору 1964. године. Докторирао је на Факултету техничких наука 2009. год., а од 2011. има звање доцента. Област интересовања су му хидраулика и комунална хидротехника.

ОЗАКОЊЕЊЕ ОБЈЕКТА У РЕПУБЛИЦИ СРБИЈИ – ПРИМЕРИ ОЗАКОЊЕЊА ОБЈЕКТА НА ПУТНОЈ МРЕЖИ**LEGALIZATION OF OBJECTS IN THE REPUBLIC OF SERBIA - EXAMPLES OF LEGALIZATION OF OBJECTS ON THE ROAD NETWORK**

Ивана Богнар, Небојша Радовић, *Факултет техничких наука, Нови Сад*

Област- ГРАЂЕВИНАРСТВО

Кратак садржај – У овом раду је описан проблем, поступак и решење озакоњења објекта применом Закона о озакоњењу објеката који је Скупштина Србије усвојила 20. новембра 2015. године и који је донет у јавном интересу, како би се највећи број незаконито изграђених објеката увео у законске токове по најповољнијим условима, у једноставној и брзој процедури. Приложен је приказ и анализа постојеће законске и техничке регулативе за озакоњење објеката као и примери из праксе који се односе на објекте на путној мрежи као што су саобраћајни прикључци и мостови.

Кључне речи: озакоњење објеката путне мреже, легализација, саобраћајни прикључци

Abstract – This paper describes the problems, procedure and solution of the legalization of the objects, in accordance with the Law on the Legalization of Objects, adopted by the Assembly of Serbia on November 20, 2015, and adopted in the public interest, in order to introduce the largest number of illegally constructed objects into legal flows for the most favorable conditions, in a simple and fast procedure. In this paper are attached presentation and analysis of the existing legal and technical regulations for the legalization of objects, as well as examples from the practice related to the objects on the road network, such as traffic junctions and bridges.

Keywords: legalization of road network objects, traffic junctions

1. УВОД

Проблем бесправне градње постоји уназад неколико деценија, али је нарочито ескалирао 90-их година прошлог века, када се бележи масовна бесправна градња, која је узрокована великим приливом становништва са ратом захваћених подручја, недостатком сваке врсте контроле над бесправном градњом, али и непостојањем планске документације која би омогућила легалну градњу. У „Службеном гласнику РС“, број 96/2015 од 26. новембра 2015. године објављен је дуго очекивани Закон о озакоњењу објеката [1], који је ступио на снагу наредног дана 27. новембра 2015. године, што се правда циљем који се

жели подстићи, односно јавним интересом у овој области која оправдава доношење и примену овог Закона по хитном поступку.

2. ОПИС ПРОЦЕСА ОЗАКОЊЕЊА ОБЈЕКТА У РЕПУБЛИЦИ СРБИЈИ СА ПОСЕБНИМ ОСВРТОМ НА ОБЈЕКТЕ ПУТНЕ МРЕЖЕ

Да би се започео процес озакоњења било ког објекта у Републици Србији потребно је да Покрајински грађевински инспектор изда Решење о рушењу. По Закону о озакоњењу објеката ("Службени гласник РС", број 96. од 26. новембра 2015. године) [1], Решење о рушењу замењује Захтев за легализацију тј., Пријаву незаконито изграђеног објекта.

2.1. Услови за озакоњење

Услови за озакоњење објеката су:

- Надлежност
- Прецизирање захтева
- Усклађеност са Плански актом
- Доказ о праву својине
- Извештај о затеченом стању са Елаборатом геодетских радова
- Сагласност од управљача јавног добра, организацији надлежној за заштиту природних, односно културних добара
- Уплата таксе за озакоњење
- Припрема и издавање Решења о озакоњењу
- Допис управном суду о достави уверења да ли је покренут управни спор против решења
- Достава правоснажног решења служби за катастар непокретности
- Достава правоснажног решења Покрајинској грађевинској инспекцији
- Упис права својине

3. ПРИКАЗ И АНАЛИЗА ПОСТОЈЕЋЕ ЗАКОНСКЕ И ТЕХНИЧКЕ РЕГУЛАТИВЕ ЗА ОЗАКОЊЕЊЕ ОБЈЕКТА

У Сектору за грађевинске послове, спровођење обједињене процедуре и озакоњење обављају се послови који се односе на:

- издавање локацијских услова;
- издавање грађевинске дозволе;
- издавање решења из члана 145. Закона о планирању и изградњи;
- пријаву радова;
- издавање употребне дозволе;

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био в.проф. др Небојша Радовић.

- прибављање услова за пројектовање; односно прикључење објеката на инфраструктурну мрежу;
- прибављање исправа и других докумената које издају имаоци јавних овлашћења а услов су за изградњу објеката, односно за издавање локацијских услова, грађевинске и употребне дозволе из надлежности Министарства;
- као и обезбеђење услова за прикључење на инфраструктурну мрежу и за упис права својине на изграђеном објекту.

ЗАКОНИ који се примењују у процесу озакоњења:

- **Закон о планирању и изградњи** [3]

Грађевинску дозволу за изградњу објеката издаје Министарство надлежно за послове грађевинарства (у даљем тексту: Министарство), ако овим законом није другачије одређено.

- **Закон о општем управном поступку** [4]

Управни поступак јесте скуп правила која државни органи и организације, органи и организације покрајинске аутономије и органи и организације јединица локалне самоуправе, установе, јавна предузећа, посебни органи преко којих се остварује регулаторна функција и правна и физичка лица којима су поверена јавна овлашћења примењују када поступају у управним стварима.

- **Закон о републичким административним таксама** [5]

Овим законом уређују се републичке административне таксе. За списе и радње у управним стварима, као и за друге списе и радње код органа, плаћају се таксе по одредбама овог закона.

- **Закон о државном премеру и катастру** [2]

Овим законом уређују се стручни послови и послови државне управе који се односе на државни премер, катастар непокретности, катастар водова, основне геодетске радове, адресни регистар, топографско-картографску делатност, процену вредности непокретности, геодетско-катастарски информациони систем и Националну инфраструктуру геопросторних података и геодетски радови у инжењерско-техничким областима.

4. ОПИС МЕТОДА ЗА ПРОЦЕНУ ВРЕДНОСТИ ОБЈЕКТА НА ПУТНОЈ МРЕЖИ [6]

4.1. Масовна процена вредности

Масовна процена је дефинисана као системска процена групе непокретности на одеђени дан коришћењем стандардизованих метода и статичког тестирања. Оно што масовној процени даје велику предност је што покрива целину територије једне државе. Тачност масовне процене зависи од улазно-излазних података (података са тржишта за анализе и катастарских података на основу којих се приписују вредности), као и од детаљности модела, односно фактора који фигуришу у њему.

Систем масовне процене обухвата:

- Прикупљање података;
- Анализа тржишта непокретности;

- Апроксимацију тржишних сила кроз факторе у математичком моделу;
- Преписивање вредности свих непокретности на основу утврђеног модела.

4.2. Процена вредности и управљање путном мрежом

Главна сврха процене вредности путне мреже је оцена ефикасности активности управљања и одржавања путевима, путем израженог стварног стања мреже. Идеја прорачуна је да се покаже да ли држава „губи“ или „не губи“ своју путну мрежу и, ако је то тачно, да се процени ред величине и брзине те појаве. Ако је држава свесна да постоји значајан губитак путне имовине, могуће је остварити промене актуелне политике ради бољег очувања путева чиме би циљ био задовољен.

Путна мрежа генерално представља највећу имовину у делу јавне инфраструктуре. Управљање путном имовином се дефинише као систематски процес економичног одржавања, унапређења и експлоатације физичке имовине.

Процена вредности претпоставља одређивање монетарне вредности те имовине. Вредност се обично обрачунава кроз рачуноводствено књиговодство и исказује у билансу стања финансијских извештаја на крају године.

Постоји више метода за процену вредности путне мреже (нпр. као књиговодствена вредност, тржишна вредност, цена замене, садашња вредност, номинална вредност, реална вредност, опорезива вредност или употребна вредност).

4.2.1 Методологија

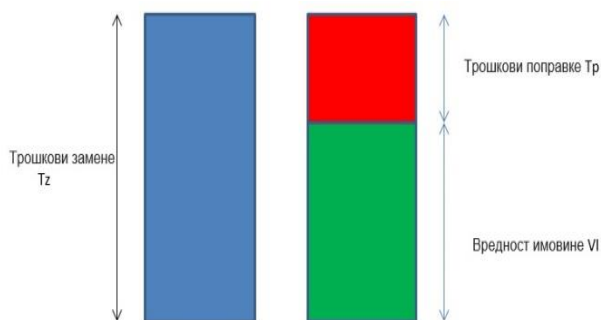
Идеална ситуација приликом процене вредности путне имовине је да на располагању постоји комплетан инвентар свих елемената путне конструкције, као што су косине, коловозна конструкција, мостови, тунели, мали објекти, опрема пута и друго, са забележеном вредности инвестиције/изградње и утврђеним тренутним стањем.

То су потпуно исте информације које се користе и приликом доношења политике или појединих активности у вези са путевима, за дефинисање планских и програмских шема.

Вредност пута који није нов се срачунава умањењем вредности коју би исти имао када би био нов (вредност замене или максимална теоријска вредност) за величину трошкова довођења тог пута из постојећег стања, које је слабије услед неког степена пропадања, у веома добро стање (слика 1.).

Веома добро стање одражава вредност новог пута оног дана када је пуштен у саобраћај.

Вредност која се одузима представља онај трошак уклањања оштећења које пут може трпети и још увек бити у проходном стању.



Слика 1. Концепт процене вредности [6]

4.2.2. Вредност путне мреже

Путна инфраструктура подлеже техничким доказима, али понекад није укључена у рачуноводствене евиденције. Са друге стране, законски прописи захтевају имовину - укључујући путну инфраструктуру - коју треба пратити финансијским картама.

За решавање овог проблема органи власти на путу морају створити почетни баланс проценом финансијске вредности њихове имовине. Ово је сложен моделски задатак који захтева комбинацију знања и вештина у рачуноводству, грађевинарству и анализи података.

Овај рад има за циљ да карактерише особине "идеалног" модела за процену вредности путева и анализира три практична приступа овом проблему. Путеве се такође могу проценити са других гледишта: друштвене, еколошке, војни итд. Вредности које се добијају за сваку од ових категорија су различите, јер описују различите погодности које доносе путеви.

4.2.3. Како проценити вредност путне мреже?

Основна правила обично су регулисана законом. За потребе рачуноводства потребне су две врсте вредности:

- почетна вредност (трошак) средства (V)
- његова тренутна вредност (C)

Разлика између ова два описује укупну нарушавање вредности имовине. (D)

$$V=C-D \quad (1)$$

Процена фиксне имовине се састоји или у процени своје тржишне вредности или при израчунавању трошкова замене са новом, која има упоредиве техничке карактеристике. Путна инфраструктура није предмет тржишне размене, па се у овом случају користи приступ трошковима.

Пут је подељен на исте секторе, за које су сви параметри константни. Затим, за сваки један сектор користи једначину (2) за израчунавање површине сектора и једначину (3) како би се добила јединица трошка замене вредности.

На крају, вредност замене целе трасе пута која се одређује по једначини (4), се добија сумирањем трошкова за сваки одсек. Једини проблем у овом поступку је та функција која описује јединични трошак у једначини (3) није позната и мора се моделирати на основу расположивих података.

Такво моделирање је најважнији задатак у процени имовине, па је зато потребна посебна пажња.

$$1. \text{ Површина- } A(i) [m^2] \\ A(i) = (\text{дужина дела } i^{th}) * (\text{ширина дела } i^{th}) \quad (2)$$

$$2. \text{ Цена јединице- } C(i) [EUR/m^2]$$

$$C(i) = f_{C(i)}(\dots) \quad (3)$$

$$3. \text{ Трошкови замене- } C [EUR]$$

$$C = \sum_{i \in U} C(i) \cdot A(i) \quad (4)$$

Својства модела процене имовине:

- Адекватност
- Објективност
- Прегледност
- Формалност
- Употребљивост

4.2.4. Кораци процене [5]

Процена путне инфраструктуре састоји се од следећих корака:

1. Установити принципе, основе и законе за вредновање имовине;
2. Пописати имовину која представља основу за процену вредности путне инфраструктуре;
3. Израчунати полазне вредности имовине путне инфраструктуре што укључује следеће;
4. Изводити одговарајуће јединице стопе за различите имовинске групе и подгрупе;
5. Израчунати бруто трошкове замене за сваку имовину унутар групе или подгрупе;
6. Израчунати утрошак имовине што укључује следеће:
 - рачунајући у годинама амортизацију-процењивање у годинама погоршање и израчунавање губитка вредности кад је потребно;
7. Израчунати амортизовани трошак замене што укључује следеће:
 - израчунати амортизовани трошак замене тако што ће се заменити трошак који рефлектује тренутну старост, стање, учинак имовине;
 - годишње регулисање вредности имовине на рачуну амортизације и погоршања вредности у току године;
8. Припремити извештај о процени.

Кратак преглед кључних корака за процену стања имовине путне мреже:

- утрошак имовине се манифестује у условима амортизације и тренутног стања;

Амортизовани трошкови замене су тренутна новчана вредност имовине или групе имовине, срачунато са Gross Replacement Value (GRV)- Бруто заменљива вредност, пре било каквог прилагођавања за могуће споразуме о нетирању) синоним за нето књиговодствену вредност;

- Бруто заменљива вредност је укупна вредност замене путне инфраструктурне вредности као део постојеће путне мреже;
- Опадање нето вредности имовине услед изненадног или непредвидивог смањења стања и/или учинка имовине, упоређеним са претходним нивоом процене, који није препознатљив од стране амортизације;



Слика 2. Кораци процене вредности путне инфраструктуре [5]

- Једнака вредност имовине је средство које пружа исте потенцијалне перформансе као постојећа имовина, али одражава се на данашњу модерну технологију
- Извештај о процени је извештај који сумира резултате процене и обезбеђује информацијску подршку која се односи на организацију имовине путне инфраструктуре.

Имовина мора бити предмет целокупне ревалоризације, бар једном у пет година.

5. ЗАКЉУЧАК

Главни интерес је утврђивање релативне промене вредности имовине, а не саме апсолутне вредности, што заузврат олакшава процену степена и брзине пропадања. Зато је неопходно обављати периодичан прорачун, како би се могла упоредити садашња вредности имовине са вредношћу током претходних периода. Познавање апсолутне вредности је од секундарног значаја. Обрачун треба да се обавља у правилним интервалима, најмање сваке две до три године, максимално до пет година. Ово ће омогућити процену успешности или неуспешности примењене политике одржавања путева.

6. ЛИТЕРАТУРА

- [1] Закон о озакоњењу објеката ("Службени гласник РС", број 96 од 26. новембра 2015. године).
- [2] Правилник о катастарском премеру и катастру непокретности ("Службени гласник РС", бр. 7/2016 од 29.01.2016. године, а ступио је на снагу 6.02.2016. године).
- [3] Законом о планирању и изградњи („Службени гласник РС“, бр. 72/2009.).
- [4] Закон о општем управном поступку ("Службени гласник РС", бр. 18/2016).
- [5] Закон о републичким административним таксама ("Службени гласник РС", бр. 43/2003, 51/2003 - испр., 61/2005, 101/2005 - др. закон, 5/2009, 54/2009, 50/2011, 70/2011 - усклађени дин. изн., 55/2012 - усклађени дин. изн., 93/2012, 47/2013 - усклађени дин. изн., 65/2013 - др. закон, 57/2014 - усклађени дин. изн., 45/2015 - усклађени дин. изн., 83/2015, 112/2015, 50/2016 - усклађени дин. изн., 61/2017 - усклађени дин. изн., 113/2017 и 3/2018 - испр.).
- [5] ROAD ASSET EVALUATION MODELS, Systems Research Institute, Polish Academy of Sciences, Karol Opara.
- [6] Игор Јокановић, "Процена вредности управљања путном мрежом", Универзитет у Новом Саду, Грађевински факултет Суботица, Козарачка 2а, Суботица, Република Србија
- [7] Материјал са сајта Министарства грађевинарства, саобраћаја и инфраструктуре <http://www.mgsi.gov.rs/lat>
- [8] Материјал са сајта <http://www.geosrbija.rs>
- [9] Материјал са сајта <http://satelitskisnimci.rgz.gov.rs>

Кратка биографија:



Ивана Богнар, рођена у Новом Саду 1988. године. Мастер рад на Факултету техничких наука из области Грађевинарства – Путеви, железнице и аеродроми је одбранила 2018. године.



Небојша Радовић, рођен је у Београду 1962. године. Докторирао је на Факултету техничких наука у Новом Саду 2006. године, а од 2015. године је у звању ванредног професора на Катедри за геотехнику и саобраћајнице.

МЕТОДОЛОШКЕ ОСНОВЕ ПЛАНИРАЊА И ПРОЈЕКТОВАЊА ОБИЛАЗНИЦА – ПРИМЕР ОБИЛАЗНИЦЕ ОКО КАЊИЖЕ

METHODOLOGICAL BASICS OF PLANNING AND DESIGNING OF BYPASSES - EXAMPLE OF KANJIŽA BYPASS

Милош Нешковић, Факултет техничких наука, Нови Сад

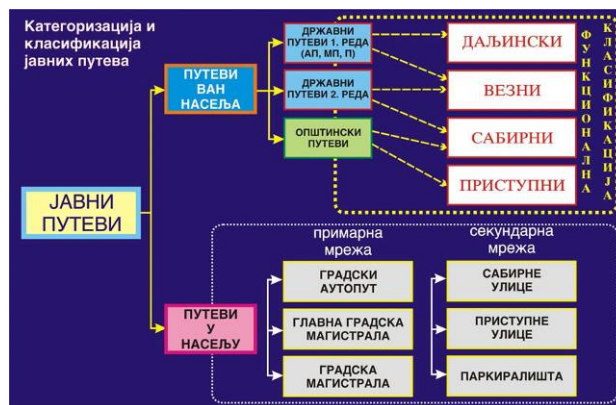
Област- ГРАЂЕВИНАРСТВО

Кратак садржај – Овај рад приказује методологију планирања и пројектовања обилазница као и анализу односа између градске и ванградске путне мреже. Приказано је идејно – инжењерско рјешење обилазног пута око Кањиже.

Кључне речи: Планирање, пројектовање, обилазнице

Abstract – This work presents methodology of planning and designing of bypasses as well as analysis between urban and rural road network. Conceptual engineering solution of the Kanjiža bypass is presented.

Keywords: Planning, designing, bypasses



Слика 1. Функционална класификација [1]

1. УВОД

Методологија планирања и пројектовања путева (ванградских и градских) представља хијерархијски уређен процес који подразумева формализоване поступке како би се са довољно извесности и поузданости донијеле потребне одлуке [1].

Процес започиње концепцијским приступом утврђивању оптималног путног коридора (Генерални пројекат), а затим преко избора оптималне трасе пута у наведеном коридору (Идејни пројекат) и утврђивања оптималних инжењерских рјешења за изградњу пута (Главни и Извођачки пројекат) завршава се формирањем релевантних база података, историјских и активних, о изведеном објекту (Архивски пројекат).

2. ТЕХНИЧКА И ЗАКОНСКА РЕГУЛАТИВА ПУТНЕ ИНФРАСТРУКТУРЕ

Административна категоризација јавних путева (према Закону о јавним путевима) обухвата путеве ван насеља и путеве у насељу сагласно њиховом положају у реалном простору (слика.1).

Путеви ван насеља се дијеле на:

- државне путеве (у надлежности Р. Србије, односно Републичке дирекције за путеве)
- општинске путеве (у надлежности локалне самоуправе)

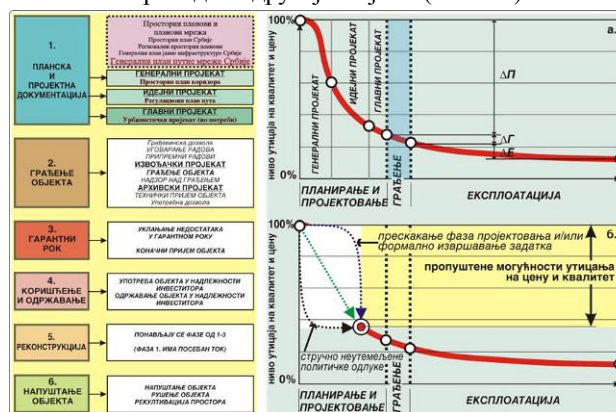
НАПОМЕНА:

Овај рад је проистекао из мастер рада чији је ментор био др Небојша Радовић, в.проф.

Поред категоризације јавних путева, од изузетног је значаја извршити њихову класификацију сагласно улози у путној мрежи и одговарајућим техничким и функционалним карактеристикама [1]. У том погледу посебно се издваја тзв. „функционална класификација“ (слика 1) којом се једнозначно дефинише значај и улога пута у мрежи и успоставља директна веза са мјеродавним брзинама, односно, димензионисањем његових пројектних елемената.

3. ПРОЦЕС ПЛАНИРАЊА И ПРОЈЕКТОВАЊА

Процес планирања, пројектовања, грађења, одржавања и експлоатације путева, стартује са нивоа Генералног плана путне мреже дефинисањем програмских услова и Пројектног задатка за израду Генералног пројекта, а завршава рекултивацијом простора напуштеног пута пошто је исцрпљен животни вијек пута и путно земљиште приведено другој намјени (слика 2).



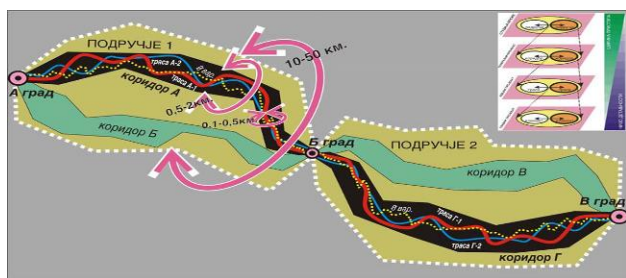
Слика 2. Алгоритам животног века грађевинског објекта (пута) [1]

Захваљујући унапређењу технологије планирања и пројектовања (најшира примјена CAD-а, CAE-а, CIC-а) данас је могуће, прије физичке градње пута, објективно процијенити најзначајније утицаје:

загађење животне средине,
сигурност саобраћаја,
ниво услуге,
пропусну моћ,
однос пута и околине,
естетику трасе,
потребна финансијска средства и сл.

3.1 Избор локације за изградњу пута

Један од основних циљева процеса планирања путева (Генерални и Идејни пројекат) јесте избор оптималне локације путног правца из чега проистичу суштинске карактеристике пута [1]. Проблем локације (гдје?) јавља се као кључни проблем у свим фазама планирања и пројектовања путева (слика 3).



Слика.3 Избор оптималне локације у зависности од фазе планирања/пројектовања пута [1]

На нивоу Генералног пројекта потребно је одредити оптимални коридор у оквиру дефинисаног подручја, док се на нивоу Идејног пројекта истражује и једнозначно дефинише оптимална траса у усвојеном оптималном коридору на нивоу претходне фазе – Генералног пројекта.

4. АНАЛИЗА ПРИМЈЕНЕ РАЗЛИЧИТИХ КОНЦЕПЦИЈА У ПЛАНИРАЊУ И ПРОЈЕКТОВАЊУ ОБИЛАЗНИХ ПУТЕВА

У широј утицајној зони градског насеља на ванградском путу се јављају двије групе саобраћајних токова:

- **пролазни токови**, којима треба омогућити наставак кретања под приближно истим условима ванградског пута и
- **изворно - циљни токови**, који се са ванградског пута усмјеравају на градску путну мрежу коју истовремено користе и интерна градска кретања.

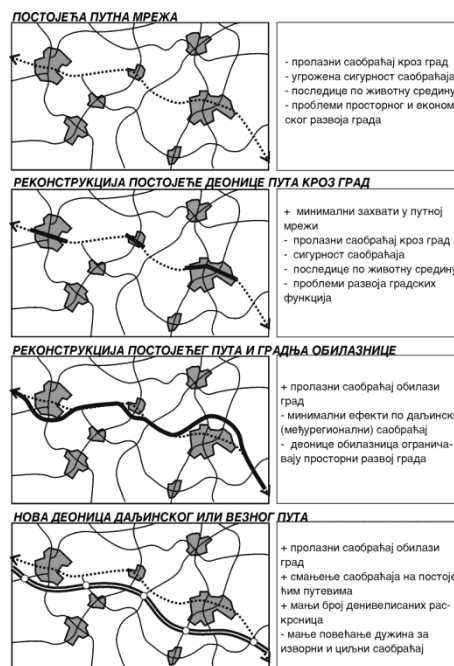
У оквиру овог поглавља дефинисане су три основне концепцијске алтернативе:

- реконструкција постојеће дионице ванградског пута кроз градско насеље по мјерилима градске саобраћајнице
- градња обилазнице уз реконструкцију ванградских дионица постојећег пута и
- изградња нове дионице даљинског или везног пута.

Свака од алтернатива има одређене позитивне и негативне карактеристике а подручје њихове примјене дефинисано је нивоом градског насеља и функционалним рангом пута [2].

4.1 Концепцијска рјешења односа ванградског пута (обилазнице) и градског насеља

Када се каже развој градских насеља и путне мреже која их повезује, мисли се на развој постојеће путне мреже која се обично не спаја са даљинским токовима [2]. Постојећа мрежа пролази кроз градска насеља, гдје се ширењем града формира ивична градња дуж путне дионице оптерећујући је градским видовима превоза и честим прикључивањем урбаних садржаја.



Слика.4 Могућа концепцијска рјешења односа ванградског пута и градског насеља [2]

За разрјешење оваквог просторно - функционалног сукоба (слика 4) могу се формулисати три концепцијске алтернативе:

- реконструкција постојеће дионице пута кроз градско подручје подразумијева да пролазни саобраћај користи исту дионицу као и изворно-циљни токови заједно са интерним градским кретањима
- изградња обилазнице је првенствено намијењена пролазном саобраћају уз дјелимично коришћење за изворно - циљна кретања у зависности од локације прикључака те се пројектује и гради по принципима који важе за ванградску дионицу
- изградња нове дионице даљинског или везног пута напушта концепт класичне обилазнице те се у суштини ради о дионици ванградског пута на ширем подручју града.

5. ИДЕЈНО-ИНЖЕЊЕРСКО РЈЕШЕЊЕ ОБИЛАЗНОГ ПУТА ОКО КАЊИЖЕ

Предмет рада је израда прве фазе обилазног пута око Кањиже у приближној дужини од 4638 m који почиње на km 0+384,81,а завршава се на km 5+022.00. Дио обилазног пута од 0+000.00 до km 0+384,57 као и дио од km 5+022.81 до km 5+081.81 представља прикључење обилазног пута на државне путеве првог и другог реда. То прикључење су две трокраке раскрснице,и као такве су предмет посебне пројектне документације.

5.1 Програмски параметри за трасирање

Програмски услови за трасирање дефинишу се преко:

- Класификације пута
- Саобраћајне функције
- Мјеродавне брзине
- Мјеродавних параметара возача, возила и пута

Класификација пута се одређује према административној и функционалној класификацији [3]. По овој класификацији разликују се четири врсте путева према основним функцијама:

- Приступни путеви (ПП)
- Сабирни путеви (СП)
- Везни путеви (ВП)
- Даљински путеви (ДП)

За одређивање карактеристика терена користимо следећу табелу (табела 1):

Табела 1. Показатељи за прелиминарно одређивање карактеристике терена

Релативна висинска разлика на 1000m одстојања	равничарски	брдовит	планински
	≤ 50 m	50 – 150 m	≥ 150 m
Нагиб падина	$\leq 1:10$	1:10 – 1:2	$\geq 1:2$

Пошто се релативне висинске разлике дуж цијеле трасе, гледано на сваких 1000m, крећу у распону од 0.03-6.43m, ријеч је о **равничарском** путу.

Основна брзина је полазни програмски параметар који у суштини представља индикатор нивоа услуге одређеног путног правца при меродавном саобраћајном оптерећењу. Ова брзина одређује се на основу саобраћајног значаја путног правца и макро показатеља просторних ограничења (табела 2)

Табела 2. Вредност основне брзине у функцији разреда пута и топографских карактеристика

Врста пута	Карактеристике терена		
	равничарски	брдовити	планински
даљински	100	80	60
везни	80	70	50
сабирни	60	50	40
приступни	50	40	30

Основна брзина за предметну трасу је **$V_0 = 80$ km/h**.

Рачунска брзина (V_r) је усвојена теоријска вриједност која служи за прорачун граничних геометријских елемената који се могу примјенити у процесу пројектовања путева. Рачунска брзина се може одредити на основу Основне брзине V_0 и то:

$$V_r = V_0 + 20 \quad [1]$$

$$V_r = 80 + 20 = 100 \text{ km/h}$$

5.2 Нормални попречни профил пута

Попречним профилем се утврђује садржај и димензије функционалних елемената у складу са поставкама саобраћајног програма [3]. Возне траке као елемент попречног профила су намјењене искључиво за проточни саобраћај а њихова ширина се одређује на основу следеће табеле (табела 3):

Табела 3. Ширина возних трака у функцији рачунске брзине

V_r (km/h)	t_v (m)	Тип пута и карактер терена
$V_r > 100$	$t_v = 3,75$ m	АП (Равничарски)
$80 < V_r < 100$	$t_v = 3,50$ m	АП (Брдски, планински), ВП, П
$60 < V_r < 80$	$t_v = 3,25$ m	П
$40 < V_r < 60$	$t_v = 3,00$ m	П
$V_r \leq 40$	$t_v = 2,75$ m	П

5.3 Анализа саобраћајног оптерећења

У анализи саобраћајног оптерећења узимају се у обзир следећи параметри [4]:

- Просјечни годишњи дневни 24-часовни број тешких теретних возила (са процентуалним учешћем појединих категорија) у почетној години експлоатације пута. Одређује се првенствено бројањем саобраћаја и прогнозама
- Просјечна годишња стопа раста броја тешких теретних возила у пројектном периоду, која се одређује на основу података о почетној години уз примјену дугорочних прогноза саобраћајних токова
- Осовинска оптерећења репрезентативних возила
- Просјечна искоришћеност носивости тешких теретних возила (ако не постоје прецизни подаци, препоручује се искоришћеност од 70%)
- Расподјела саобраћајног оптерећења по тракама

5.4 Коловозна конструкција

Димензионисање коловозне конструкције урађено је према важећем SRPS.U.C4.012. стандарду [5].

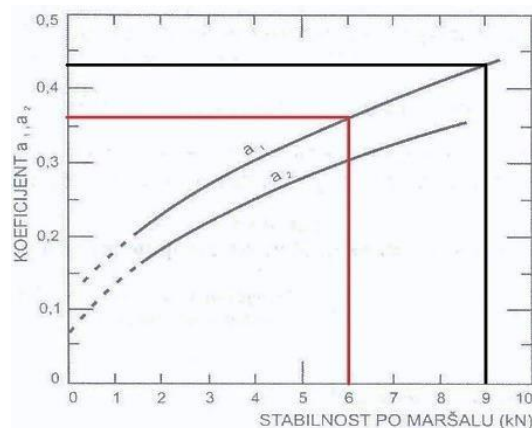
Поступак димензионисања се састоји од две фазе:

- Одређивање мјеродавних параметара за димензионисање,
- Одређивања састава и дебљине појединих слојева на основу дијаграма.

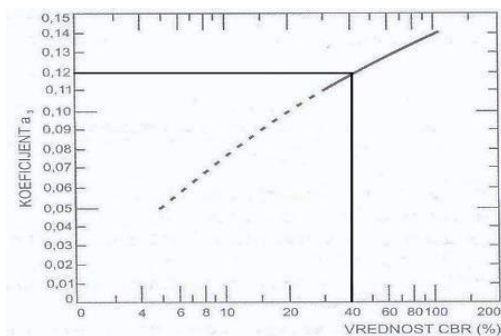
Мјеродавни параметри за димензионисање су:

- пројектни период, 20 година
- саобраћајно оптерећење, $T_v = 2.0 \times 106 \text{ ECO } 82 \text{ kN}$
- носивост материјала постељице, $\text{CBR} \geq 4 \%$, $\text{CBRNS} = 40\%$ за неvezани камени агрегат.

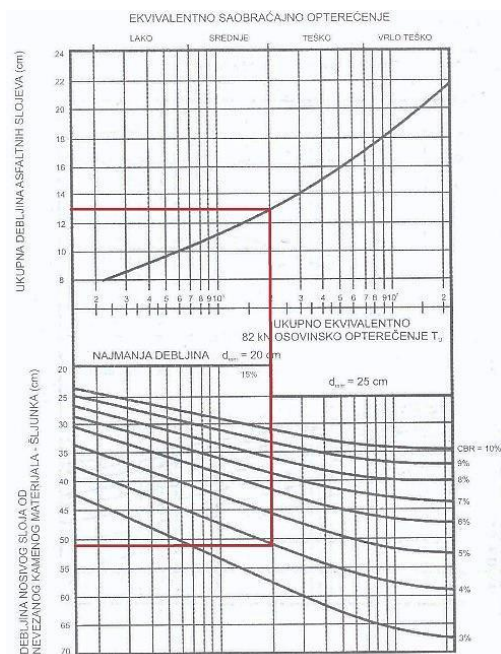
Израчунате стабилности по Маршалу за АБ и БНС су следеће: $\text{STAB} = 8 \text{ KN}$; $\text{STBNS} = 9 \text{ KN}$;



Слика 5. Коефицијент замјене за асфалт бетон АБ и носиви слој од битуменизованог дробљеног каменог материјала БНС [5]



Слика 6. Коефицијент замијене за носиви слој од зрнастог каменог материјала-шљунак, дробљени камени материјал [5]



Слика 7. Дијаграм за димензионисање коловозних конструкција од асфалтних слојева и носивих слојева од неvezаног зрнастог каменог материјала [5]

На основу „Правилника о основним условима које јавни путеви и њихови елементи морају да испуњавају са гледишта безбједности саобраћаја“ и дијаграма за димензионисање коловозне конструкције одређујемо следеће дебљине:

- укупна дебљина асфалтних слојева је $d_{as}=13\text{ cm}$;
- укупна дебљина носивих слојева од неvezаног каменог материјала је $d_n=51\text{ cm}$

Такође из наведених дијаграма читавамо и коефицијенте замијене материјала:

- За асфалт бетон, $a_1=0,43$
- За битуменом везане носеће слојеве, $a_2 = 0,36$
- За асфалтне слојеве, $a_{AS} = 0,38$
- За дробљени камени материјал, $a_3 = 0,12$
- За дробљени природни шљунак, $a_n = 0,09$

Одређивање ST^* (или индекс дебљине коловозне конструкције):

$$S_T^{закт.} = a_{AS} \times d_{as} + a_n \times d_n \quad [2]$$

$$S_T^{закт.} = 0,38 \times 13,00 + 0,09 \times 51,00 = 9,53$$

Претпостављена дебљина асфалт бетона је 5 cm. На основу ње и коефицијента замијене материјала одређујемо дебљину БНС-а на следећи начин:

$$a_{AS} \times d_{as} = a_1 \times d_{AB} + a_2 \times d_{BNS} \quad [3]$$

$$0,38 \times 13 = 0,43 \times 5,0 + 0,36 \times d_{BNS} \rightarrow d_{BNS} = 7,75$$

Усвајамо $d_{BNS} = 8,0\text{ cm}$

Одређивање дебљине НС на основу коефицијента замијене материјала:

$$0,09 \times 51 = 0,12 \times 20 + 0,09 \times d_{(0-64)} \rightarrow d_{(0-64)} = 24,33$$

Усваја се $d_{(0-63)} = 31,0\text{ cm}$

Провјера структурног броја:

$$S_T^{оств.} = a_1 \times d_{AB} + a_2 \times d_{BNS} + a_n \times d_{(0-63)} + a_3 \times d_{(0-31)} \quad [4]$$

$$S_T^{оств.} = 0,43 \times 5,0 + 0,36 \times 8,0 + 0,09 \times 31,0 + 0,12 \times 20,0 = 10,22$$

У коловозној конструкцији мора се реализовати следећи услов: $S_T^{оств.} > S_T^{закт.}$, што је у овом случају испуњено.

6. ЗАКЉУЧАК

Циљ овог рада је био приказ методологије планирања и пројектовања обилазница као и анализа односа између градске и ванградске путне мреже. Такође је приказано и идејно – инжењерско рјешење обилазног пута око Кањиже, у оквиру кога су анализирани одређени елементи трасе (ситуациони план, попречни профили и сл.) Све то је рађено на основу претходно прикупљених геодетских и геотехничких подлога као и анализе саобраћајног оптерећења које су биле основ за пројектовање саме обилазнице.

7. ЛИТЕРАТУРА

- [1] Анђус В., Малетин М.: Методологија планирања и пројектовања путева
- [2] Малетин М.: Обилазнице
- [3] Радовић Небојша: Скрипта са предавања из предмета:Одабрана поглавља из пројектовња путева, Факултет Техничких Наука Нови Сад
- [4] Узелац Ђорђе: Предавања из путних и градских саобраћајница, Факултет Техничких Наука Нови Сад 2012
- [5] СРПС У. Ц4.012 – Димензионисање нових асфалтних коловозних конструкција, Београд, 1981

Кратка биографија:



Милош Нешковић рођен је у Љубовији 1992.год. Основне студије на Факултету техничких наука из области Грађевинарства - Путеви, жељезнице и аеродроми завршио је 2016.год. Тренутно је на мастер студијама на смијеру Путеви, жељезнице и аеродроми

HIDRAULIČKA ANALIZA VODOSNABDEVANJA OPŠTINE SRBOBRAN

HYDRAULIC ANALYSIS OF THE WATER SUPPLY SYSTEMS OF THE SRBOBRAN MUNICIPALITY

Siniša Sadžakov, *Fakultet tehničkih nauka, Novi Sad*

Oblast – GRAĐEVINARSTVO

Kratak sadržaj – U okviru ovog rada predstavljena je konceptualna ideja razvoja centralizovanog vodosnabdevanja opštine Srbobran. Na osnovu uvida u planski dokument došlo se do urabnističkih planskih podataka i bilansa namene prostora. Najveći aspekt je dat na proširenje radne aktivnosti i pretvaranje poljoprivrednog zemljišta slabije bonitetne klase u javnoj svojini u zemljište radnih zona. Procentualno proširivanje radnih zona, kao velikih potrošača vode u tehnološkim procesima, dovodi do bitnog povećanja kapaciteta za vodosnabdevanje opštine Srbobran. Analizirane su varijante procentualnog povećanja radnih zona za projektni period od 40 god. na osnovu kojih je izabrano najpogodnije rešenje u pogledu postizanja hidrauličke stabilnosti i zadovoljenja ranga povećanja eksploatacionih kapaciteta postojećeg izvorišta "Popovača".

Ključne reči: Hidraulička analiza, Srbobran, Turija, Nadalj, EPA-NET, Vodosnabdevanje

Abstract – In this master thesis, the conceptual idea of the centralized water supply system of the municipality Srbobran has been presented. Based on the insights into planning document, the urbanistic planning data and information about the area's purpose has been found. The biggest accent is given to the extension of the labor activities and turning agricultural soil of the pourer credit rating class, owned by the public, into a working zone area. Extension of the working zone areas, as a big water consumer in the technological processes, is leading to the large increase of the water supply capacity in the municipality Srbobran. Variants for the percentage increase of the working zones have been analyzed for the project period of the 40 years, based on which the most adequate solution has been chosen in terms of acquiring hydrologic stability and meeting the requirements of the exploitation capacities of the existing water sources "Popovača".

Keywords: Hydraulic analysis, Srbobran, Turija, Nadalj, EPA-NET, Water Supply sistem

1. UVOD

Opština Srbobran se nalazi u središnjem delu Vojvodine i Bačke i sačinjavaju je tri gradska naselja: Srbobran, Turija i Nadalj. U pogledu zajedničkog problema vodosnabdevanja za sva tri naselja u opštini Srbobran, jeste dotrajalo instalacija distributivnih mreža koje su u

lošem fizičkom stanju što dovodi do činjenice da sistemi nisu u stanju da izdrže proširenje kapaciteta u vidu novoprojektovanog vodosnabdevanja za naredni projektni period, a karakteristike cevovoda ne zadovoljavaju važeće tehničke standarde za hidrantsku mrežu za gašenje požara. U svim naseljima kvalitet vode za piće ne zadovoljava standarde higijenske ispravnosti i tretira se direktnim hlorisanjem.

Na osnovu uvida u planska dokumenta nadležnih javni preduzeća trebalo je izraditi konceptualnu ideju centralizovanog vodosnabdevanja za sve kategorije potrošača u sva tri naselja.

Najpogodnije mesto za tretman vode jeste u blizini samog izvorišta, sto je još jedan presudan faktor zasto se odabrao idejni koncept centralizovanog rešenja vodosnabdevanja. Cilj rada je da se rekonstrukcijom mreže, modernizacijom sistema vodosnabdevanja gubici vode u mreži smanje, omogućiti konstantan potreban pritisak i uspostavi sigurnije i kvalitetnije vodosnabdevanje kao i da se obezbedi propisani kvalitet vode za piće za sva tri naselja u opštini Srbobran.

Na osnovu uvida u planski dokument [1], došlo se do određenih potrebnih podataka koji su osnov za ovaj rad. Izvod iz ovog planskog dokumenta (slika 1.) predstavlja tabelarni prikaz bilansa postojeće i planirane namene površina, gde je najveći aspekt dat na proširenje radnih zona.

Р.бр.	Намена површина	Постојеће		Планирано	
		ha	%	ha	%
1.	Пољопривредно земљиште	26.740,33	94,12	25.193,31	88,88
	Њиве, воћњаци, виногради, ливаде и пашњаци	26.722,03		25.170,34	
	Трстици и мочваре	9,85		9,85	
	Рибњак	8,45		8,45	
	Извориште	/		4,67	
2.	Шумско земљиште	14,07	0,05	14,07	0,05¹
	Шуме и шумско земљиште	14,07		14,07	
3.	Водно земљиште²	515,29	1,50	585,59	1,80
	Канал ХС ДТД Бечеј-Богојево	240,65		240,65	
	Водоток Криваја и Јегричка	147,13		147,13	
	Акумулација	/		69,60	
	Бељанска бара	12,67		12,67	
	Баре	8,73		8,73	
	Мелиоративна каналска мрежа	106,81		106,81	
4.	Грађевинско земљиште³	1.159,86	4,33	2.636,58	9,27
	Грађевинско подручје насеља	807,56		807,56	
	Радне зоне	153,19		1.486,61	
	Викенд зоне	3,31		49,14	
	Туристички садржаји	/		23,24	
	Комунални и енергетски објекти	37,50		50,53	
	Међународни аутопут Е 75	2,10		2,10	
	Државни пут I реда	83,95		110,65	
	Државни пут II реда	26,78		26,78	
	Државни пут-пут регионалног значаја	/		25,2	
	Општински пут	21,91		28,88	
	Пристапни пут	/		2,26	
	Железничка пруга	23,65		23,65	
	Укупно 1+2+3+4	28.429,55	100	28.429,55	100

Slika 1. Bilans namene površina-izvod iz plana [1]

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio doc.dr Matija Stipičić, dipl.inž.grad.

2. POTROŠNJA VODE I KOEFICIJENTI NERAVNOMERNOSTI

Analiza je izvršena za četrdesetogodišnji projektni planski perion, 2051 god. u odnosu na poslednji javni popis stanovništva koji je bio 2011 god. Do konačnog povećanje broja stanovnika implementiranih u proračune došlo se preko sledeće jednačine;

$$N_{2051} = N_{2011} * (1 + \frac{P}{100})^n \quad (1)$$

Gde su: N_{2051} -broj stanovnika na kraju planskog perioda, n-usvojeni planski period od 40 godina, p-prosečna godišnja stopa prirasta stanovništva, N_{2011} -broj stanovnika po poslednjem popisu.

Celokupan proračun analize potrošnje vode svodi se na analizu merodavne potrošnje po određenim kategorijama potrošača.

2.1. Merodavna potrošnja

- Stanovništvo - merodavna potrošnja određena je na osnovu potreba u domaćinstvima, javnim ustanovama kao i potrebe za vodom domaćih životinja. Ovi podaci, u smislu broja korisnika i broja grla krupne i sitne stoke dobijeni su od nadležnih i javnih ustanova. Gore pomenuta potreba za potrošnjom vode stanovništva svedena je po glavi stanovnika (l/stan/dan) i predstavlja srednju dnevnu potrošnju stanovništva $Q_{sr,dn}$.
- Gubici i potrošnja za sopstvene potrebe - potrošnja vode za sopstvene potrebe vodovoda se odnosi na količinu vode koju svaki vodovodni sistem troši na pranje (ispiranje) i rekonstrukciju vodovodnih objekata. Ne postoje u praksi odgovarajući normativi koji propisuju ove količine, već se one određuju iskustveno. Orijentaciono, ova potrošnja iznosi 5 do 10 (%) srednje dnevne potrošnje, $Q_{sr,dn}$

Potrošnja vode prethodnih kategorija potrošača ulaze u sastav specifične potrošnje q_{sp} .

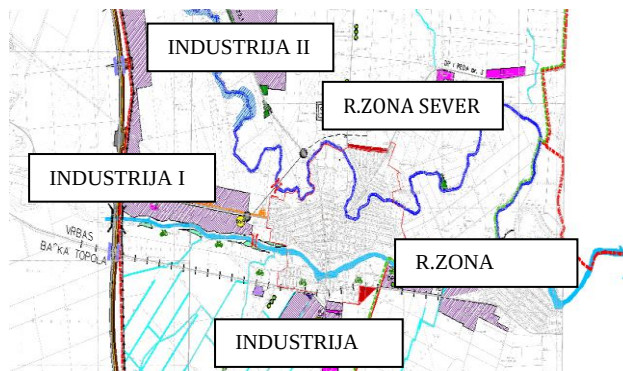
- Potrošnja vode za gašenje požara - Ova kolčina vode definisana je Pravilnikom o tehničkim normativima za hidrantsku mrežu za gašenje požara. Ovim pravilnikom se propisuju, između ostalog, tri osnovna parametra za gašenje požara hidrantskom mrežom; (potrbna količina vode za gašenje požara, najmanji pritisak i broj istovremenih požara kao i njihovog trajanje) [2].
- Potrošnja industrijskih i radnih zona - Kod određivanja potrošnje vode za industrijske potrebe planirane potrošnje za sva tri naselja koristili smo obrazac iz Nemačkih standarda DWG W410. (aktivnosti sa malom potrošnjom vode) [3].

$$Q_{sp,ind.} = 0,2 \text{ do } 0,5 \text{ L/(s * ha)} \quad (2)$$

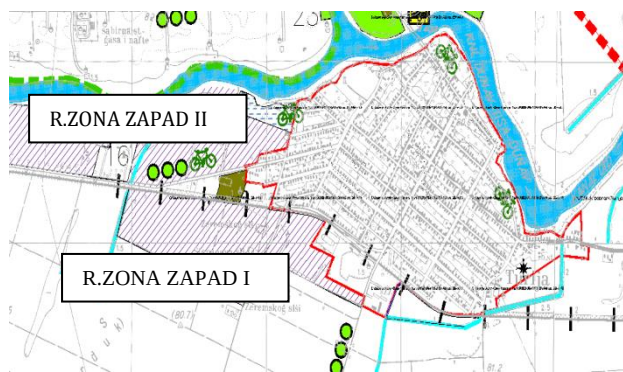
Ovim radom izvršena je komparativna analiza u različitim varijantama proširenja potrošnje industrijskih i radnih zona. S obzirom na planirani projektni period od 40 godina, u ovom radu pristupili smo obrađivanju III varijantna rešenja koja se odnose na ove bitne potrošače u naseljima.

- VARIJANTA I; 100% planiranog razvoja
- VARIJANTA II; 30% planiranog razvoja
- VARIJANTA III; 15% planiranog razvoja

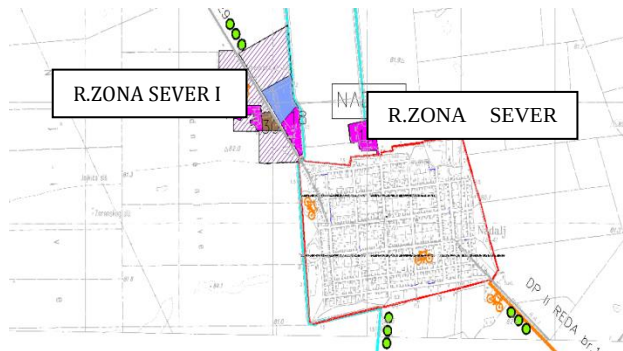
Na slikama 2, 3 i 4 prikazan je planirani razvoj industrijskih i radnih zona za sva tri naselja redosledno (Srbobran, Turija i Nadalj-izvod iz Prostornog plana) [1].



Slika 2. Planirani razvoj industrijskih i radnih zona - Srbobran [1]



Slika 3. Planirani razvoj industrijskih i radnih zona - Turija [1]



Slika 4. Planirani razvoj industrijskih i radnih zona - Nadalj [1]

2.2. Koeficijenti neravnomernosti

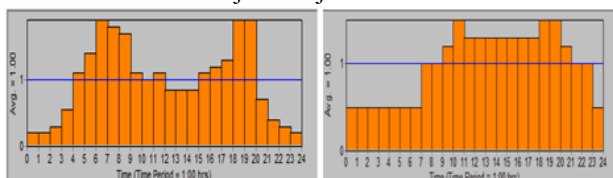
U ovom radu merodavni koeficijenti neravnomernosti za stanovništvo računati su po obrascu iz Nemačkih standarda DWG W410, gde je; (E-br.stanovnika na kraju planskog perioda, Kd-koef.max.dnevne neravnomernosti, Kh-koef.max.časovne neravnomernosti) [3].

$$fd = 3,9 * E^{-0,0752} \rightarrow Kd = fd \quad (3)$$

$$fh = 18,1 * E^{-0,1682} \rightarrow Kh = fh/fd \quad (4)$$

Koeficijenti neravnomernosti za sva tri gradska naselja usvojeni su po časovnoj raspodeli tokom 24h. Raspodela koeficijenata uzeta je na osnovu dobijenih rezultata višegodišnjeg monitoringa, (merenja i osmatranja) navika stanovništva u svrhu potrošnje vode.

Ovi podaci dobijeni su od nadležnog javno komunalnog preduzeća. Na osnovu ovih podataka uočeno je kada dolazi do maksimalne časovne potrošnje u toku dana. Koeficijenti neravnomernosti za industriju usvojeni su prema literaturi, gde su; $K_d=1,1$ a $K_h=1,5$. Na slici 5. prikazan je oblik dijagrama po časovnoj raspodeli za stanovništvo i industriju naselja Srbobran.



Slika 5. Koeficijenti časovne neravnomernosti (stanovništvo i industrija)

3. PROJEKTOVANI KAPACITETI I ANALIZA POTREBNOG BR. BUNARA

3.1. Kapaciteti

U svim varijantama merodavne količine vode, potrebne za projektovanje, razlikuju se na osnovu proširenja kapaciteta industrije. Potrebne količine vode za potrošnju stanovništva ostaju iste u sve III varijante i za sva tri naselja. Proračunski kapaciteti se svode na ($Q_{sr,dn}$ -srednji dnevni; $Q_{max,dn}$ - maksimalni dnevni i $Q_{max,h}$ - maksimalni časovni), koji se računaju po sledećim obrascima: [3].

$$Q_{sr,dn,stan.} = N_{2051} * q_{sp} \quad (5)$$

$$Q_{max,dn,stan.} = k_{max}^{dn} * Q_{sr,dn,stan.} \quad (6)$$

$$Q_{max,h,stan.} = k_{max}^h * Q_{max,dn,stan.} \quad (7)$$

Kapaciteti za potrebe industrije računati su po obrascima; [3].

$$Q_{max,dn,ind.} = k_{max}^{dn} * Q_{sp,ind.} \quad (8)$$

$$Q_{max,h,ind.} = k_{max}^h * Q_{max,dn,ind.} \quad (9)$$

3.2. Potreban broj bunara

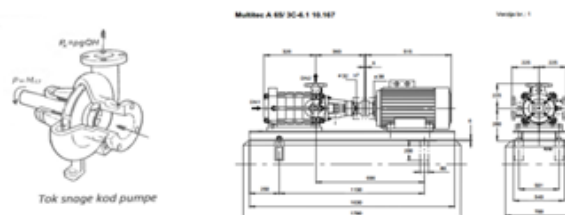
Vodosnabdevanje kompletne opštine Srbobran planirano je sa vodozahvata „Popovača“, sa određenim predtretmanom. Potrebne količine vode podrazumevaju sumirane kapacitete, odnosno potrebe za sva tri naselja. Na osnovu tih kapaciteta dolazi se do potrebnog broja bunara. Izdašnost jednog bunara procenjuje se na 10 l/s.

4. PUMPE I REZERVOARI

4.1. Pumpe

Pumpe spadaju u hidramašinsku opremu čiji je osnovni zadatak da dostave radnu tečnost do potrošača. Pumpe se u pumpnoj stanici vezuju paralelno, a obično se vezuju pumpe istih karakteristika.

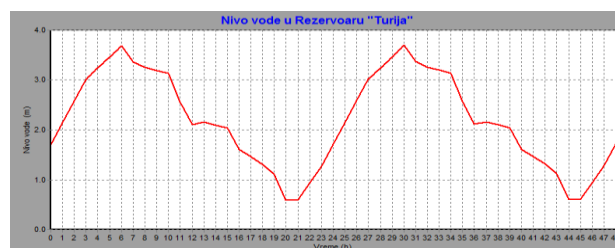
Centrifugalne pumpe (Slika 6.) su najčešće korišćene u hidrotehnici. Najznačajniji pokazatelji pumpi jesu energetski pokazatelji (protok, napor, snaga, stepen korisnosti i kavitacijska rezerva) [4].



Slika 6. Centrifugalna pumpa [4]

4.2. Rezervoari

Jedna od osnovnih uloga rezervoara je regulacija neravnomernosti potrošnje. Rezervoar se puni u periodu tokom celog dana do određenog nivoa, a prazni se u period 24h, tada nivo vode u rezervoaru isključivo zavisi od neravnomernosti potrošnje samih potrošača, prikazano je na (Slika 7.). Ukupna zapremina rezervoara sastoji se iz (zapremine za izravnavanje potrošnje, zapremine za gašenje požara i havarijske zapremine).



Slika 7. Promena nivoa vode u rezervoaru

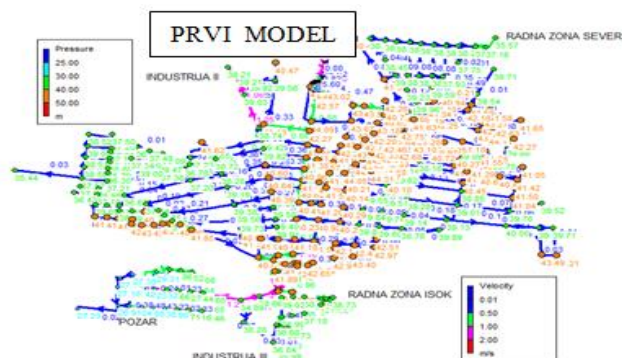
5. SOFTVERSKI PAKET EPA-NET

„Epa-net“ je programski softver za modeliranje komponenta vodovodnog sistema i sagledavanje njihovog zajedničkog rada u celini. Razvijen je od strane SAD tj. Agencije za zaštitu životne sredine (EPA). Prvi put se pojavio 1993 godine. Ovaj računarski program dizajniran je da bude istraživački alat radi poboljšanja našeg razumevanja kretanja i praćenja kvaliteta i količine pijaće vode kroz distributivnu mrežu [5].

6. SIMULACIONI MODELI

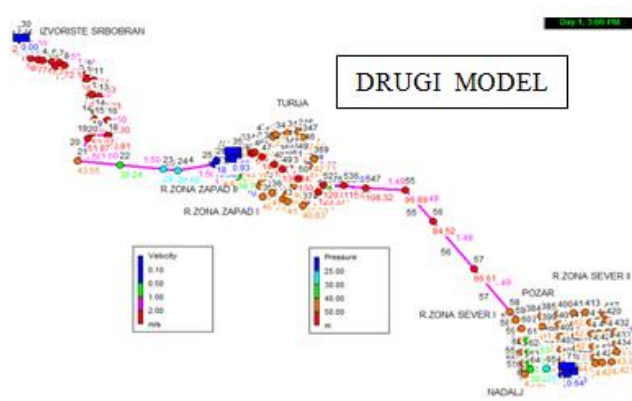
Simulacioni modeli urađeni su kao dekomponovan model u celini.

Prvi model (Slika 8.) urađen je za naselje Srbobran.



Slika 8. Dekomponovani model – prvi model

Drugi model (Slika 9.) urađen je za naselja Turiju i Nadalj zajedno sa dovodno transportnim cevovodima.



Slika 9. Dekomponovani model – drugi model

U ovom poglavlju vršeno je sagledavanje i analiziranje ponašanja distributivne mreže u slučaju tri varijante proširenja industrijskih kapaciteta u opštini Srbobran tjst. u naseljima Srbobran, Turija i Nadalj. Različite varijante usled različitih potrebnih kapaciteta daju drugačije izlazne podatke i zahtevaju promene u hidrauličkom smislu dimenzionisanja objekata u sklopu sistema.

Cilj upotrebe ovakvih analiza jeste sveobuhvatno sagledavanje prednosti i mana projektovanih sistema u smislu uticaja na distributivnu mrežu kao i sagledavanje sigurnosti i kvaliteta vodosnabdevanja. Hidrauličkim proračunom u softverskom paketu „EPA-NET“ došlo se do potrebnih prečnika cevovoda, brzina tečenja vode u cevovodima i pritiskova u čvorovima. Sve tri varijante (I,II,III) predstavljaju sistem vodosnabdevanja na potisnom principu. U analiziranim varijantama elementi i konfiguracija distributivne mreže su identični kao i kategorije potrošača.

7. ZAKLJUČAK

Nakon sagledavanja realnog činjeničnog stanja, pažljivom analizom postojećih podataka dobijenih od nadležnih ustanova i nakon sistematske hidrauličke analize, došlo se do zaključka da je u vidu rezervi kaptažnih podzemnih slojeva najpogodnije izvoriste „Popovača“ u naselju Srbobran. Izvoriste je pogodno za buduća proširenja. Gubici u vodovodnom sistemu u naseljima su u granicama oko 15% prosečne dnevne potrošnje. Da bi se ovi gubici smanjili na planskih 10% potrebno je izvršiti rekonstrukciju mreže i zamenu dotrajalih azbest-cementnih i čeličnih cevi sa PEHD 100 nazivnog pritiska PN 10 bara, za distributivne cevovode u naselju. Ukupan dužina rekonstruisane mreže za sva tri naselja iznosi; $(68138,867 + 19449,025 + 19040,421) = 106.628,313$ m.

U distributivnim mrežama naselja dominantan je prečnik Ø110 što je ujedno i minimalni prečnik prema Pravilnikou o hidrantskoj mreži za gašenje požara [2]. Projektovanjem hidrantske mreže postignut je veći stepen protivpožarne zaštite, a hidranti su postavljeni i pokrivaju celokupno područje vodosnabdevanja.

U svrhu povezivanja planiranog hidrauličkog sistema treba izgraditi nove dovodno-transportne cevovode između naselja. Cevovodi su takođe projektovani od PEHD 100 nazivnog pritiska PN16 i PN20 bara.

Za ispravno funkcionisanje i održavanje distributivne mreže predviđene su nove vodovodne armature i fazonski komadi od livenog gvožđa.

Komparativnom analizom sistema vodosnabdevanja i analizom njihovog uticaja na distributivnu mrežu došlo se do zaključka da je u tehničkom pogledu najprihvatljivija varijanta II. Ova varijanta ima pogodne hidrauličke karakteristike za celokupan sistem. U časovima vršne potrošnje pritisci u naseljima iznose od 3,5bara do 4,5bara, čime je zadovoljen uslov da pritisak u kritičnom čvoru ne sme da padne ispod 2,5bara.

Brzine tečenja u distributivnim mrežama u naseljima kreću se u intervalu od 0.01(m/s) do 1.19 (m/s). Brzine koje se javljaju u ovom opsegu imaju za posledicu male linijske gubitke. Ovakvi gubici daju pogodne karakteristike (blagu krivu cevovoda) prilikom odabira pumpi za distributivni sistem u naseljima. Preporuka je da brzina vode u cevima ne bi trebala da bude manja od 0.4(m/s) zbog održanja kvaliteta vode.

Zbog tog razloga se preporučuje da se u deonicama sa brzinom manjom od 0.4(m/s) češće vrši ispiranje pomoću hidranata ispiraća. Brzine u dovodno-transportnim cevovodima su konstantne i nisu manje od 1,5(m/s), što ima povoljan uticaj pri pojavi hidrauličkog udara. Hidraulički udar je jedna mnogo bitna pojava u sistemu koja može da izazove havarije velikih razmera; oštećenje pumpi, kidanje cevovoda i oštećenje cevne armature i spojeva.

Ublažavanje hidrauličkog udara u sistemu postiže se ugrađivanjem usisno ispusnih vazdušnih ventila i nepovratnih ventila na cevovodima. Pumpe se u pumpnoj stanici štite nepovratnim odbojnim klapnama i ekspanzionim posudama, koje takođe imaju pogodan uticaj na ublažavanje hidrauličkog udara.

8. LITERATURA

- [1] *Prostorni plan opštine Srbobran* - Nosilac izrade plana: Odeljenje za urbanizam stambeno-komunalne poslove i zaštitu životne sredine - AP Vojvodina - Obrađivač: JP Zavod za urbanizam Vojvodine, Novi Sad 2013.
- [2] Službeni list SFRJ: *Pravilnik o tehničkim normama za hidrantsku mrežu za gašenje požara*.
- [3] doc.dr Matija Stipić: *Skripta iz predmeta komunalna hidrotehnika, Deo1-Snabdevanje naselja vodom* 2017.
- [4] Prof.dr Uzelac Dušan: *Hidromašinska oprema*; Fakultet tehničkih nauka-Univerzitet u Novom Sadu 2015.
- [5] Lewis A. Rossman: *EPANET 2 USER MANUEL*

Kratka biografija:



Siniša Sadžakov rođen je u Vrbasu 1987. godine. Diplomski rad iz oblasti građevinarstvo – rečna hidraulika odbranio je 2014. godine na Građevinskom fakultetu u Subotici, Univerzitet u Novom Sadu. Master rad iz oblasti komunalna hidrotehnika odbranio je na Fakultetu tehničkih nauka u Novom Sadu 2018.godine.

РАЧУНАРСКИ ПОДРЖАНО (CAD) ПРОЈЕКТОВАЊЕ САОБРАЋАЈНИЦА- ПРИМЕР ПРИМЕНЕ СОФТВЕРА PLATEIA НА ПРОЈЕКТУ ПУТА ГРГУРЕВЦИ – БЕШЕНОВО**COMPUTER AIDED (CAD) DESIGN OF ROADS - APPLICATION OF PLATEIA SOFTWARE ON THE EXAMPLE OF THE ROAD DESIGN GRGUREVCI – BEŠENOVO**

Урош Ивковић, Небојша Радовић, Факултет Техничких Наука, Нови Сад

Област - ГРАЂЕВИНАРСТВО

Кратак садржај - У овом раду су на примеру путне везе између насеља Гргуревци и насеља Бешеново приказане могућности примене програмског пакета Plateia. Описана је комплетна процедура пројектовања путева, као и алати који се користе и који помажу у дефинисању будуће трасе пута, уздужног профила трасе са нивелетом пута, дефинисању карактеристичних попречних профила, цртање попречних профила и обрачун количина радова.

Кључне речи: пројектовање, пут, Plateia, CAD

Abstract – In this paper, on the example of a road connection between the village of Grgurevci and the village of Bešenovo, the possibilities of applying the Plateia program package are shown. The complete road design procedure is described, as well as the used tools which help in defining of the layout, longitudinal profile of the route with the road level, defining characteristic cross-sections, drawing of cross-section and calculating of quantities.

Keywords: design, road, Plateia, CAD

1. УВОД

Процес планирања, пројектовања, грађења, одржавања и експлоатације путева представља хијерархијски уређен низ активности. Сам процес се, зависно од врсте и нивоа инвестиције, мора водити по строго дефинисаним процедурама, а то су:

- Студија мреже,
- Генерални пројекат,
- Идејни пројекат,
- Главни пројекат,
- Извођачки пројекат,
- Архивски пројекат.

Одговорност пројектанта јесте да изабране конструктивне параметре усклади са сигурношћу и потребама корисника, да усклади објекат са захтевима аспекта животне средине и да разматрањем трошкова, функционалности и ефикасности изабраних параметара, пројектује објекат.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био в. проф. др Небојша Радовић.

2. ФАЗЕ ПРОЈЕКТОВАЊА И ИЗГРАДЊЕ ПУТЕВА

Пројектовање и изградња пута одвијају се кроз следеће фазе:

- Припремне радове за потребе пројектовања путева
- Пројектовање пута
- Извођење радова на изградњи путева

Фазе се међусобно разликују по времену трајања, обиму послова, финансијских средстава неопходних за успешну реализацију као и по степену ангажовања стручњака одређеног профила.

Основни предуслов за приступање наредној фази јесте успешна и тачна реализација претходне фазе.

2.1 Припремни радови за потребе пројектовања путева

Овој фази припадају разни истражни радови као што су геолошко и геомеханичко испитивање тла, а затим геодетски радови на изради геодетских подлога за потребе пројектовања путева.

Поменути геодетски радови подразумевају: прибављање постојеће геодетске документације од надлежног органа, успостављање геодетске мреже, тј геодетске основе, геодетско снимање терена и израду геодетских подлога за пројектовање путева.

2.2 Пројектовање путева

Пројектовању путева приступа се након израде геодетске подлоге за потребе пројектовања путева која укључује и формиран дигитални модел терена - ДМТ. Формирање дигиталног модела терена се може обавити уз помоћ команди које пружа програмски пакет који је приказан у овом мастер раду [4].

2.3 Извођење радова на изградњи путева

Пре почетка извођења икаквих радова морају бити решени правно-имовински односи на подручју изградње.

То подразумева да је, ако је то било неопходно, претходно извршена експропријација земљишта. Па се након експропријације приступа изградњи пута.

3. ПРИМЕНА РАЧУНАРА У ПРОЈЕКТОВАЊУ ПУТЕВА

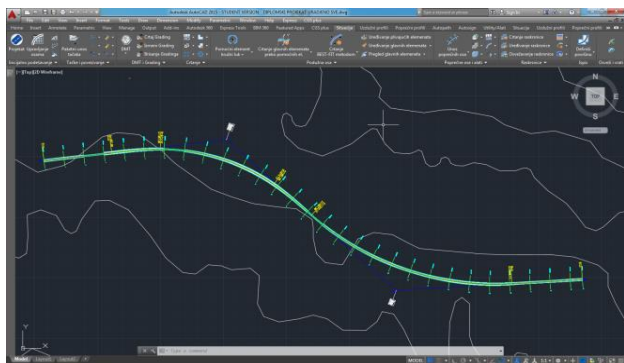
Последњих неколико деценија употреба рачунара у пројектовању путева постала је уобичајена појава. Рачунарски програми су у многоме унапредили пројектантску праксу, јер омогућавају једноставније и брже исцртавање пројектне документације. Програми се могу користити у многим сегментима пројектовања, као што су: трасирање, избор оптималне трасе, дефинисање количине радова...

3.1 Могућности примене програмског пакета Plateia при пројектовању путева

Plateia је софтверски пакет који служи пројектовању нових и пројектовању реконструкција путева. Софтвер нуди велики број команди, од неких једноставних до команди за 3D моделирање које се користе за анализе пројектних решења. Plateia помаже пројектанту у припреми графичких прилога и техничке документације, израчунавању предмерских количина. 3D моделирање омогућава различите облике пројектних анализа, као што су оптичке анализе, услови дренажа воде са путева итд. Plateia је дизајнирана да ради са неколико CAD апликација, међутим када ради у корелацији са AutoCAD Civil 3D-ом, Plateia омогућава кориснику да користи оба програма. Продукт је компаније CGSplus са базом у Словенији, и важно је напоменути да је ово један од програма који се најчешће користи у фирмама у окружењу које се баве пројектовањем путева.

3.2 Радно окружење

Радно окружење Plateia-e је веома препознатљиво, с обзиром да она ради на платформи AutoCAD-a. Тако да се оно састоји из простора за цртање, које и заузима највећи простор, заглавља програма и падајућих менија који су смештени испод заглавља. Осим стандардних менија и алатки које су укључене у пакет AutoCAD-a (Home, Insert, Annotate, Layout, Parametric View, Manage...), Plateia нам пружа још мноштво нових алата и менија који ће бити укратко описани у дипломском – МАСТЕР раду, а приказ радног окружења програма се може видети на слици 1.



Слика 1. Приказ радног окружења програма Plateia 2015

Уз Plateia-у се лако управља великим пројектима који имају дуге трасе и на хиљаде попречних профила. Пројекти нису ограничени ни по величини нити

њихов учинак трпи велики утицај приликом рада при великим пројектима, укључујући веома дуге трасе, подужне профиле и попречне профиле. Попречни профили могу да садрже висок степен детаља са опцијама синхронизације на захтев, имају велику брзину обраде и постојане податке.

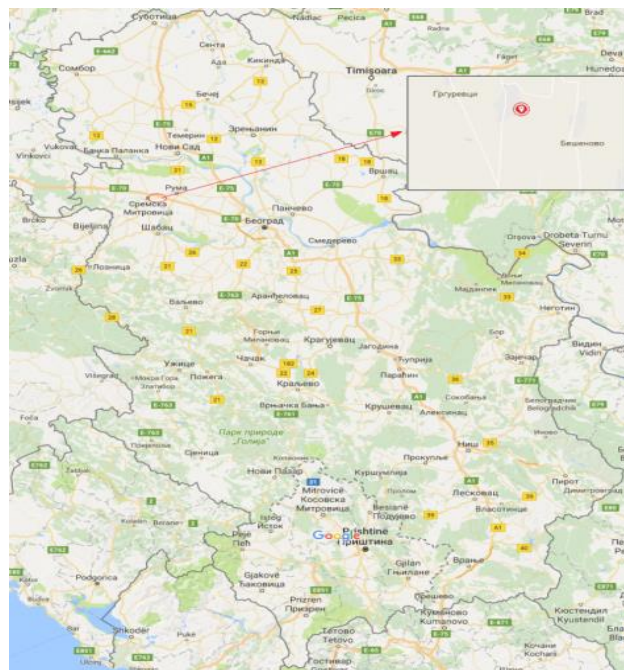
Користећи свој флексибилан, динамички модел података, Plateia подржава BIM (Building Information Model) токове и процесе рада. Пажљиво дизајнирани и креирани Plateia кориснички интерфејс и процеси рада су у сагласности са инжењерском праксом пројектовања путева. То чини Plateia-у изненађујуће лаком за учење и једноставном за коришћење.

4. ПРИМЕР ПРИМЕНЕ ПРОГРАМА PLATEIA ПРИ ПРОЈЕКТОВАЊУ ПУТЕВА

4.1 Увод

Предмет дипломског рада је израда путне везе између Гргуреваца и Бешенова у приближној дужини од 2860 m који почиње на km 0+000.00, а завршава се на km 2+860.32. Потребно је да се изврши изградња нове трасе кроз ненасељено место које представља пољопривредно обрадиво и необрадиво земљиште. С обзиром да је у питању нова траса захтева се да то буде савремена коловозна конструкција, а да се при томе поштују и сви стандарди у погледу екологије и заштите животне средине. Положај трасе је приказан на слици 2.

Пројектоване су 2 хоризонталне S кривине и 4 вертикалне кривине, две конкавне и две конвексне. Траса је пројектована на начин да у потпуности задовољи аспекте безбедности и удобности, економичности и уклапања у окружење. Посебно се водило рачуна, колико су то омогућавала геометријска ограничења, о избегавању квалитетних обрадивих површина и трасирању по необрађеном и запуштеном земљишту.



Слика 2. Географски положај пројектне трасе

4.2 Анализа саобраћаја

Пројектни параметри пута су:

- Везни пут
- ПГДС = 5330 воз/дан

На основу пројектних параметара рачунају се пројектни елементи пута: категорија пута, рачунска и претходна брзина, ширина саобраћајних и ивичних трака, ширина банкина и ригола...

Пројектним задатком се дефинише пројектни период од 20 година.

4.3 Елементи пројектне геометрије

Програмски услови према важећем Правилнику [3] се дефинишу преко:

- Класификације пута,
- Саобраћајне функције,
- Мередавне брзине и
- Мередавних параметара возача, возила и пута.

Класификација пута се одређује према административној и функционалној класификацији. По овој класификацији разликују се четири врсте путева према основним функцијама:

- Приступни пут (ПП)
- Сабирни пут (СП)
- **Везни пут (ВП)**
- Даљински пут (ДП)

А зависно од просторног нивоа на коме се одвија функција пута, четири наведене врсте путева формирају осам функционалних типова:

- Приступни путеви локални (ПП-л)
- Приступни путеви подручни (ПП-п)
- Сабирни путеви подручни (СП-п)
- Сабирни путеви регионални (СП-р)
- **Везни путеви регионални (ВП-р)**
- Везни путеви међурегионални (ВП-м)
- Даљински путеви међурегионални (ДП-м)
- Даљински путеви државни, међудржавни (ДП-д)

На основу наших података закључујемо да се у овом предметном пројекту ради о **државном путу II реда (ВП-р)**.

4.4 Елементи ситуационог и нивелационог плана

Задата деоница је решена са две хоризонталне кривине следећих радијуса:

- $R_1 = 974,728 \text{ m}$
- $R_2 = 1344,118 \text{ m}$

Целокупно решење нивелационог плана приказано је у подужном профилу у размери 1:100/1000. У подужном профилу дате су све коте новопроектваног коловоза.

Подужно вођење трасе је условљено стањем на терену. С обзиром да се ради о равничарском терену на коме се вода дуго задржава и јако споро повлачи са терена, неопходно је издићи нивелету изнад терена и

трасу поставити у насип како би се избегли проблеми са водом, као и ефикасно одвођење како површинске, тако и воде из трупа пута.

4.5 Анализа саобраћајног оптерећења

Мередавно саобраћајно оптерећење је срачунато за експлоатациони период од 20 година, што правилник SRPS U.C4.012, када се ради о димензионисању асфалтних коловозних конструкција, поставља као правило.

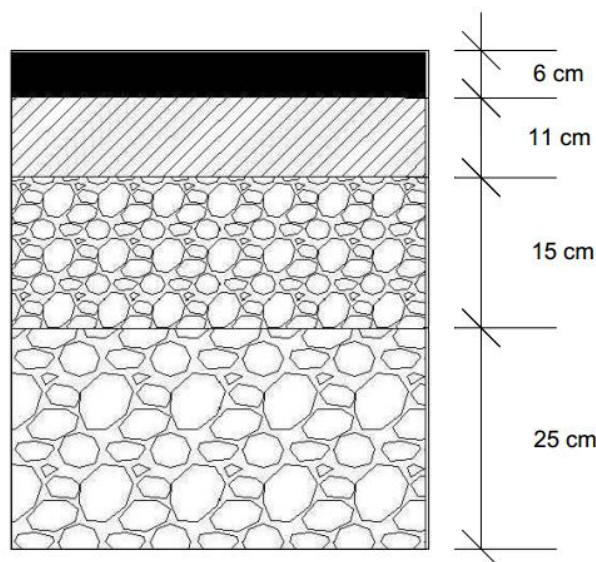
Просечан годишњи дневни саобраћај (ПГДС) за предметну деоницу износи 5330 возила/дан.

Пројектни услови су:

- Пројектни период $p = 20$ година
- Просечна искоришћеност носивости возила је 70%
- Просечна годишња стопа раста броја тешких теретних возила (ТТВ) износи $r = 4\%$
- Фактор раста саобраћаја у пројектном периоду од 20 година, $q = 31\%$

4.6 Димензионисање коловозне конструкције

Димензионисање коловозне конструкције урађено је према важећем SRPS.U.C4.012. стандарду. Димензионисање нових асфалтних конструкција према критеријуму носивости је поступак при пројектовању којим се одређује састав и дебљина коловозне конструкције на основу мередавних параметара за димензионисање. Под појмом асфалтна конструкција подразумева се вишеслојна конструкција која се састоји од асфалтног застора и носивих слојева (горњих и доњих), а графички приказ тих слојева се може видети на слици 3.



Слика 3. Графички приказ слојева коловозне конструкције

AB-11s	$d = 6 \text{ cm}$
BNS-32s	$d = 11 \text{ cm}$
дробљени камени агрегат 0/31 mm	$d = 15 \text{ cm}$
дробљени камени агрегат 0/63 mm	$d = 25 \text{ cm}$
$\Sigma = 57 \text{ cm}$	

5. ЗАКЉУЧАК

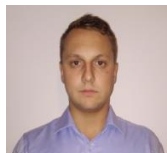
У предметном дипломском - мастер раду је приказано како се програмски пакет Plateia може применити при пројектовању путева и саобраћајница.

Као генерални закључак истиче се једна битна чињеница, а то је да су и Plateia и сви остали програмски пакети тог типа у великој мери убрзали процес израде пројектних задатака и значајно помогли пројектанту у управљању пројектима и као такви од изузетне су драгоцености у области пројектовања путева и градских саобраћајница у 21-ом веку.

6. ЛИТЕРАТУРА

- [1] Радовић Н.: Скрипта са предавања из предмета:
Одабрана поглавља из пројектовања путева,
Писана предавања, Факултет Техничких Наука,
Нови Сад, 2012/2013
- [2] Узелац Ђ.: *Коловозне конструкције*, Факултет
Техничких Наука, Нови Сад, 2014.
- [3] Правилник о условима које са аспекта безбедности
саобраћаја морају да испуњавају путни објекти и
други елементи јавног пута, Београд, Службени
гласник Р. Србије 41/09 и 53/10
- [4] Просен Д., Драгић П.: *Основе програма Plateia и
Ferrovia, CGSplusD.O.O.*, Љубљана 2015
- [5] Приручник за пројектовање путева, ЈП Путеви
Србије

Кратка биографија:



Урош Ивковић, рођен је у Вршцу 1991. године. Мастер рад на Факултету техничких наука из области Грађевинарства – Путеви, железнице и аеродроми је одбранио 2018. године. контакт: uros_ivkovic@hotmail.com



Небојша Радовић, рођен је у Београду 1962. год.. Докторирао је на Факултету техничких наука у Новом Саду 2006. год., а од 2015. год. је у звању ванредног професора на Катедри за геотехнику и саобраћајнице.

ANALIZA KARAKTERISTIKA POVREDA NA RADU NA GRADILIŠTIMA REPUBLIKE SRBIJE**ANALYSIS OF THE CHARACTERISTICS OF WORK INJURIES ON THE CONSTRUCTION SITES IN THE REPUBLIC OF SERBIA**

Milena Senjak, Vladimir Mučenski - Fakultet tehničkih nauka, Novi Sad

Oblast - GRAĐEVINARSTVO

Kratak sadržaj – Na osnovu prijava povreda na radu, formirana je baza podataka koja je kasnije služila za detaljniju analizu. Obrada podataka iz baze nas upućuje na povećane rizike povreda na određenim vrstama radova, u određenom starosnom dobu radnika, deo tela koji je povređen, načine povređivanja i dr. Izvor prijavljenih povreda na radu je Ministarstvo za rad, zapošljavanje, boračka i socijalna pitanja, odnosno Uprava za bezbednost i zdravlje na radu.

Ključne reči: Baza podataka, povreda na radu, rizici.

Abstract - Data base which later served for detailed analysis was based on reports of work injuries. Processing of informations from data base guide us to increased risks of injuries on specific kinds of workes, in the specific age of the workers, part of the body which is injured, the way of injuring of the workers and etc. The source of the reported work injuries is Ministry of labour, employment, veterans' affairs and social affairs, safety and health at work.

Key words: Database, injury at work, risks

1. UVOD

Građevinarstvo je delatnost u kojoj su povrede na radu gotovo svakodnevne i nažalost, sve učestalije. U ovoj grani privrede je najveći broj smrtnih slučajeva što je u glavnom posledica nepridržavanja osnovnih propisa zaštite na radu. Osnovni zadatak zaštite na radu jeste normalno, nesmetano i bezbedno odvijanje radnih zadataka u nekoj radnoj sredini.

Radna sredina obuhvata prostor, mašine, pribor, uređaje, tehnološki postupak, kao i uslove u kojima se radi (temperatura, buka, zračenje, vlažnost, štetni gasovi, dim itd.). Interes društva, kao i svakog pojedinca je da se ostvari najviši nivo bezbednosti i zdravlja na radu, da se neželjene posledice kao što su povrede na radu, profesionalne bolesti i bolesti u vezi sa radom svedu na najmanju moguću meru, odnosno da se ostvare uslovi rada u kojima bi zaposleni imao osećaj zadovoljstva pri obavljanju svojih profesionalnih zadataka.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji je mentor bio doc. dr Vladimir Mučenski.

2. DEFINICIJA RIZIKA

Građevinarstvo je veoma složen proces koji sačinjava niz aktivnosti i događaja, kao i samih učesnika koji te aktivnosti realizuju. Različiti resursi, velika finansijska ulaganja, veliki obimi poslova dovode do pojave neizvesnosti oko realizacije definisanog cilja, odnosno rizik od pojave nepredviđenih situacija koje će biti nepogodne za postizanje cilja. To se može nazvati rizik projekta.

Da bi smo te rizike sveli na minimum, moramo ih prvo predvideti, a to ćemo najlakše učiniti detaljnijom analizom projekta, odnosno sistemskim pristupom projektu. Ovakav proces kontrole nad rizikom nazivamo upravljanje rizicima.

Rizik je neizvestan događaj ili stanje koje, ukoliko se dogodi, može pozitivno ili negativno uticati na ciljeve projekta (vreme, budžet, obim, kvalitet).

3. KARAKTERISTIKE PROCESA IZGRADNJE SA ASPEKTA BEZBEDNOSTI I ZDRAVLJA NA RADU**3.1. Građevinski objekat kao jedinstven proizvod**

Osnovna specifičnost kada je proces izgradnje u pitanju, u odnosu na procese drugih industrija, jeste unikatnost, odnosno specifičnost samog proizvoda. Što je objekat koji se gradi veći, teže je kontrolisati njegovu izgradnju, sama kompleksnost procesa znatno utiče na bezbednost i zdravlje na radu.

Istraživanja su pokazala da samo prenos materijala na gradilištu, bez da radnici učestvuju u ručnom prenosu, izazove oko 29% povreda na radu. Iz toga se može zaključiti da je jako važno organizovanje samog gradilišta. (Mučenski, 2018)

Značajni faktori koji utiču na bezbednost zdravlja na radu jesu: uslovi rada, temperatura i vlažnost, uticaj vetra, buka, vibracije i prašina.

4. BAZA PODATAKA POVREDA NA RADU-MODEL BAZA

Ne samo građevinarstvo, već svaka grana industrije generiše rizike zaštite na radu, a sam broj opasnih situacija proističe iz organizovanosti procesa rada da se omogući bezbedan rad. Obučenost radnika da obavljaju neku operaciju na bezbedan način takođe utiče na pojavu opasne situacije. Što nam je proces složeniji to su rizici

veći, i više vremena je potrebno utrošiti da bi se taj proces organizovao na bezbedan način.

Iako je proces građenja veoma složen, moguće ga je definisati u potpunosti, ali samo uz sistematičan pristup, kada će svi elementi biti definisani i identifikovani, a onda i njihove međusobne veze. Prilikom upravljanja rizicima zaštite na radu bitno je uraditi prethodno napomenuto, odnosno identifikovati elemente procesa i njihovu povezanost.

Ukoliko se neki rizik ne prepozna, njemu se ne posvećuje nikakva pažnja, niti se prati tokom projekta, iz tih razloga dolazi do pojave novih opasnosti

Osnovna identifikacija rizika se vrši iz istorijskih baza podataka, članovima menadžment tima, svim članovima projekta (radnici preduzeća koji izvode radove, podizvođači, dobavljači...), eksternim stručnjacima, itd. Svaki od navedenih činilaca jeste potencijalan izvor znanja i iskustva u identifikaciji rizika. Iako je tako, ukoliko se rizici iz prethodno realizovanih procesa građenja ne sačuvaju u pismenoj ili elektronskoj formi velika je verovatnoća da će biti izgubljeni, odnosno neiskorišćeni. Prema tome kvalitetne baze podataka su veoma značajan resurs u procesu kontrole rizika zaštite na radu.

4.1. Formiranje baze podataka povreda na radu

Za potrebe istraživanja izvršeno je formiranje baze podataka povreda na radu koje su nastale prilikom realizacije građevinskih radova.

Ova baza podataka je formirana sa ciljem definisanja potencijalnih izvora rizika, njihove povezanosti sa različitim vrstama radova (zemljani, betonski, tesarski, itd.), korišćenim materijalima, alatima, mehanizacijom kao i karakteristikama radne snage.

Podaci koji sačinjavaju bazu su dobijeni iz Ministarstva za rad, zapošljavanje, boračka i socijalna pitanja, odnosno iz Uprave za bezbednost i zdravlje na radu. Podaci se odnose na građevinska preduzeća Republike Srbije i obuhvataju informacije o povredama koje su nastale u period od 2013. godine do 2017. godine.

Povrede koje su bila analizirane u svrhe ove baze podataka, određene su na osnovu člana 51. stav 2. Zakona o bezbednosti i zdravlja na radu ("Službeni glasnik RS", broj 101/05), prema Pravilniku o sadržaju i načinu izdavanja obrasca izveštaja o povredi na radu i profesionalnom oboljenju. ("Službeni glasnik RS", broj 72/06, 84/04-ispravka i 4/16).

Baza podataka povreda nastalih na gradilištima Republike Srbije, prilikom realizacije radova u visokogradnji formirana je na osnovu podataka iz 113 prijava povreda u okviru kojih su definisani svi zahtevani podaci.

4.2. Struktura baze podataka povreda na radu

Baza podataka je pravljen tako da sadrži informacije o što većem broju parametara potrebnih za identifikaciju rizika zaštite na radu, a same informacije su zavisile od dostupnih podataka iz obrazaca za prijavu povreda.

Bazu podataka čini pet grupa podataka i to:

- grupa podataka o povređenom radniku,
- grupa podataka o vremenskoj dimenziji nastale povrede,
- grupa podataka o vrsti radova i radnim operacijama realizovanim u trenutku povređivanja,
- grupa podataka o izvoru povrede, uzrocima povrede i načinu povređivanja
- grupa podataka o posledicama povređivanja.

Podaci iz novoformirane baze podataka mogu se iskoristiti za određivanje verovatnoće i veličina posledica povrede. Pri tome, grupe podataka o povređenom radniku, vrsti radova i radnim operacijama kao i načinu povređivanja i uzrocima povrede mogu se numerički povezati sa veličinom verovatnoće nastupanja povrede. S druge strane, grupa podataka o posledicama povređivanja određuje parametar veličine posledica. Parametar o vremenskoj dimenziji povrede u kombinaciji sa dinamičkim planovima (godišnji, mesečni, sedmični i dnevni planovi) može ukazati na povećanu verovatnoću povređivanja za pojedine vrste radova tokom godine, meseca, radne sedmice ili radnog dana.

5. ANALIZA PODATAKA

Podatke iz prijave povreda na radu koje smo uneli u bazu podataka smo dalje analizirali kako bismo dobili konkretne rezultate, iskoristive u svrhe identifikacije rizika povreda na radu u okviru građevinskih projekata.

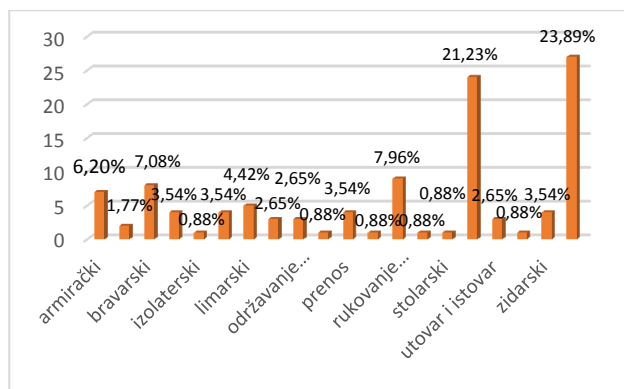
Analizirali smo:

- Vrste (grupe) radova na kojima su se dešavale povrede
- Starosnu dob radnika koji su doživeli povrede
- Težinu povrede koja je nastala
- Deo tela koji je povređen
- Način povređivanja

5.1. Vrste (grupe) radova na kojima su se dešavale povrede

Najveći broj povreda desio se na tesarskim radovima i iznosi 21,23%, kao i na nepoznatim radovima, koji nisu uspešno definisani zbog nedostatka podataka prilikom prijave povrede, a ima ih čak 23,89%. U poglavlju 9 je rečeno da upravo usled propusta evidentiranja povreda na radu imamo toliko slučajeva svrstanih u kategoriju nepoznatih radova.

Najčešći slučaj povređivanja, kad su u pitanju tesarski radovi, jeste prilikom demontaže oplata. Tada dođe do preturanja skele, klizanja merdevina, pada podupirača, pada dela oplata ili slično i to obično dovede do pada radnika na nivo ispod, a zatim i do njegovog povređivanja. Drugi najzastupljeniji slučaj povređivanja jeste prilikom krojenja oplata ili skraćivanja letvi, odnosno prilikom rada na kružnoj testeri.

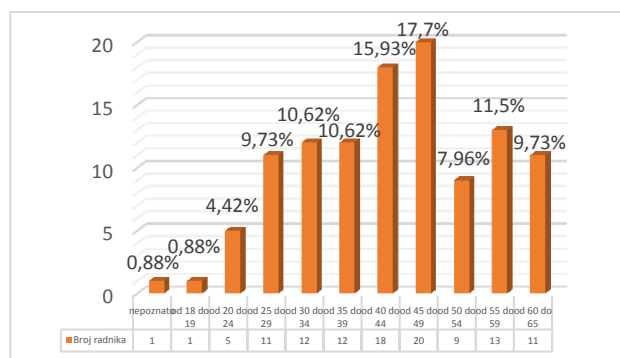


Slika 5.1. Prikaz povreda po vrstama radova

5.2. Starosna dob radnika prilikom povređivanja

Prema analiziranim podacima, možemo da zaključimo da su radnici koji pripadaju starosnoj dobi između 40 i 50 godina najpodložniji povredama. Zaključeno je da je 15.93% povreda zabeleženo kod starosne dobi između 40 i 44 godine, a 17.7% kod starosti radnika između 45 i 49 godina. Iako mlađi radnici, sa manje iskustva generišu veće rizike, u ovoj analizi se pokazalo drugačije.

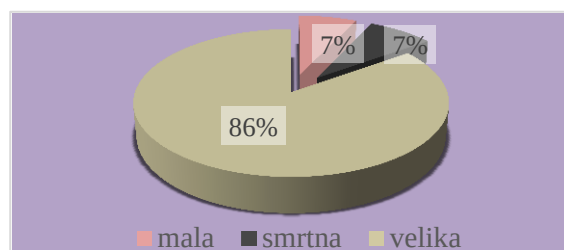
Prema istraživanjima zaključeno je da su radnici sa više iskustva svesniji rizika od povrede na poslovima u građevinarstvu, što dovodi do toga da će pre prihvatiti lična sredstva zaštite kao obavezan segment u poslu.



Slika 5.2. Procentualni prikaz povreda na radu u zavisnosti od starosti radnika

5.3. Težina povrede na radu

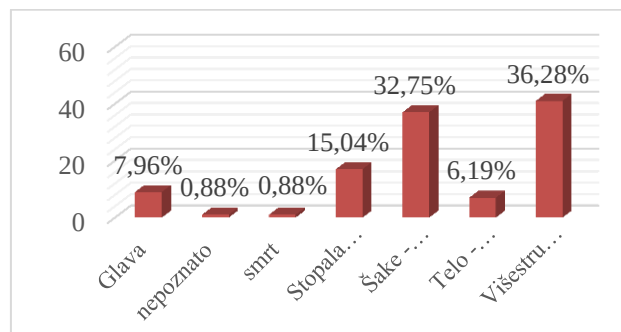
Kada su u pitanju težine povrede koje su analizirane najveći broj povreda je okarakterisano kao velika (teška) povreda na radu, takvih je čak 86%. Smrtnih je registrovano 7%, a malih 7%.



Slika 5.3. Prikaz povreda na radu prema veličini nastale povrede

5.4. Deo tela koji je povređen

Višestruke povrede tela su najučestaliji ishod i to se desilo u 36% slučajeva, takođe jako često stradaju šake, odnosno ruke što je zabeleženo u 33% slučajeva. Posebnu pažnju treba posvetiti i povredi nogu, odnosno stopala, jer je prema analizama zabeleženo 15% povreda upravo stopala. Najčešći slučaj stradanja šake-ruku, jeste u toku tesarskih radova. Preko 30% povreda šake je posledica upravo tesarskih radova, za preko 20% uzrok je rukovanje mehanizacijom, a među preostalim povredama se nalaze ostali radovi.



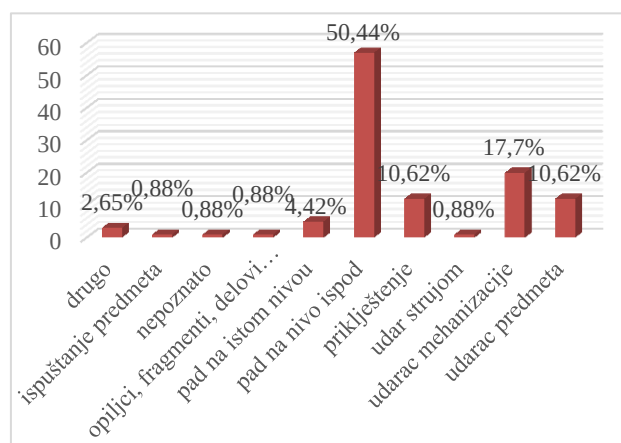
Slika 5.4. Prikaz dobijenih rezultata pri analizi povređenih delova tela

5.5. Način povređivanja

Očigledno je najučestaliji slučaj načina povređivanja jeste pad na nivo ispod i uzima 50,44% od ukupnih. Pad na nivo ispod obuhvata povrede nastale usled pada radnika na površinu koja se nalazi ispod površine na kojoj je započet pad.

Neke od najčešćih situacija prilikom pada na nivo ispod su:

- Pad sa merdevina ili skele prilikom demontaže oplata
- Pad sa merdevina ili skele prilikom molersko farbarskih radova
- Pad prilikom izlaska iz neke građevinske mašine
- Pad sa prikolice kamiona prilikom utovara – istovara
- Pad sa visine prilikom postavljanja električnih instalacija i dr.



Slika 5.5. Prikaz dobijenih rezultata pri analizi načina povređivanja

6. ZAKLJUČAK

Pronalazak uzroka, odnosno definisanje odgovornosti za nastale povrede važno je kako bi se identifikovali propusti procesa izgradnje, a onda ti propusti korigovali. Tako treba utvrditi i da li je povreda nastala usled greške preduzeća ili usled nebezbednog rada i ponašanja radnika.

Formiranje ovakve baze, mi na savremen način identifikujemo rizike koji se javljaju na projektima i njima upravljamo. Kako ne bismo imali pogrešne upravljačke akcije, moraju biti pažljivo i detaljno analizirane i tumačene. Iz tog razloga, za potrebe ovog istraživanja je formiran Model BAZA tj. baza podataka koja obuhvata 113 povreda nastalih na gradilištima Republike Srbije prilikom izgradnje objekata visokogradnje. Podaci se odnose na građevinska preduzeća Republike Srbije i obuhvataju informacije o povredama koje su nastale u periodu od 2013. godine do 2017. godine.

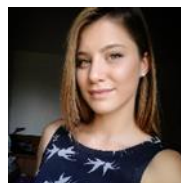
U okiru analize podataka vidimo koji su to rizični radovi, rizične grupe radnika gledajući njihovu starost, način povređivanja i povrede koje nastaju, kao i težinu povrede, odnosno posledice. Povezivanjem ovih podataka mi možemo da predupredimo naše upravljanje rizicima zaštite na radu u okviru građevinskih projekata, umanjimo ih ili potpuno eliminišemo. Zaštita na radu je oduvek bila jedan od osnovnih problema kada je realizacija građevinskih projekata u pitanju. Same zakonske mere nisu dovoljne da bi kvalitet zaštite bio na zadovoljavajućem nivou, prema tome treba poduzeti dodatne mere. Obzirom da je proces izgradnje veoma složen, ta složenost generiše veliki broj rizika na radu.

Pravljenjem ovakvih baza podataka, znatno bi se unapredila identifikacija rizika, upravljanje tim rizicima i zaštita radnika u građevinarstvu.

7. LITERATURA

- [1] Mučenski V. (2013), "Model semikvantitativne procene rizika zaštite na radu za procese izgradnje"- doktorska disertacija, Univerzitet u Novom Sadu, Fakultet tehničkih nauka
- [2] Mučenski V. (2018), "Teorijske osnove bezbednosti i zdravlja na radu u građevinarstvu", Univerzitet u Novom Sadu, Fakultet tehničkih nauka
- [3] Jelena Markičević, "Bezbednost i zdravlje na radu u građevinarstvu u oblasti visokogradnje", VPTŠ Užice
- [4] Zakon o bezbednosti i zdravlju na radu, ("Službeni glasnik RS 101/05 i 91/15") 2005, 2015. godina
- [5] Andrejević M. (2016), Sigurnost i zaštita u građevinarstvu, Završni rad
- [6] Pravilnik o sadržaju i načinu izdavanja obrasca izveštaja o povredi na radu i profesionalnom oboljenju. ("Službeni glasnik RS", broj 72/06, 84/04- ispravka i 4/16).
- [7] Pravilnik o evidencijama u oblasti bezbednosti i zdravlja na radu ("Službeni glasnik RS", broj 62/07 i 102/15)

Kratka biografija:



Milena Senjak rođena je u Somboru 1994. godine. Diplomirala je 2017. godine, na Fakultetu tehničkih nauka, iz oblasti Građevinarstva, smer Konstrukcije. Master rad na Fakultetu tehničkih nauka iz oblasti Građevinarstva - Organizacija i tehnologija građenja odbranila je 2018. godine.

PROCENA STANJA, DOGRADNJA I ENERGETSKA SANACIJA PORODIČNE STAMBENE ZGRADE**ASSESSMENT, RAISING AND ENERGY RENEWAL OF THE FAMILY RESIDENTAL BUILDING**

Nataša Rudakijević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – GRAĐEVINARSTVO

Kratak sadržaj – U radu je prikazna procena stanja stambene zgrade S+P u Veterniku. Radi utvrđivanja stepena oštećenja kao i vrste sanacionih radova sproveden je makroskopski pregled dostupnih elemenata. Na osnovu analize registrovanih oštećenja i proračuna energetske efikasnosti dat je opis sanacionih mera u cilju povećanja trajnosti i unapređenja energetske efikasnosti objekta. Predviđena je i nadogradnja objekta.

Ključne reči – procena stanja, energetska efikasnost, oštećenja, sanacija, termoizolacioni materijali, gas beton
Abstract – In this paper the assessment of a residential building in Veternik is shown. In order to determine the level and cause of possible damages as well as the type of repairing measures a visual inspection of structural elements was undertaken. Based on this examination and the energy efficiency calculation detailed description of repairing measures is given with the main goal of improving energy efficiency and structural durability. Adding of one additional floor is planned.

Keywords – assesment, energy efficiency, damages, repairing, thermal insulation, cellular concrete

1. UVOD

Rad se sastoji iz dva dela, teorijsko-istraživačkog i stručnog dela. U prvom delu rada, koji je teorijsko-istraživačkog karaktera, analizirani su građevinski materijali – gas beton i Mutipor.

Prikazan je istorijat gas betona, čiji je izumitelj Aksel Erikson. Gas beton je poznat i pod nazivom Siporeks, Ytong, Durox i Hebel. Objasnjen je proces proizvodnje, a potom svojstva gas betona. To je materijal koga karakterišu brojne pozitivne osobine.

Navedena su područja primene kao i asortiman proizvoda dostupnih u Republici Srbiji. Prikazane su svojstva, asortiman proizvoda i primena Mutipor termoizolacionih ploča.

Stručni deo rada obuhvata procenu stanja jednospratnog stambenog objekta koji se nalazi u Veterniku. Stručnim delom rada prikazan je detaljan proračun energetske efikasnosti objekta pre i nakon izvršene energetske sanacije, kao i projekat dogradnje objekta.

2. GAS BETON**2.1. Istorijat**

Gas beton, kakav poznajemo danas, prvi je napravio švedski arhitekta i inovator Johan Aksel Erikson. Proizvod je patentiran 1924 godine. Prva industrijska linija za proizvodnju gas betona započela je rad 1929. godine u švedskom gradu Yxhult.

2.2. Svojstva

Gas beton ima mnogo pozitivnih karakteristika:

- Mineralnog porekla i neškodljiv građevinski materijal. Procenat prirodne radioaktivnosti je daleko manji od drugih građevinskih materijala. Zapreminska masa gas betona je 400 - 700kg/m³.
- Zvučna i termoizolacija: Zahvaljujući poroznoj strukturi, gas beton ima dobre termičke i zvučne izolacione karakteristike. Koeficijent toplotne provodljivosti siporeksa je oko $\lambda = 0,14 \text{ W/mK}$.
- Nosivost: Nosivost gas betona raspoređena je po klasama čvrstoće **2,5/0,4; 3/0,45; 4/0,5; 6/0,65** s područjem čvrstoće na pritisak od **2,5 do 6 N/mm²**. Zahvaljujući ćelijastoj strukturi obezbeđeno je postizanje jednake čvrstoće u svim smerovima bloka.
- Zaštita od požara: Gas beton ne sadrži štetne supstance u gasovitom stanju i već kod malih debljina zidova zadovoljava zahteve za otpornost na požar prema propisima i ne traži dodatne mere zaštite. Zbog niske toplotne provodljivosti porast temperature na strani zida suprotnoj od vatre je mali.
- Otpornost pri zemljotresu: Zahvaljujući maloj težini i sposobnosti da preuzme seizmičke sile iz svih smerova, gas beton ima dobru otpornost na zemljotres.
- Paropropusnost: sastav gas betona i njegova ćelijasta struktura omogućavaju izuzetnu paropropusnost i daju mu prednost u odnosu na druge građevinske materijale na tržištu.

2.3. Primena

Gas beton, kao materijal se može primenjivati u svim područjima građenja – od jednostavnog oblaganja i uređenja enterijera do zahtevnih masivnih građevina.

Objekti koji se najčešće grade upotrebom blokova od gas betona su: porodične kuće, stambeno-poslovni objekti, industrijski objekti, kao i adaptacije, nadogradnje i modernizacije postojećih objekata.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila dr Mirjana Malešev, red. prof.

3. MULTIPOR

Multipor je mineralna termoizolaciona ploča proizvedena sa ciljem bolje izolacije građevina. Kao i gas beton, multipor spada u neškodljive građevinske materijale.

3.1. Svojstva

Multipor je materijal koji spada u termoizolacione materijale mineralnog porekla. Proračunski koeficijent toplotne provodljivosti je $\lambda=0,045\text{W/mK}$. Multipor je dobar zvučni izolator. Stabilnog je oblika, nestišljiv i otporan na pritisak. Lako se obrađuje alatima za drvo, što ga čini praktičnim u gradnji. Multipor je negoriv materijal. Otporan je na vlagu i paropropustan.

3.2. Multipor proizvodi

U Multipor proizvode, pored Multipor poloča, ubrajamo i Multipor laki tankoslojni malter i mrežicu za armiranje maltera.

3.3. Područja primene

Multipor ploče se mogu koristiti za izolaciju međuspratnih konstrukcija sa donje, ali i sa gornje strane, spoljnih zidova sa spoljašnje ili unutrašnje strane, kao i za rešavanje termičkih mostova na skeletnim konstrukcijama.

4. PROCENA STANJA OBJEKTA

4.1. Opis objekta

Predmetni objekat je porodična stambena zgrada, spratnosti S+P (suteran i prizemlje). Objekat je približno pravougaone osnove, spoljašnjih dimenzija 17,25 x 12,25 m. Bruto površina osnove objekta je 197 m², a neto površina iznosi 151,67 m².

Objekat je građen u kombinovanom sistemu sa nosećim zidovima od opeke i stubovima od armiranog betona, na koje se oslanja kruta tavanica koju čine armiranobetonske grede i armiranobetonska sitnorebrasta ploča (TM tavanica). Fundiranje je izvršeno na trakastim temeljima ispod zidova i temeljima samcima ispod stubova, koji su povezani kontra gredom. Ploča iznad suterena i ploča iznad prizemlja izvedene su kao sitnorebraste ploče – TM tavanice.

4.2. Vizuelni pregled konstrukcije

U okviru procene stanja konstrukcije zgrade, obavljen je vizuelni pregled svih dostupnih elemenata konstrukcije. Prilikom pregleda, nisu uočena značajnija oštećenja na fasadi objekta. Registrovana oštećenja su postojala prilikom građenja, došlo je do neznatnog pogoršanja stanja fasadnih blokova. Korišćeni su fasadni blokovi van klase. Vizuelnim pregledom nisu uočena oštećenja ni na perlitnom malteru, niti na kamenoj oblozi. Unutrašnjost objekta u prizemlju je omalterisana i vizuelnim pregledom nije bilo moguće uočiti postojanje oštećenja koja bi narušila integritet konstrukcije. Vizuelnim pregledom elemenata konstrukcije u suterenu, na elementima koji su dostupni za pregled, uočeni su defekti i oštećenja:

- Na pojedinim gredama iznad suterena uočena je neravna površina, nedovoljan zaštitni sloj betona (vidljive uzengije), odlomljene ivice, oslikavanje oplata, linijska segregacija (Slika 1)

- Na svim stubovima dostupnim za pregled uočeni su nepravilno izveden prekid i nastavak betoniranja, zone nedovoljno kompaktiranog betona, linijska segregacija i oslikavanje oplata. (Slika 2)



Slika 1. Detalj grede – odlomljene ivice, otkrivena armatura sa površinskom korozijom

Podna ploča u suterenu je nearmirana, betonirana na podlozi. Na delovima koji su dostupni za pregled, uočene su prsline usled dugotrajnog hidrauličkog skupljanja. Širina prsline se kreće od 0,3 do 0,4 mm. Na drvenoj građi krovne konstrukcije nisu uočena oštećenja, nema buđi, isušivanja ili pucanja. Na keramičkim crepovima uočeno je iscvetavanje soli. Nema tragova procurivanja



Slika 2. Nepravilno izveden prekid i nastavak betoniranja stuba i grede u suterenu

4.3. Nedestruktivna ispitivanja

Nedestruktivnim metodama ispitana je dubina karbonizovanog sloja betona i izmerena je površinska tvrdoća betona upotrebom sklerometra. Karbonizacijom je zahvaćen samo tanak površinski sloj betona (1-3 mm), pa možemo zaključiti da nema prisutne karbonizacije zaštitnog sloja betona. Ispitivanje sklerometrom je odrađeno na 8 nasumično odabranih mesta, četiri puta na stubovima, dva puta na gredama, jedno mesto na kontragredi i jedno na podnoj ploči.

Tabela 1. Primer ispitivanja sklerometrom na stubu

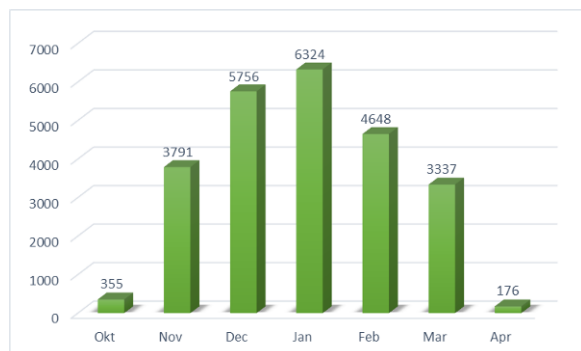
Mesto ispitivanja	Stub POS S02 u osi C4			
	(istočna strana)		(severna strana)	
Veličine odskoka	42	41	40	39
	36	41	44	43
	36	38	38	45
	43	41	39	41
	37	39	40	45
	42	38	40	44
Indeks sklerometra	39,50		41,50	
Čvrstoća pri pritisku	40±7,0		43±7,2	

4.4. Zaključak

Na osnovu analize podataka dobijenih vizuelnim pregledom dostupnih elemenata konstrukcije, te rezultata nedestruktivnih ispitivanja, zaključeno je da nosivost, stabilnost, trajnost, upotrebljivost i funkcionalnost konstrukcije nisu narušeni.

5. PRORAČUN ENERGETSKE EFIKASNOSTI

Pri proračunu energetske efikasnosti urađen je kompletan proračun toplotne provodljivosti građevinskih elemenata koji čine termički omotač zgrade, proračun difuzije vodene pare, proračun gubitaka i dobitaka toplote, te proračun godišnje potrebne finalne energije za grejanje. Na slici 3 prikazan je dijagram potrebne energije u kWh za grejanje po mesecima. Ukupna godišnja potrebna energija za grejanje iznosi 24387 kWh/a, dok je specifična potrebna godišnja energija 167 kWh/m²a. Ovim proračunom je zaključeno da je postojeći objekat trenutno energetskog razreda F i da ne zadovoljava energetske zahteve za postojeće objekte prema Pravilniku o energetskoj efikasnosti zgrada.

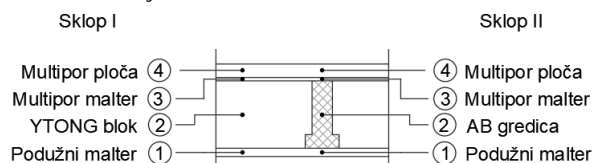


Slika 3. Dijagram potrebne energije u kWh za grejanje po mesecima

6. NADOGRADNJA OBJEKTA

Planirana je nadogradnja stambenog objekta kojom će biti dodat jedan sprat. Na spratu će biti projektovan još jedan stan, dok će potkrovlje ostati neizmenjeno i služitiće kao ostava koja se ne greje. Spratna visina nadograđene etaže je 2,6 m, pa ukupna visina objekta iznad terena iznosi 10,15 m.

Zidovi nove etaže će se graditi YTONG blokovima, noseći zidovi debljine 25 cm, a pregradni zidovi debljine 10 cm. Na mestima sučeljavanja zidova predviđeno je izvođenje vertikalnih serklaža, ovi serklaži će biti povezani armaturom za postojeću međuspratnu konstrukciju. Kao međuspratna konstrukcija biće izvedena YTONG bela tavanice (slika 4), u polumotažnom sistemu. U pitanju je sitnorebrasta AB tavanica, debljine 20cm, sa razmakom rebara 0,685 m.



Slika 4. Međuspratne konstrukcije (bela tavanica)

Izvršena je kontrola napona u nosećim zidovima i temeljima, kao i provera nosivosti greda i stubova, zbog

povećanog opterećenja dodavanjem jedne etaže. Stubovi, grede i zidovi prolaze kontrolu i nije ih potrebno ojačavati, dok je deo temeljne konstrukcije potrebno ojačati, odnosno proširiti, kako bi se povećala kontaktna površina temelja i tla, čime se smanjuje napon u tlu.

7. MERE ZA UNAPREĐENJE ENERGETSKE EFIKASNOSTI

U cilju poboljšanja energetskih svojstava zgrade predviđena je energetska sanacija:

- spoljašnjih zidova zgrade zamenom perlitnog maltera Multipor termoizolacionim pločama;
- unutrašnjeg zida ka negrejanom prostoru postavljanjem plute;
- međuspratne konstrukcije iznad negrejanog prostora postavljanjem Multipor termoizolacionih ploča;
- predviđena je i zamena drvenih vrata adekvatnom PVC stolarijom.

U tabeli 2 dat je pregled koeficijenata prolaza toplote kroz termički omotač objekta pre i posle energetske sanacije.

Tabela 2. Pregled koeficijenata „u“ kroz termički omotač objekta pre i posle energetske sanacije

Ozna-ka	Upr W/m ² K	Upo W/m ² K	Umax W/m ² K	pr	po
Z101	0,406	0,406	0,40	NE	NE
Z101a	0,366	0,273	0,40	DA	DA
Z101b	0,367	0,274	0,40	DA	DA
Z103	1,073	0,337	0,40	NE	DA
Z104	0,404	0,294	0,40	DA	DA
Z201	/	0,281	0,40	/	DA
Z201a	/	0,280	0,40	/	DA
Z01b	1,526	1,526	0,55	NE	NE
Z01c	1,568	1,568	0,55	NE	NE
Z105	2,164	0,825	0,55	NE	NE
PP01	2,442	2,442	0,40	NE	NE
MK01	1,241	0,331	0,40	NE	DA
MK02	1,352	0,338	0,40	NE	DA
MK101	0,493	/	0,40	NE	/
MK201	/	0,391	1,50	/	DA
PR1	1,681	1,681	1,50	NE	NE
PR2	1,697	1,697	1,50	NE	NE
PR3	1,628	1,628	1,50	NE	NE
PR4	1,778	1,778	1,50	NE	NE
PR5	1,642	1,642	1,60	NE	NE
KV1	2,990	1,515	1,60	NE	DA
V2	3,024	3,024	1,60	NE	NE
V3	2,315	1,178	0,40	NE	DA

Legenda:

- Upr - koeficijent prolaza toplote pre sanacije
- Upo - koeficijent prolaza toplote nakon sanacije
- U_{max} - max dozvoljeni koeficijent prolaza toplote
- pr - zadovoljenje uslova pre sanacije
- po - zadovoljenje uslova nakon sanacije
- Z - spoljašnji ili unutrašnji zid
- MK - međuspratna tavanica
- PR - prozori, V - vrata, KV - klizna vrata

7.1. Zaključak

Vrednosti godišnje potrošnje finalne energije za grejanje nakon energetske sanacije i nadogradnje objekta drastično su smanjene. U tabeli 3 prikazani su rezultati proračuna pre i nakon energetske sanacije i nadogradnje objekta.

Tabela 3. Pregled potrebne energije za grejanje objekta pre i posle sanacije

Sanacija	GP kWh/a	SGPE kWh/m ² a	RGPE %	Razred
pre	24 387	167	223	F
posle	21 097	70	94	C

Nakon nadogradnje objekta i primene navedenih sanacionih mera u svrhu unapređenja energetske efikasnosti objekta, on će u potpunosti zadovoljiti energetske zahteve za postojeće zgrade prema *Pravilniku o energetskoj efikasnosti zgrada*, "Sl.glasnik RS", br.061/2011, i preći iz energetskog razreda F u razred C.

8. LITERATURA

- [1] Inženjerska komora Srbije: Pravilnik o energetskoj efikasnosti zgrada, "Sl.glasnik RS", br. 61/2011, Beograd
- [2] Inženjerska komora Srbije: Predavanja za obuku o energetskoj efikasnosti zgrada, Beograd, 2012.
- [3] Malešev M., Radonjanin V.: Trajnost i procena stanja betonskih konstrukcija, Skripta sa predavanja, Fakultet tehničkih nauka, Novi Sad
- [4] Radonjanin V., Malešev M.: Sanacija betonskih konstrukcija, Skripta sa predavanja, Fakultet tehničkih nauka, Novi Sad
- [5] Grupa autora: Beton i armirani beton prema BAB 87, tom 1 - Priručnik, tom 2 - Prilozi, Univerzitetska štampa, Beograd, 1995.
- [9] www.casopis-gradjevinar.hr
- [10] www.ytong.rs

Kratka biografija:



Nataša Rudakijević rođena je u Novom Sadu 1989. godine. Master rad na Fakultetu tehničkih nauka iz oblasti Građevinarstvo – Konstrukcije odbranila je 2018. god.

PROJEKAT ARMIRANOBETONSKE VIŠESPRATNE STAMBENO-POSLOVNE ZGRADE PREMA EVROKODU I POREĐENJE EVROPSKIH I DOMAĆIH PROPISA

THE PROJECT OF MULTI-STOREY REINFORCED CONCRETE BUILDING ACCORDING TO EUROCODE AND COMPARISON WITH NATIONAL STANDARDS

Ivana Gvero, *Fakultet tehničkih nauka, Novi Sad*

Oblast – GRAĐEVINARSTVO

Kratak sadržaj – U prvom delu rada prikazan je projekat armiranobetonske višespratne zgrade S+Pr+5, a u drugom delu je prikazano poređenje evropskih i domaćih propisa.

Ključne reči: Zbornik FTN, Studentski radovi, Uputstvo Armiranobetonska zgrada, evropski standardi, poređenje propisa

Abstract – The first part of the work consist the project of multi-storey reinforced concrete building, basement + ground floor + 5 stories, and the second part consist of comparison of European and Serbian regulations.

Keywords: Proceedings of the FTS, Student papers, Instructions, reinforced concrete building, European standards, comparison regulations

1. UVOD

Projektnim zadatkom predviđeno je projektovanje armiranobetonske višespratne stambeno-poslovne zgrade spratnosti S+Pr+5 prema Evrokodu. Konstrukcijski sistem zgrade je ukrućeni AB skelet. Objekat se nalazi na lokaciji Beograd.

2. OPIS PROJEKTA

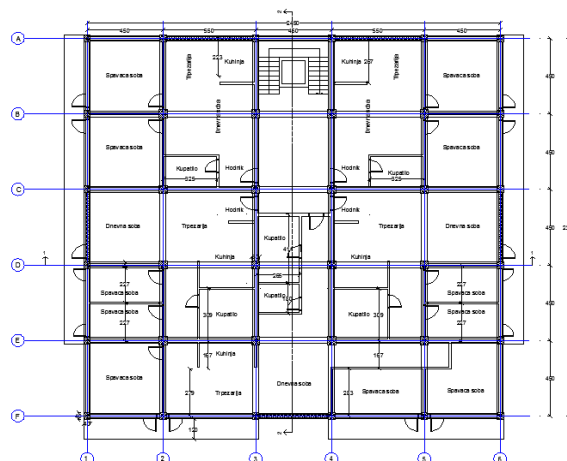
2.1. Projektni zadatak i arhitektonsko rešenje

Prema projektnom zadatku projektovan je stambeno poslovni objekat. Objekat je spratnosti suteran + prizemlje + 5 spratova. Dimenzije objekta u osnovi su 24,5m x 22,5m. Ukupna visina objekta iznad kote terena iznosi 19,5m. Suterenska etaža ima funkciju parkinga za laka putnička vozila. U prizemlju se nalazi 3 lokala. a na spratovima po 5 stambenih jedinica različitih površina. Vertikalna komunikacija između spratova ostvaruje se pomoću dvokrakog stepeništa i lifta, osim na poslednjem spratu, gde je za izlaz na krov kroz otvor u krovnoj ploči ostvaruje pomoću merdevina. Dimenzije stepenika (širina i visina) su određene tako da što približnije odgovaraju optimalnim dimenzijama. Širina stepenišnog kraka iznosi 120 cm. Obrada podova se razlikuje u zavisnosti od namene prostorija.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Đorđe Ladinović, red.prof.

Podovi u kupatilima, kuhinji hodnicima, kao i podovi u lokalima se oblažu keramičkim pločicama. Završna obrada podova u sobama je parket. Fasadni zidovi su izrađeni od pune opeke debljine 25 cm i demit fasade. Sa unutrašnje strane su malterisani krečnim malterom, gletovani i obojeni disperzivnim bojama. Unutrašnji zidovi su takođe od pune opeke, debljine 25 i 12 cm, malterisani krečnim malterom, gletovani i obojeni disperzivnim bojama. Krovna konstrukcija se izvodi kao ravan krov. Da bi se obezbedilo efikasno odvođenje vode urađen je sloj za pad u nagibu.



Slika 1- Osnova tipskog sprata

2.2. Konstruktivni sistem zgrade

Konstruktivni sistem objekta je ukrućeni skeletni. Dimenzije svih greda su 35/50 cm. Dimenzije fasadnih stubova su 40/40 cm, dimenzije unutrašnjih stubova su 50/50 cm.

Međuspratna konstrukcija je projektovana kao sistem kontinualnih krstasto armiranih ploča debljine 15 cm. Debljina stepenišne ploče je 15 cm.

Zidovi za ukrućenje su debljine 20 cm. Suterenska etaža je oivičena zidovima po celom obimu, osim na delu gde se nalazi ulaz u garažu.

Objekat je fundiran na temeljnoj ploči debljine 40 cm koja je ojačana temeljnim gredama dimenzija 120/60 cm. Ispod temeljne ploče nalazi se sloj mršavog betona debljine 10 cm i sloj šljunka debljine 20 cm.

Klasa betona svih nosećih elemenata konstrukcije je C30/37, a kvalitet armature B500C.

2.3. Analiza opterećenja

Za projektovanje i dimenzionisanje elemenata konstrukcije razmatrana su sva opterećenja koja mogu delovati na konstrukciju u toku eksploatacije, kao i njihove najnepovoljnije kombinacije. Analizom su obuhvaćena sledeća opterećenja:

- stalno opterećenje
- korisno opterećenje
- opterećenje vetrom
- opterećenje snegom
- seizmičko opterećenje

Stalno opterećenje obuhvata težinu svih nosećih (grede, stubovi, ploče, AB zidovi) i nenosećih elemenata konstrukcije, kao i težinu podova, obloga. Korisno opterećenje se usvaja prema standardu Evrokod 1 EN 1991-1 1:2002. Intenzitet korisnog opterećenja zavisi od namene prostorije. Opterećenje snegom se računa prema evropskom standardu EN 1991-1-3:2003[1] za krovove nagiba između 0° i 30° i aplicira se na konstrukciju u vidu jednakopodeljenog površinskog opterećenja. Opterećenje vetrom je sračunato prema standardu Evrokod EN 1991-1-4:2005. Na konstrukciju je naneto kao linijsko opterećenje. Seizmičko opterećenje se izračunava pomoću softvera Tower 6.0 multimodalnom spektralnom analizom. Ulazni podaci koji se koriste prilikom proračuna su:

- Objekat se nalazi na tlu C kategorije
- Odnos a/a_g jednak je 0,2
- Koeficijent prigušenja jednak je 0,05

2.4. Modeliranje konstrukcije

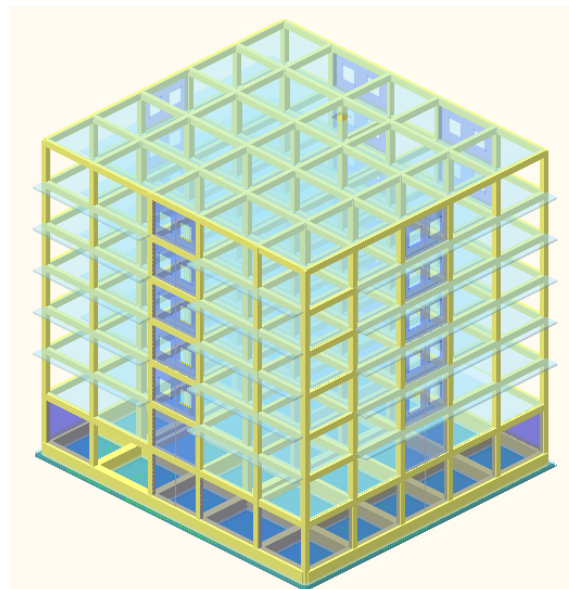
Modeliranje konstrukcije, statički i dinamički proračun urađeni su u softveru Tower 6.0. Osnovni noseći sistem konstrukcije je skeletni sistem sa zidovima za ukrućenje. Model se sastoji iz površinskih (ploče, zidovi) i linijskih (grede, stubovi) konačnih elemenata. Prilikom modeliranja uzet je u obzir pad torziona i savojne krutosti elemenata konstrukcije, i to tako što je torziona krutost greda smanjena na veličinu koja odgovara 10% krutosti homogenog betonskog preseka, dok je savojna krutost smanjena na 50%, savojna krutost stubova i zidova za ukrućenje redukovana je na vrednost koja odgovara 50% krutosti homogenog betonskog preseka.

Opterećenja se nanose na model kao linijska i površinska. Prilikom formiranja proračunskog modela korišćena je mreža konačnih elemenata veličine 0,50m. Tlo je modelirano pomoću Vinklerovog modela, pri čemu je usvojena vrednost koeficijenta posteljice od 20000 kN/m^3 .

Pomoću softvera izvršena je modalna analiza i na osnovu toga su dobijene svojstvene vrednosti i oblici oscilovanja konstrukcije, koji služe za proračun seizmičkih sila. Analiza dejstva horizontalnih opterećenja, kao i modalna analiza, pretpostavlja nedeformabilnost tavanične konstrukcije u svojoj ravni.

Seizmički proračun urađen je multimodalnom spektralnom analizom. Proračun konstrukcije izvršen je prema linearnoj teoriji elastičnosti. Primenom linearne teorije elastičnosti pretpostavlja se geometrijska i materijalna nelinearnost.

Na slici 2 prikazan je 3D model konstrukcije.



Slika 2- 3D model konstrukcije

2.5. Dimenzionisanje i armiranje elemenata

Dimenzionisanje elemenata konstrukcije je izvršeno u softverskom paketu Tower 6.0. Izvršeno je dimenzionisanje temeljne ploče, ploče prizemlja i ploče tipskog sprata, kao i ramova u osi E i osi 6. Svi elementi su dimenzionisani u skladu sa pravilnikom Evrokod za merodavne granične kombinacije opterećenja.

Sve ploče su armirane glavnom armaturom u oba pravca. Grede su dimenzionisane kao jednostruo armirane, dok su stubovi dimenzionisani kao koso savijani, i armirani su simetrično.

Svi konstrukcijski elementi su od betona klase C30/37, dok je kvalitet armature B500C.

2.5. Proračunske kontrole

Izvršene su sledeće proračunske kontrole:

- Kontrola normalizovane vrednosti aksijalne sile u stubovima
- Kontrola normalizovane vrednosti aksijalne sile u zidovima
- Kontrola napona u tlu
- Kontrola relativnog spratnog pomeranja
- Kontrola pomerljivosti konstrukcije

Nakon analize zaključeno je da konstrukcija zadovoljava sve proračunske kontrole.

3. POREĐENJE PROPISA

U sklopu istraživačkog dela rada, izvršena je uporedna analiza trenutno važećih propisa u našoj zemlji i evropskih propisa.

3.1. Svojstva materijala

Čelik za armiranje prema PBAB87 ima oznake koje pokazuju vrstu armature, njenu granicu razvlačenja i karakterističnu čvrstoću. Prema evropskim standardima, koji su usvojeni i kod nas od 2009.godine, čelik se obeležava kao B500A, B500B i B500C. Takođe, minimalna granica razvlačenja je 500 MPa.

Prema evropskim propisima za betone se ne koristi termin "marka", kao u domaćim, već "klasa".

Klase betona označene su sa C xx/yy". Sa xx je označena klasa koja se odnosi na betonski cilindar, a sa yy klasa koja se odnosi na betonsku kocku (na primer C20/25).

3.2. Dejstva i parcijalni koeficijenti sigurnosti

Klasifikacija dejstava je skoro ista prema domaćim i prema evropskim standardima.

Koeficijenti sigurnosti su prema Evrokodu manji, a kombinacije realnije. To rezultira manjim statičkim uticajima.

3.3. Smicanje i torzija

Prema BAB87 u zavisnosti od intenziteta smičuće sile i rezultujućeg smičućeg napona, mogu da se jave slučajevi kada nije potrebna dodatna armatura, kada se deo transverzalne sile poverava betonu, a deo armaturi i kada se celokupna smičuća sila poverava armaturi. U Evrokodu, ukoliko je potrebna armatura za prijem uticaja, celokupna smičuća sila se poverava armaturi.

3.4. Proračun

3.4.1 Analiza opterećenja

Oba standarda predviđaju 5 istih grupa opterećenja: stalno, korisno, sneg, vetar i seizmičko.

Stalno opterećenje prema oba standarda je identično, obuhvata konstruktivne elemente, obloge i nekonstrukтивne elemente na objektu. Vetar nikad nije merodavno za dimenzionisanje betonskih konstrukcija.

Opterećenje snegom, za predmetni objekat koji je rađen prema odredbama iz Evrokoda, uzeto je kao 1 kN/m^2 i aplicirano po krovnoj ravni.

Prema domaćim propisima takođe bi bilo naneto kao 1 kN/m^2 iako je prema oba standarda malo više nego propisano.

Predmet analize biće usvajanje korisnog i proračun seizmičkog opterećenja.

Korisno opterećenje se usvaja po odredbama svakog pravilnika posebno, u zavisnosti od predviđene namene prostorije. U predloženim vrednostima nema puno razlike, osim što se prema Evrokodu korisno opterećenje deli u više kategorija i podkategorija i nudi umanjenja nekih predloženih vrednosti zbog male verovatnoće istovremenog delovanja svih kategorija opterećenja u punom iznosu.

Kod seizmičke analize glavna razlika je u metodi za proračun. Prema domaćim propisima predviđa se korišćenje statički ekvivalentne metode, koja uzima u obzir samo prvi ton oscilovanja.

Prema evropskim propisima predviđa se upotreba multimodalne spektralne analize, koja uzima u obzir više tonova oscilovanja.

Zbog toga, evropski propisi su pogodniji kada se radi o objektima veće spratnosti.

Pre seizmičkog proračuna neophodno je odraditi modalnu analizu da bi se odredile svojstvene vrednosti konstrukcije.

Mase na model od apliciranih grupa opterećenja se grupišu u nivoima tavanica. Razlikuju se faktori učešća masa prema domaćim i evropskim propisima.

3.4.2. Modeliranje konstrukcije

Evrokod, za razliku od domaćih propisa, propisuje smanjenje savojne i torzone krutosti kod grednih nosača, kao i savojne krutosti kod stubova i seizmičkih zidova, kako bi se uzela u obzir isprskalost poprečnog preseka usled apliciranih opterećenja. Prema Pravilniku BAB87 predviđa se samo smanjenje torzione krutosti kod greda.

3.4.3. Proračunske kontrole

Oba pravilnika zahtevaju proračunske kontrole pre početka dimenzionisanja. Kontrole su slične, odnose se na kontrolisanje napona i pomeranja, a razlikuju se prema načinu sprovođenja.

Kontrola napona u stubovima i seizmičkim zidovima prema domaćim propisima se radi tako što se prosečni normalni napon usled eksploatacionih gravitacionih opterećenja poredi sa dozvoljenim naponom. Prema evropskim propisima računa se vrednost aksijalne sile u stubovima za seizmičku proračunsku situaciju. Primenjuju se parcijalni koeficijenti za opterećenje i za materijal. Dobijena vrednost se poredi sa dozvoljenom, koja zavisi od odabrane klase duktilnosti.

Kontrola napona u tlu prema oba propisa se radi tako što se poredi dobijeni naponi sa dozvoljenim naponima u tlu, koji se predstavljaju preko Vinklerovog modela. Naponi u tlu se računaju korišćenjem eksploatacionih opterećenja.

Kontrola pomeranja prema domaćim propisima se radi samo za vrh konstrukcije, dok se prema Evrokodu 8 zahteva kontrola relativnih spratnih pomeranja svih etaža. Kontrola pomeranja radi se za eksploatacionu seizmičku kombinaciju opterećenja.

3.4.4 Dimenzionisanje elemenata

Određivanje debljine zaštitnog sloja betona do armature se razlikuje. Pravilnik BAB87 propisuje debljinu ovog sloja na osnovu vrsta elementa i stepena agresivnosti sredine.

U Evrokodu debljina zaštitnog sloja zavisi od prečnika usvojene armature i od agresivnosti sredina, koja je definisana preko klasa izloženosti.

Domaći propisi za kontrolu na probijanje predviđaju korišćenje normalne sile u stubu od eksploatacionog opterećenja, bez uticaja momenata. Smičuća površ se uzima kružna u osnovi. Položaj kritičnog preseka se kontroliše na rastojanju 0,5hs. Evropski propisi za kontrolu na probijanje koriste granične kombinacije i zbog toga se dobija veća normalna sila. Kontrolni obim je na rastojanju 2hs od krajnje ivice stuba. Oblik krive kritičnog obima se bira u skladu s oblikom opterećene površi (preseka stuba).

3.4.5. Oblikovanje detalja

Oba pravilnika imaju za ideju da obezbede objektu dovoljnu seizmičku otpornost, u smislu pojave oštećenja na očekivanim mestima, koja se lako saniraju, a da ne dođe do ozbiljnih oštećenja ili rušenja konstrukcije. To se naziva koncept programiranog ponašanja i predstavlja niz konstruktivnih mera koje je neophodno preduzeti.

Evrokod se bavi detaljnijom analizom ovog problema. Propisuje se niz kontrola koje moraju da se zadovolje kako bi se objekat smatrao seizmički otpornim. Odredbe

za oblikovanje detalja se najviše odnose na primarne seizmičke elemente - grede, stubove i seizmičke zidove.

4. ZAKLJUČAK

Poređenjem propisa zaključeno je sledeće:

Koiršćeni materijali - oznake i nazivi betona i čelika su drugačije i povećana je granica razvlačenja čelika na 500Mpa. Grupe opterećenja na koje se konstrukcije analiziraju su iste. Evrokod nudi više kategorija korisnog opterećenja.

Metode proračuna seizmičkog proračuna su drugačije i metoda koju propisuju evropski propisi daje tačnije rezultate, pogotovo kada se radi o objektima većih spratnosti.

Proračunske kontrole su slične, razlikuje se način sprovođenja. Kontrola pomeranja u Evrokodu je sigurnija, jer sprečava pojavu tzv. mekog sprata.

Duktilnost, prema SRPS metodama, faktor duktilnosti se usvaja kao uniformna vrednost za sve armiranobetonske zgrade, dok prema Evrokodu koeficijenti duktilnosti zavise od više različitih parametara. Prema Evrokodu, konstrukcije se klasifikuju prema klasama duktilnosti: niska, srednja i visoka.

Oblikovanje detalja za lokalnu duktilnost prema Evrokodu je dosta komplikovanije, ali se detaljno definišu oblikovanje i armiranje primarnih seizmičkih elementata. Zbog toga se sa sigurnošću može tvrditi da će se plastični zglobovi, pri zemljotresu formirati na odgovarajućim mestima - krajevima greda.

Domaći standardi u pomenutim segmentima manje su komplikovani od evropskih. Evropski standard dosta više pažnje posvećuje duktilnosti elemenata i oblikovanju detalja za postizanje iste. Domaći pravilnik je sigurno dovoljno dobar, jer zgrade izgrađene pre nekoliko decenija još uvek postoje i u dobrom su stanju.

5. LITERATURA

- [1] Evrokod 0: Osnove proračuna konstrukcija, Beograd, februar 2006.
- [2] Evrokod 1: Dejstva na konstrukcije, Beograd, novembar 2009.
- [3] Evrokod 2: Proračun betonskih konstrukcija, Beograd, februar 2006.
- [4] Evrokod 8: Proračun seizmički otpornih konstrukcija, Beograd, novembar 2009.
- [5] Grupa autora, Beton i armirani beton prema BAB 87, knjiga 1, Univerzitetska štampa, Beograd 2000.
- [6] Grupa autora, Beton i armirani beton prema BAB87, knjiga 2, Univerzitetska štampa, Beograd 2000.
- [7] Pravilnik o tehničkim normativima za izgradnju objekata visokogradnje u seizmički aktivnim područjima, Službeni list SFRJ br. 31/81
- [8] Dr. Zoran Brujić: "Materijal za predavanja iz predmeta Betonske konstrukcije", Novi Sad 2015.
- [9] Dr. Zoran Brujić: "Betonske konstrukcije u zgradarstvu (prema Evrokodu)", Novi Sad 2018.
- [10] www.radimpex.rs - Uputstvo za Tower 6
- [11] www.radimpex.rs - Uputstvo za ArmCAD 2005

Kratka biografija:



Ivana Gvero rođena je u Beogradu 1992. godine. Master rad na Fakultetu tehničkih nauka, iz oblasti Građevinarstva – Seizmička analiza konstrukcija, odbranila je 2018. god.

PROJEKTOVANJE ČVORA GREDA – STUB U ARMIRANOBETONSKIM KONSTRUKCIJAMA USLED SEIZMIČKOG DEJSTVA

DESIGN OF REINFORCED CONCRETE BEAM – COLUMN JOINT UNDER SEISMIC ACTION

Sanja Lazić, Fakultet tehničkih nauka, Novi Sad

Oblast – GRAĐEVINARSTVO

Kratak sadržaj – U ovom radu je analizirano ponašanje čvora greda-stub u armiranobetonskim skeletnim konstrukcijama pri delovanju seizmičkog opterećenja i data su rešenja za njegovo projektovanje prema evropskom pravilniku Evrokod 8.

Abstract – In this paper, the behaviour of reinforced concrete beam-column joint under seismic action is analysed and given methods for their designing according Eurocode 8.

Ključne reči: čvor greda-stub, smičuća sila, sila prijanjanja betona i armature, smičuća armatura

1. UVOD

Kao jedna od kritičnih tačaka u skeletnom konstrukcijama projektovanim na seizmičko dejstvo su čvorovi greda-stub. Usled seizmičkih momenata savijanja suprotnog znaka, direktno sa gornje i donje strane čvora kao i u gredama sa obe strane čvora, javljaju se velike smičuće sile u oba pravca koje mogu izazvati slom čvora. Na osnovu principa programiranog ponašanja poželjno je dozvoliti pojavu plastičnih zglobova na krajevima grede, ali ne u stubovima i u jezgri samog čvora. Ideja je održati kontinuitet stubova, kao kičme konstrukcije dozvoljavajući pojavu plastičnih zglobova samo u uklještenju sa temeljnom konstrukcijom s ciljem formiranja mehanizma. Osnovni kriterijumi o kojima se mora voditi računa pri projektovanju čvora stub-greda za dejstvo seizmičkog opterećenja su:

- Čvor mora imati veću nosivost od maksimalne nosivosti okolnih greda u kojima se očekuje pojava plastičnog zgloba, kako bi se izbegla disipacija energije u čvoru što može značajno dovesti do pada krutosti usled cikličnog opterećenja;
- Čvor je poželjno posmatrati kao deo stuba;
- Usled manjeg i srednjeg seizmičkog dejstva čvor mora da ima elastičan odgovor;
- Deformacija čvora ne sme značajno da utiče na povećanje relativnog spratnog pomeranja.

Zadovoljenje ovih kriterijuma zahteva poznavanje ponašanja, pravno dimenzionisanje i rešavanje detalja samog čvora.

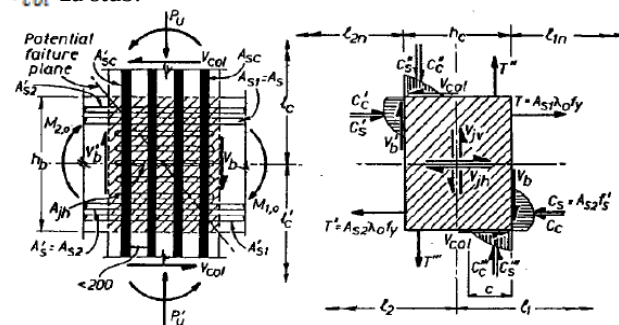
2. KARAKTERISTIKE PONAŠANJA ČVORA GREDA - STUB

Pri delovanju seizmičkog opterećenja velike seizmičke sile se unose u spoj grede i stuba i mogu dovesti do sloma čvora usled sloma sistema za prijem smicanja ili popuštanja prijanjanja između čelika i betona. Najčešće prihvaćen mehanizam prenosa smičućih sila u čvoru delom čini pritisnuta dijagonala u betonu koja potiče od betona koji vrši pritisak na betonski čvor i sila koje se sa armature prenose na betonsko jezgro. Drugi deo ovih sila nosi armatura u čvoru, horizontalna i vertikalna.

Kada je smičuće opterećenje u čvoru malo, pritisnuta dijagonala je u mogućnosti sama da primi opterećenje i armatura nije angažovana, a njena uloga je samo utezanje preseka. U slučaju velikih opterećenja, kada se pojave velike prsline u betonu, prenos zatezanja betonom više nije moguć, pa se zadatak prijema smicanja predaje armaturi u čvoru. Na osnovu ovoga se može zaključiti da je ključan podatak za dimenzionisanje čvora upravo nosivost pritisnute dijagonale u betonu, a povećanje njene nosivosti postiže se povećanjem efektivne površine čvora.

2.1. Smičuće sile u čvoru

Kako bi se analizirale unutrašnje sile koje se javljaju u čvoru u toku seizmičkog opterećenja neophodno je izdvojiti čvor iz konstrukcije (Slika 1(levo)) [2], a uticaje od susjednih elemenata i armature zameniti ekvivalentnim silama (Slika 1(desno)) [2]. Sile zatezanja koje prenosi samo armatura, jer se smatra da je već došlo do pojave plastičnih zglobova u gredama, su označene sa T , dok su sile pritiska betona i armature označeni sa C_c i C_s , respektivno. Smičuće sile su označene sa V_b za gredu i V_{col} za stub.



Slika 1. – Spoljašnji uticaji i unutrašnje sile u čvoru
Posmatrajući sliku 1(desno) iz sume horizontalnih sila za čvor dobija se vrednost za horizontalnu silu smicanja u čvoru

$$V_{jh} = T + T' - V_{col} \quad (1)$$

odnosno, izraz za određivanje sile za čvor u objektu

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Đorđe Ladinović, red. prof.

$$V_{jh} = \gamma_{Rd} (A_{S1} + A_{S2}) f_y - V_{col} \quad (2)$$

i za čvor na ivici objekta

$$V_{jh} = \gamma_{Rd} A_{S1} f_y - V_{col} \quad (3)$$

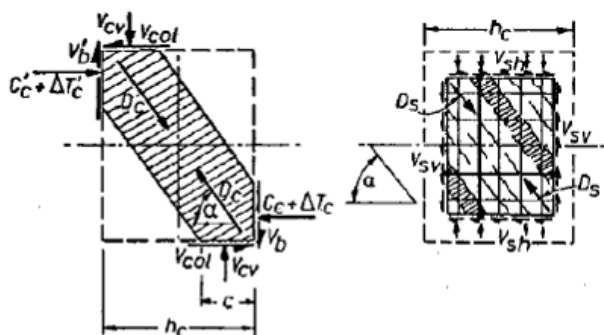
gde je

A_{S1} , A_{S2} - površina armature gornje i donje zone grede;
 γ_{Rd} - faktor kojim se uvodi povećanje momentnih nosivosti usled očvršćavanja čelika i ima vrednost 1,0 za DCM i 1,2 za DCH [1].

Kada je u pitanju vertikalna komponenta smičuće sile, izvođenje njene vrednosti je mnogo komplikovanije zbog podužne armature stuba koja prolazi kroz čvor, pa se zadovoljavajuće vrednosti dobijaju preko izraza

$$V_{jv} = \frac{h_b}{h_c} V_{jh} \quad (4)$$

uz pretpostavku da su vrednosti smičuće vertikalne i horizontalne sile međusobno zavisne.



Slika 2. – Dva mehanizma prijema smičućih sila

Pri delovanju seizmičkog opterećenja kao posledica suprotnih momenata savijanja na krajevima grede sa obe strane čvora, kao i u stubovima sa gornje i donje strane čvora, javljaju se pritisnute zone betona koje u čvoru greda-stub izazivaju formiranje pritisnute dijagonale D_c (slika2 (levo)) [2]. Početni ugao pod kojim se dijagonala formira približno iznosi $\alpha \approx \tan^{-1}(h_b/h_c)$, pa njena vertikalna i horizontalna komponenta imaju vrednost

$$\begin{aligned} V_{ch} &= D_c \cos \alpha \\ V_{cv} &= D_c \sin \alpha \end{aligned} \quad (5)$$

Upravno na pravac pružanja pritisnute dijagonale javiće se naponi zatezanja u betonu. Kada su opterećenja mala, betonsko jezgro će moći samo da nosi napon zatezanja koji se u njemu javlja. Sa povećanjem opterećenja, u gredama dolazi do pojave plastičnih zglobova, a prekoračuje se i napon zatezanja u čvoru.

Sile sa glavne armature grede i stuba koje prolaze kroz čvor grade u betonu pritisnutu dijagonalu D_s (slika 2(desno)) [2]. U ovom stadijumu beton više nije u mogućnosti da prihvati napone zatezanja koji se javljaju upravno na formiranu dijagonalu, pa tu ulogu preuzima armatura.

Ako se pretpostavi da se i dijagonala D_s u betonu formira pod istim uglom α kao i dijagonala D_c njena horizontalna komponenta će imati vrednost

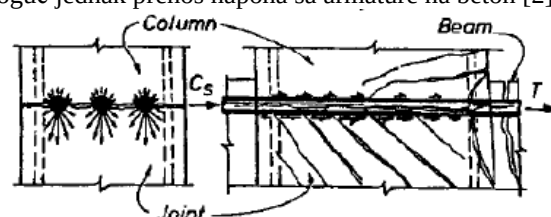
$$V_{sh} = D_s \cos \alpha = V_{jh} - V_{ch} \quad (6)$$

Horizontalna komponenta V_{sh} se osigurava horizontalnom armaturom, dok se prihvatanje vertikalne komponente V_{sv} prepušta glavnoj armaturi stuba, osim kod jednoosno armiranih stubova kada je potrebno projektovati dodatnu armaturu u zoni čvora.

2.2. Nosivost veze betona i armature u čvoru

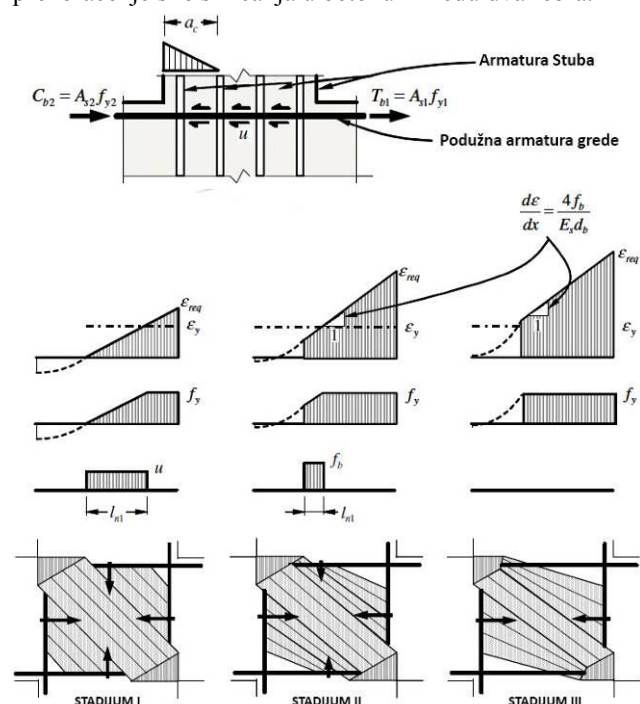
Nosivost veze između betona i armature ima veliki značaj u konstrukcijama opterećenim na seizmičko opterećenje, posebno u oblastima u kojima se javljaju uslovi naizmenničkog deformiranja i neelastičnog ponašanja konstrukcije. U skeletnim konstrukcijama ova mesta su spojevi grede i stubova u kojima se može očekivati proklizavanje glavne armature grede zbog plastičnih zglobova koji se formiraju na njihovim krajevima. Slabljenje ove veze ne mora uvek značiti nagli gubitak nosivosti, ali može značajno uticati na krutosti rama.

Slabljenje spoja između glavne armature grede koja prolazi kroz čvor i betona u čvoru počinje kada armatura dostigne svoju granicu razvlačenja, a to se dešava uvek u zoni plastičnog zgloba kao posledica nejednakih deformacija armature i okolnog betona. Proklizavanje zategnute armature u čvoru ima nepovoljan uticaj zbog položaja same šipke na ivici elementa zbog čega nije moguć jednak prenos napona sa armature na beton [2].



Slika 3. – Distribucija napona sa armature na beton

Do narušavanja spoja usidrene šipke i betona dolazi usled drobljenja betona koji se nalazi ispred rebra armature i prekoračenje sile smicanja u betonu između dva rebra.



Slika 4. – Promena vrednosti napona i dilatacija u šipki glavne armature grede unutar spoja po pojavi plastičnog zgloba u gredi

Na slici 4 [4] opisana je pojava gubitka nosivosti spoja između armature i betona. U stadijumu I (slika 4) došlo je do pojave zgloba u gredi, dilatacija u armaturi gornje zone u oblasti plastičnog zgloba ϵ_{req} je tek prekoračila

graničnu vrednost ε_y , a napon u armaturi u ovoj zoni nema priraštaj (dostignuta je granica razvlačenja f_y). Dužina sidrenja je redukovana na dužinu l_{n1} koja predstavlja oblast u kojoj nije narušena veza između armature i betona, odnosno, oblast između dela šipke u kojoj je došlo do prekoračenja napona razvlačenja u armaturi i oblasti u kojoj je armatura izložena pritisku. U ovom stadijumu vrednosti smičućih sila u vertikalnom i horizontalnom pravcu je približno isti. U stadijumu II (slika 4) povećava se dilatacija u armaturi kao i dužina u kojoj je armatura dostigla granični napon. Ovo znači da je zona sidrenja armature smanjena, a usled toga i napon na kontaktnoj površini povećan. U stadijumu III cela dužina armature šipke koja je izložena zatezanju je prekoračila graničnu dilataciju i plastično se deformiše, a veza između armature i betona je potpuno narušena i dolazi do povećanja horizontalnih pomeranja u čvoru.

3. DIMENZIONISANJE ČVORA GREDA - STUB

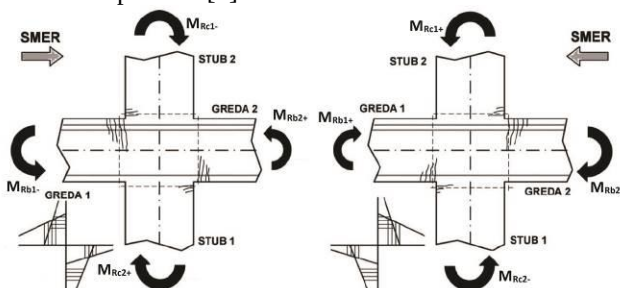
3.1. Savojna nosivost stuba

Stubovi kao elementi koji su dominantno opterećeni aksijalno, u slučaju seizmičkog opterećenja izloženi su delovanju momenata savijanja i velikih smičućih sila, što je nepovoljno u pogledu ostvarivanja duktilnog ponašanja pri pojavi plastičnog zgloba u stubu. Pri projektovanju cilj je kod stubova zadržati elastično ponašanje, odnosno izbeći pojavu plastičnih zglobova, a njihovu pojavu omogućiti na krajevima greda i samim tim grede projektovati na duktilno ponašanje i disipaciju energije plastičnim deformisanjem.

Kontrola savojne nosivosti stubova, u Evrokodu 8, uzima se tako da zbir proračunskih vrednosti momenata nosivosti stubova vezanih u čvoru mora biti za 30% veći od zbira proračunskih vrednosti momenata nosivosti na krajevima greda u čvoru, odnosno

$$\sum M_{Rd,c} \geq 1,3 \cdot \sum M_{Rd,b} \quad (7)$$

S obzirom na to da je moment nosivost stuba u funkciji aksijalnog opterećenja, za stub se mora koristiti najnepovoljnija vrednost aksijalne sile u opsegu sila koje odgovaraju seizmičkim proračunskim situacijama, odnosno, sila za koju se dobija najmanja vrednost momenta nosivosti stuba. Uslov se mora proveriti za dva horizontalna pravca u konstrukciji, kao i za dva alternativna smera u svakom od pravaca [1].



Slika 5. Momenti u čvoru za jedan smer seizmičkog dejstva

Posmatrani uslov kontroliše nosivost stubova na osnovu njihovih geometrijskih karakteristika i usvojene armature, a ne na osnovu statičkih uticaja od određenih kombinacija opterećenja, što u ovom slučaju znači da će za bilo koju

vrednost seizmičkog opterećenja doći do pojave plastičnog zloba u gredi, a ne u stubu.

Prema EC8 postoje određeni slučajevi kada se zahtev savojne nosivosti stubova može i drugačije posmatrati:

1. U ravnim okvirima sa najmanje četiri stuba približno istih dimenzija poprečnih preseka se dozvoljava da jedan od četiri stuba u posmatranom horizontalnom pravcu ne mora da zadovolji dati uslov.
2. Uslov se ne mora proveravati za spoj stuba poslednjeg sprata i krovne grede gdje pojava plastičnog zgloba neće značajno uticati na globalno stanje konstrukcije.
3. Kod dvospratnih zgrada je neophodno normalizovani napon u stubovima prizemlja $v_d = N_d / (A_c f_{cd})$ ograničiti na maksimalno 0,3 za seizmičke kombinacije opterećenja, zbog povećanog zahteva smičuće nosivosti u ovim stubovima.

Uzimajući u obzir da je zadovoljenje uslova lokalne duktilnosti mnogo zahtevnije od graničnih uslova pri dimezionisanju stubova za slučaj seizmičkog opterećenja, preporučuje se dimezionisanje stubova na osnovu ovog uslova [1].

3.2. Rešavanje detalja armature

Kao armatura za prijem smicanja u čvoru projektuju se uzengije i spiralna armatura koja ima zadatak prijema horizontalnih sila smicanja dok se kao vertikalna armatura uzimaju unutrašnje šipke glavne armature stuba za koje se očekuje da će biti izložene pritisku, tako da će zadržati potrebnu nosivost za prijem vertikalnih sila smicanja.

Kod konstrukcija koje se projektuju za srednju klasu duktilnosti zahteva se da horizontalna armatura za utezanje preseka ne sme biti manja od one u susjednim kritičnim oblastima stubova. U slučaju kada se u čvoru sučeljavaju grede sa sve četiri strane čija je širina namanje 75% paralelne dimenzije stuba dozvoljava se povećanje razmaka između uzengija ali ne na više od 150mm.

Za konstrukcije koje se projektuju za visoku klasu duktilnosti ograničava se vrednost smičuće sile u čvoru iz uslova da napon pritiska u čvoru ne prekorači vrednost nosivosti betona na pritisak uz prisustvo poprečnih dilatacija zatezanja

- a) za unurašnji čvor greda-stub

$$V_{jhd} \leq \eta \cdot f_{cd} \sqrt{1 - \frac{v_d}{\eta}} \cdot b_j \cdot h_{jc} \quad (8)$$

- b) za spoljašnji čvor greda-stub

$$V_{jhd} \leq 0,8 \cdot \eta \cdot f_{cd} \sqrt{1 - \frac{v_d}{\eta}} \cdot b_j \cdot h_{jc} \quad (9)$$

$$\eta = 0,6 \cdot \left(1 - \frac{f_{ck}}{250 \text{ MPa}} \right) \quad (10)$$

gde je:

h_{jc} - rastojanje između krajnjih slojeva armature stuba;

V_{jhd} - smičuća sila u čvoru;

b_j - efektivna širina čvora i definiše se kao:

$$b_j = \begin{cases} \min\{b_c; (b_w + 0,5h_c)\} & \text{za } b_c \geq b_w \\ \min\{b_w; (b_c + 0,5h_c)\} & \text{za } b_c < b_w \end{cases} \quad (11)$$

Količina horizontalne armature koja je potrebna u čvoru za utezanje i prijem sila zatezanja određuje se preko izraza

a) za unurašnji čvor greda-stub

$$A_{jh} f_{yhd} \geq \gamma_{Rd} (A_{S1} + A_{S2}) f_{yd} (1 - 0,8v_d) \quad (12)$$

b) za spoljašnji čvor greda-stub

$$A_{jh} f_{yhd} \geq \gamma_{Rd} A_{S2} f_{yd} (1 - 0,8v_d) \quad (13)$$

Ovako određena količina horizontalnih uzengija postavlja se ravnomerno raspoređena po visini čvora, gde se kod spoljašnjih čvorova preporučuje obuhvatanje povijenih šipki glavne armature uzengijama.

Vertikalna armatura čvora je obezbeđena u vidu srednjih armaturnih šipki glavne armature stuba ili kao dodatne šipke armature umetnute u poprečni presek stuba. Količina ove armature mora da zadovolji uslov

$$A_{sv,i} \geq \frac{2}{3} A_{sh} \left(\frac{h_{jc}}{h_{jw}} \right) \quad (14)$$

gde je $A_{sv,i}$ ukupna površina srednjih šipki armature na odgovarajućim stranama stuba.

Da bi se izbegao lom usled gubitka prijanjanja data su i geometrijska ograničenja kojima se definiše maksimalni prečnik armature koja prolazi kroz čvor i to:

a) za unurašnji čvor greda-stub

$$\frac{d_{bL}}{h_c} \leq \frac{7,5 f_{ctm}}{\gamma_{Rd} f_{yd}} \cdot \frac{1 + 0,8v_d}{1 + 0,75k_D \rho' / \rho_{\max}} \quad (15)$$

b) za spoljašnji čvor greda-stub

$$\frac{d_{bL}}{h_c} \leq \frac{7,5 f_{ctm}}{\gamma_{Rd} f_{yd}} \cdot (1 + 0,8v_d) \quad (16)$$

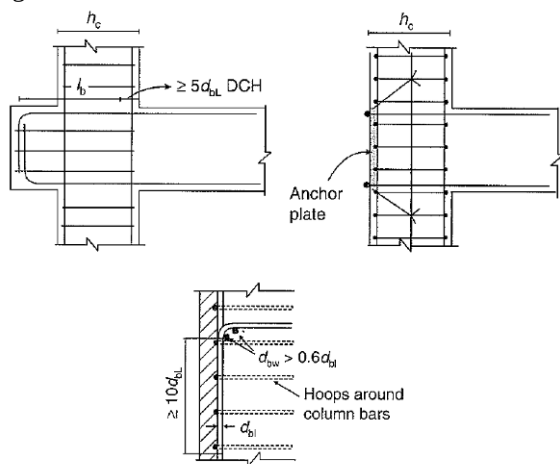
gde je:

h_c - širina stuba paralelna armaturnim šipkama grede;

k_D - faktor klase duktilnosti (2/3 za DCM i 1,0 za DCH);

γ_{Rd} - faktor nepouzdanosti (1,0 za DCM i 1,2 za DCH);

$\rho_{\max}$ - maksimalni dozvoljeni koeficijent armiranja greda zategnutom armaturom.



Slika 6. – Osiguranje dužine sidrenja u spoljašnjem čvoru

Ukoliko zbog male širine stuba uslov nije moguće ispuniti u spoljašnjim čvorovima potrebno je obezbediti dužinu sidrenja izvođenjem ispusta ili sidrenjem armature preko ankerne ploče ili poprečne armature (slika 6) [1]. Pri

ankerisanju armature potrebno je da dužina šipke nakon povijanja bude veća ili jednaka od $10d_{bL}$.

4. ZAKLJUČAK

Obezbeđenje duktilnog ponašanja skeletnih konstrukcija primarno zavisi od rešenja detalja čvora stub-greda. Pri delovanju seizmičkog opterećenja velike seizmičke sile se unose u spoj grede i stuba i mogu dovesti do sloma čvora usled sloma sistema za prijem smicanja ili popuštanja prijanjanja između čelika i betona. Unutrašnje sile prenesene sa susednih elemenata na čvor rezultuju smičućim silama u horizontalnom i vertikalnom pravcu. Ove smičuće sile vode pojavi dijagonalnih napona pritiska i zatezanja u čvoru. Najčešće prihvaćen mehanizam prenosa smičućih sila u čvoru delom čini pritisnuta dijagonala u betonu koja potiče od betona koji vrši pritisak na betonski čvor i sila koje se sa armature prenose na betonsko jezgro. Drugi deo ovih sila nosi armatura u čvoru, horizontalna i vertikalna.

Do slabljenje nosivosti čvora može doći i usled loma pri gubitku prijanjanja između betona i glavne armature grede. Ovo se dešava kada armatura dostigne svoju granicu razvlačenja usled nejednakih deformacija armature i okolnog betona što kao posledicu ima pad nosivosti spoja i povećanje horizontalnih pomeranja.

Značajnu ulogu u očuvanju nosivosti čvora ima horizontalna armatura u vidu uzengija koja pored prijema sila zatezanja ima ulogu utezanja betona u jezgru čvora što obezbeđuje bolju nosivost betona na pritisak.

5. LITERATURA

[1] Fardis, M. N., Carvalho, E., Elnashai, A., Faccioli, E., Pinto, P., Plumier, A.: *Designers' Guide to EN 1998-1 and EN 1998-5, Eurocode 8: Design of Structures for Earthquake Resistance, General Rules, Seismic Actions, Design Rules for Buildings, Foundations and Retaining Structures*; Thomas Telford, London 2005

[2] Paulay, T., Priestley, M.J.N.: *Seismic Design of Reinforced Concrete and Masonry Buildings*; John Wiley & Sons, Inc., 1992

[3] Uma, S.R., Jain, S.K.: *Seismic design of beam-column joints in RC moment resisting frames – Review of codes*; Structural Engineering and Mechanics, Vol. 23, No. 5 (2006) 579-597

[4] Hong, S., Lee, S.: *Strut and Tie models for deformation of reinforced concrete beam-column joints dependent on plastic hinge behavior of beams*; World Conference on Earthquake Engineering, Vancouver, B.C., Canada, Paper No. 1026 (2004)

Kratka biografija:



Sanja Lazić je rođena u Doboju 1987. godine. Master rad iz oblasti građevinarstvo - konstrukcije odbranila je u julu 2018. godine na Fakultetu tehničkih nauka.

PROCENA STANJA I SANACIJA KONSTRUKCIJE MEHANIZOVANOG PODNOG SKLADIŠTA ZA ŽITARICE**ASSESSMENT AND REPAIR OF STRUCTURE OF MECHANIZED FLAT GRAIN STORAGE WAREHOUSE**Bojan Milivojević, *Fakultet tehničkih nauka, Novi Sad***Oblast – GRAĐEVINARSTVO**

Kratak sadržaj – Rad se sastoji iz teorijskog i stručnog dela. U teorijskom delu su prikazani objekti za skladištenje materijala i specifičnosti projektovanja silosa. U stručnom delu prikazani su procena stanja, statički proračun i predlog mera sanacije podnog skladišta za žito.

Ključne reči: Silosi, podno skladište, procena stanja, statički proračun, sanacija

Abstract – The paper consists of theoretical and practical part. The theoretical part shows structures for storing materials and characteristics of structural design of silos. The practical part shows assessment of current condition, structural design and the suggestions for repair of the mechanized flat grain storage warehouse.

Keywords: Silos, flat grain storage, assessment of current condition, structural analysis design, repair

1. UVOD

Rad se sastoji iz dva dela, teorijskog i stručnog. U prvom, teorijskom, delu analizirani su različiti objekti za skladištenje materijala, kao i specifičnosti projektovanja silosa. Prvo je izvršena klasifikacija različitih objekata za skladištenje koji se javljaju u praksi, dok su potom prikazane osnovne preporuke za projektovanje silosa uz prikaz odredbi iz trenutno važećeg Standarda SRPS ISO 11697. Nakon toga prikazana su različita dejstva koja mogu biti od interesa za sam proces projektovanja jednog silosnog postrojenja, kao i česti uzroci za rušenje silosa. Stručni deo rada obuhvata utvrđivanje postojećeg stanja konstrukcije mehanizovanog podnog skladišta za žitarice u Perlezu, sprovođenje kontrolnog statičkog proračuna prema odredbama Pravilnika BAB 87 i SRPS, kao i predlog mera za izvođenje kvalitetne sanacije.

2. OBJEKTI ZA SKLADIŠTENJE MATERIJALA I SPECIFIČNOSTI PROJEKTOVANJA SILOSA

Ove vrste objekata mogu biti izvedene kao armiranobetonski, čelični ili drveni (danas retko), a osnovni podatak za pravilan odabir, kako sa funkcionalno-tehnološkog tako i sa ekonomskog aspekta, leži u tome koji materijal će biti skladišten u razmatranoj vrsti objekta. Osnovni tipovi objekata za skladištenje su:

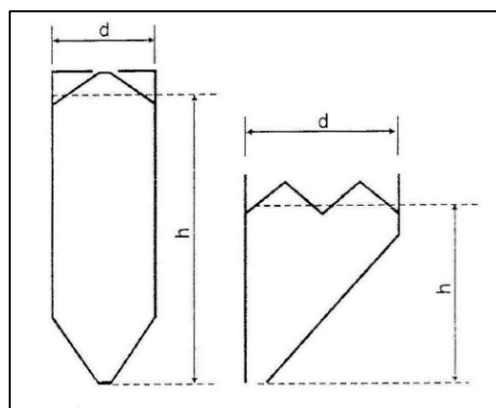
- Otvoreni objekti - deponije;
- Zatvoreni objekti - podna skladišta, bunker i silosi.

Otvoreni objekti nazivaju se još i deponije rasute robe i razlikuju se deponije na tlu i one koje su ukopane. Najčešće se ovakav vid skladištenja koristi za čuvanje šljunka, peska, uglja, šećerne repe, rude, kamena, otpadnih materija, itd.

Zatvoreni objekti predstavljaju građevine različitih dimenzija i geometrijskih oblika koje se po tipovima dele na nadzemne i poluukopane i na osnovu upotrebljenog materijala od kojeg su izrađene na armiranobetonske i čelične. Same karakteristike zavise od materijala koji se skladišti i primenjene tehnologije realizacije pretovarnih procesa.

Podna skladišta su objekti (hale) kod kojih se odlaganje rastresitog materijala vrši direktno na pod (podnu ploču). Obično imaju oblik koji prati konture materijala koji se smešta unutar njih.

Objekti za skladištenje nasutih (rastresitih) materijala kod kojih je visina „h“ u odnosu na svetlu širinu „d“ mala nazivaju se bunkerima, a u suprotnom reč je o silosima (Slika 1.). Odnos koji predstavlja granicu između silosa i bunkera je $h/d=1.5$ ($h/d<1.5$ - bunker; $h/d>1.5$ - silosi).



Slika 1. Razlika u odnosu dimenzija između silosa (levo) i bunkera (desno)

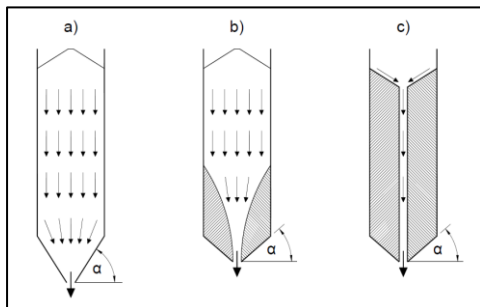
Bunker i silosi, najčešće se grade za potrebe skladištenja u blizini odgovarajućih poljoprivrednih ili industrijskih pogona (industrijski silosi) i u područjima gde se materije proizvode, ali su takođe zastupljeni i na mestima kao što su luke i železničke stanice gde se koriste tzv. pretovarni silosi. Materijali koji se koriste za izradu bunkera i silosa su čelik armirani beton i prednapregnuti beton.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vlastimir Radonjanin, red. prof.

2.1. Najbitniji parametri za projektovanje silosa

Najbitniji parametri za projektovanje silosa su zapreminska masa, ugao unutrašnjeg trenja materijala, i karakteristični način pražnjenja ćelija (Slika 2.)

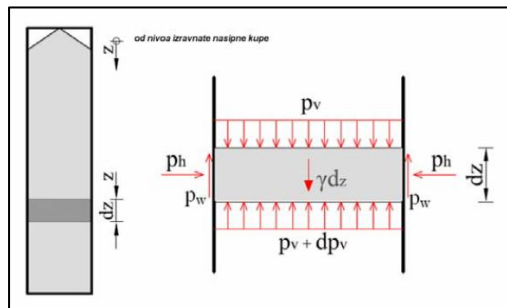


Slika 2. Osnovni tipovi isticanja materijala

2.2. Dejstva na AB silose

Dejstva koja su razmatrana su korisno opterećenje-pritisaci u ćelijama od uskladištenog materijala, temperaturna dejstva, opterećenja usled eksplozija i požara, diferencijalna sleganja i seizmička dejstva.

Opterećenje koje u najvećoj meri opredeljuje dimenzionisanje konstruktivnih elemenata je korisno opterećenje od uskladištenog materijala. Proračun pritisaka na zidove ćelija usled dejstva materijala koji se skladišti unutar silosa se proračunava u skladu sa teorijom Janssen-Koene-a na način da se uslovi ravnoteže posmatraju na diferencijalnom sloju materijala na dubini z (Slika 3.).



Slika 3. Diferencijalni sloj materijala i pripadajuća opterećenja

2.3. Uzroci rušenja silosa

Rušenje silosa nije redak slučaj i predstavlja polje intenzivnog istraživanja. U skladu sa tim, izdvaja se nekoliko različitih uzroka zbog kojih dolazi do rušenja silosa. Mogu se podeliti na sledeće:

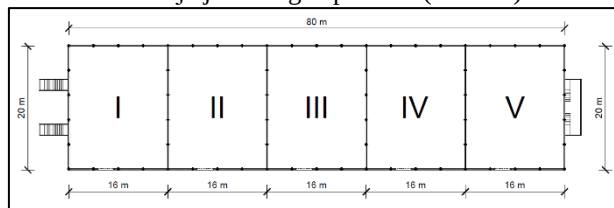
- Greške pri projektovanju;
- Greške pri izvođenju;
- Neadekvatna ekspanzija;
- Neadekvatno održavanje ili
- Kombinacija nekih od gore navedenih uzroka.

Ovo znači da često za greške nije odgovoran pojedinac već da odgovornost snosi određena grupa ljudi. Potencijalno odgovorni za ovakve situacije mogu biti projektanti, izvođači, dobavljači (snabdevači) materijala, investitori, korisnici i ostali.

3. PROCENA STANJA PODNOG SKLADIŠTA

3.1. Tehnički opis konstrukcije

Projektnim zadatkom predviđena je procena stanja, sprovođenje kontrolnog proračuna i predlog sanacionih mera za mehanizovano podno skladište za smeštaj žita u mestu Perlez. Objekat je zamišljen kao petoćelijsko skladište za žito u zrnju (Slika 4.) u čijoj neposrednoj blizini se nalaze sušara i silosi za smeštaj žitarica. Ceo sistem je međusobno povezan sistemom transportera (jedan sa gornje strane objekta i dva koja se nalaze ispod samog objekta u odgovarajućim kanalima). Ukupan projektovani kapacitet skladišta je cca 600 vagona žita u zrnju, a kapacitet pojedine ćelije je cca 120 vagona. U osnovi objekat je osovinskih gabarita 20 x 80 m, prizemnog tipa. Ukupan gabarit po dužini je izdvojen poprečnim zidovima na svakih 16 m i na taj način je formirano 5 ćelija jednakog kapaciteta (Slika 4.).



Slika 4. Osnova prizemlja, oznake ćelija i osovinski gabariti

Osnovna koncepcija konstrukcije je čelični skelet sa ispunom od montažnih armiranobetonskih zidova. Krovni nosač je rešetkasta konstrukcija od valjanih „L“ i „U“ profila sa krovnom spregom u ravni gornjeg pojasa. Stubovi su sistema proste grede na koje deluje sila potiska od rinfuze koju na stubove prenose armiranobetonski montažni zidni nosači i normalna sila od krovnog opterećenja. Horizontalne sile stub predaje spregovima u ravni donjeg pojasa rešetke i zategama u podu. Rezultujuću silu horizontalni spregovi u ravni donjeg pojasa rešetke, preko vertikalnih spregova, predaju zidnim platnima.

3.2. Procena stanja

Predmet detaljne procene stanja su zidovi, podne ploče skladišta, stubovi, rešetkasta krovna konstrukcija i pripadajući spregovi (u ravni gornjeg i donjeg pojasa), čelično stepenište i zidani zidovi stepeništa kanala.

Defekti i oštećenja koja se javljaju na armiranobetonskim elementima su: nedovoljna debljina zaštitnog sloja, šupljikava površina, betonska gnezda i zone segregacije, korozija betona, plastična deformacija, mrlje na površini, prslina i pukotine (Slika 5.)

Na čeličnim elementima nisu uočeni defekti, dok se od oštećenja javljaju plastična deformacija stubova i korozija (Slika 6.).

Na zidanim zidovima koji su pregledani nisu uočeni defekti, već samo oštećenja i to najviše od tragova biološke korozije i prslina.



Slika 5. Prsline kose orijentacije na zidovima



Slika 6. Korozija stuba

Od nedestruktivnih metoda ispitivanja primenjena je metoda merenja površinske tvrdoće betona. Metodom Sklerometra je ispitana površinska tvrdoća betona i na osnovu dobijenih vrednosti i statističke obrade rezultata pokazno je da su na pojedinim elementima dobijene manje vrednosti čvrstoće betona od projektom predviđenih.

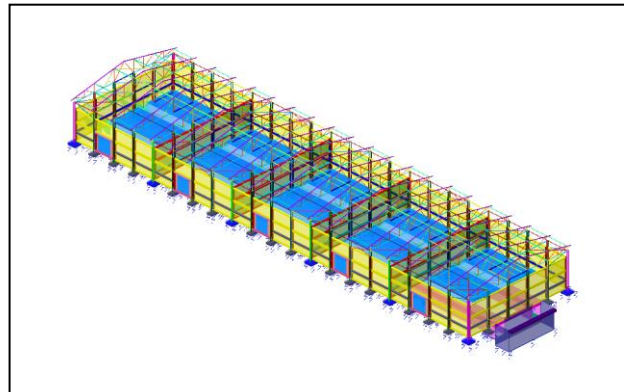
4. KONTROLNI PRORAČUN KONSTRUKCIJE

Kontrolni (statički) proračun konstrukcije je obavezan segment svakog ozbiljnijeg projekta sanacije i njime se predmetna (postojeća) konstrukcija podvrgava proračunu prema trenutno važećim Pravilnicima, Standardima i uputstvima za projektovanje. Svrha ovakvog proračuna je dokaz da objekat (svi njegovi konstruktivni elementi), za koji se vrši procena stanja i (eventualne) konstrukcijske sanacije, može da prima i prenosi sva opterećenja u skladu sa odredbama važećih Pravilnika i da ispunjava sve zahtevane kriterijume. Pre samog proračuna izvršeni su modeliranje konstrukcije i analiza opterećenja.

Kontrolni proračun koji je sproveden izvršen je u skladu sa odredbama Pravilnika BAB 87 (za armiranobetonske elemente) i SRPS (za čelične elemente). Po završenom proračunu sprovedene su tzv. proračunske kontrole, a

podeljene su u dve grupe - kontrola napona u tlu i kontrole koje se direktno odnose na konstruktivne elemente.

Modeliranje konstrukcije i adekvatan statički i dinamički proračun izvršen je u softverskom paketu Tower 7, koji služi za statičku i dinamičku analizu konstrukcija, a koji je zasnovan na metodi konačnih elemenata. Izgled modela je prikazan na Slici 7.



Slika 7. Izgled proračunskog modela konstrukcije

Nakon kontrola upoređeni su rezultati dobijeni pri originalnom statičkom proračunu i dimenzionisanju sa rezultatima novosprovedenog (kontrolnog) proračuna i na osnovu ovog upoređivanja doneti su zaključci o potrebnim ojačanjima i sanacionim radovima konstrukcijske prirode. Takođe u okviru ovog poglavlja je izvršeno usvajanje potrebnih količina armature za ojačanje površinskih elemenata u vidu zidova, kao i proračun i usvajanje karbonskih traka potrebnih za ojačanje linijskih elemenata.

5. PREDLOG MERA SANACIJE

Uzimajući u obzir uočene defekte i oštećenja pregledanih elemenata i rezultate kontrolnog proračuna, u ovom poglavlju su date mere kojima se predviđa saniranje. Klasifikacija elemenata je izvršena po materijalnoj pripadnosti, dajući pre toga predlog radova koji bi trebalo da prethode samim sanacionim, a takođe i završne radove koji bi trebalo da budu izvedeni po završetku konstrukcijske i nekonstrukcijske sanacije.

U radove koji prethode saniranju se ubrajaju uklanjanje vegetacije oko objekta, koja bi mogla smetati u procesu izvođenja sanacionih radova; uklanjanje dotrajale opreme (elevatori i transporter) i dotrajalih instalacija iz objekta; demontaža i obezbeđenje adekvatnog postupka uklanjanja postojećeg krovnog pokrivača u vidu azbest-cementnih ploča i uklanjanje drvenih rožnjača (sa krovni nosača objekta i sa krovni nosača u delu nad stepenišnim prostorom na ulasku u kanale); uklanjanje maltera sa spoljašnjih površina zidova i sa spojeva zidova i stubova sa unutrašnje strane zidni nosača; uklanjanje trotoara oko objekta.

U sanacione radove konstrukcijske prirode spadaju: uklanjanje i zamena postojećih rožnjača; ojačavanje zidova stepeništa i zidova kanala za smeštaj transportera;

ojačavanje zidova i greda u sklopu jame za smeštaj elevatora; uklanjanje postojećih i izvođenje novih podnih ploča; uklanjanje, izvođenje i montaža novih stubova i zidova na mestu postojećih koji su pretrpeli plastičnu deformaciju; ojačavanje zidova čelija i ostvarivanje novih veza između zidova i stubova; ponovno ostvarivanje veze između krovnih nosača (sa pripadajućim spregovima) i stubova; uklanjanje i zamena postojećih štapova sprega u krovnoj ravni, koji nisu zadovoljili proračunske kriterijume; rušenje zidanih zidova stepeništa kanala i uklanjanje nosača (ankerovanih u zidove) i zidanje novih zidova i postavljanje novih nosača.

U sanacione radove nekonstrukcijske prirode spadaju: postavljanje nove krovne obloge; uklanjanje tragova korozije sa čeličnih elemenata i nanošenje odgovarajućih zaštitnih premaza; uklanjanje betona u zonama nepovezanih slojeva (betonskih gnezda) i zona segregacije na zidnim nosačima (na spoljašnjoj i na unutrašnjoj strani elemenata) i reparacija ovih mesta; injektiranje prslina i pukotina u okviru zidnih nosača; reparacija oštećenih betonskih površina zidova; uklanjanje bioloških tragova i mrlja sa površina zidova i repariranje tih zona.

U završne radove su svrstani postavljanje nove obloge na fasade, zamenu postojećih prozora i vrata novim, izvođenje novog trotoara oko objekta, malterisanje unutrašnjih i spoljašnjih površina zidova i zamenu postojećeg sistema za odvođenje atmosferske vode sa objekta novim.

Na kraju su date faze po kojima je predviđeno da se svi prethodno pobrojani radovi izvedu.

6. ZAKLJUČAK

Procena stanja podnog skladišta urađena je analiziranjem dostupne projektno-tehničke dokumentacije, detaljnim vizuelnim pregledom i primenom nedestruktivnih metoda i jasno ukazuje na nekoliko problema, koji se ogledaju u nedovoljnoj pažnji posvećenoj trajnosti konstrukcije prvenstveno u pogledu održavanja i sprovođenja redovnih kontrolnih pregleda. Klasifikacija uočenih defekata i oštećenja je sprovedena prema materijalnoj pripadnosti pojedinih elemenata iz čega proističe zaključak da je svim elementima konstrukcije u manjoj ili većoj meri potreban određen vid konstrukcijske ili nekonstrukcijske sanacije. Potpuna slika o svim merama koje su potrebne da bi se sprovela adekvatna sanacija konstrukcije je formirana nakon sprovođenja statičkog proračun, prema trenutno važećim odredbama.

Nakon sprovedenog proračuna utvrđeno je da su većini čeličnih elemenat potrebni radovi nekonstrukcijske sanacije, dok su većini armiranobetonskih elemenata potrebni radovi konstrukcijske sanacije. Za saniranje svakog od elemenata konstrukcije predviđene su odgovarajuće mere za saniranje uz detaljne opise procedure njihovog sprovođenja i upotrebe odgovarajućih reparaturnih materijala i zaštitnih premaza.

7. LITERATURA

- [1] M. Malešev, V. Radonjanin: Trajnost i procena stanja betonskih konstrukcija - Materijal sa predavanja, Fakultet tehničkih nauka, Novi Sad
- [2] M. Malešev, V. Radonjanin: Sanacija betonskih konstrukcija - Materijal sa predavanja, Fakultet tehničkih nauka, Novi Sad
- [3] Grupa autora: Beton i armirani beton prema BAB 87, tom 1 - Priručnik, tom 2 - Prilozi, Univerzitetska štampa, Beograd, 1995.
- [4] Z. Brujić: Betonske konstrukcije inženjerskih objekata - Materijal sa predavanja
- [5] V. Hasanović - Silosi, Sarajevo, 2001

Kratka biografija:



Bojan Milivojević rođen je u Beogradu 1993. god. Osnovne akademske studije upisuje 2011. godine, a diplomski rad iz oblasti Betonske konstrukcije brani 2016. godine. Master studije smer-konstrukcije upisuje 2017. godine i Master rad na Fakultetu tehničkih nauka iz oblasti Građevinarstvo-Procena stanja i sanacija konstrukcija odbranio je 2018.god.

SISTEMI ZAŠTITE PLATNIH KARTICA**SYSTEMS OF CASH CARD PROTECTION**Nataša Čačić, *Fakultet tehničkih nauka, Novi Sad***Oblast – POŠTANSKI SAOBRAĆAJ I
TELEKOMUNIKACIJE**

Kratak sadržaj – Sve učestalije dinamične potrebe ljudi i razvoj tehnologije, u velikoj meri utiče na razvoj bankarskog poslovanja. Dostupnost različitih, društveno prihvatljivih i cenovno konkurentnih bankarskih usluga uz upotrebu tehnologija, postaje temelj današnjeg bankarstva i društva uopšte. U cilju zaštite korisnika bankarskih usluga od rizika zloupotrebe podataka putem elektronskog bankarstva, kroz ovaj rad bice definisani sistemi zaštite privatnosti podataka prilikom korišćenja ATM (Automated Teller Machine) uređaja kroz upotrebu biometrije kao modela identifikacije.

Ključne reči: Platne kartice, Biometrija, Zloupotreba, Elektronsko poslovanje

Abstract – The increasingly frequent dynamic needs of people and the development of technology have greatly influence the development of banking operations. The availability of various, socially acceptable and cost-competitive banking services with the use of technologies is becoming the basis of today's banking and society in general. In order to protect users of banking services from the risk of data abuse via electronic banking, this work will define data protection systems when using ATM (Automated Teller Machine) devices through the biometrics as an identification model.

Keywords: Cash card, Biometry, Misuse, E-banking

1. UVOD

Razvoj informaciono-komunikacionih tehnologija kao i globalizacija tržišta doveli su do pojave elektronskog poslovanja kao lakšeg načina obavljanja brojnih poslova u odnosu na klasične načine. Elektronsko bankarstvo odlikuje se svojom brzinom, jednostavnošću i pouzdanošću, čime se postiže maksimalna ugodnost prilikom rada što je izuzetno bitno za korisnike. Primena kartica kao savremenog bezgotovinskog oblika plaćanja i jednostavnost pri korišćenju, doveo je do drastičnog rasta velikog broja korisnika a geografska pokrivenost platnim karticama povećava se iz godine u godinu. Iako su se kartice sa magnetnim trakama pokazale kao isplative i praktične, iskustva su pokazala da su moguće pretnje sigurnosti pri korišćenju platnih kartica sa magnetnom trakom značajno izražene.

Profil osoba koje vrše zloupotrebe takođe se izmenio na način da ovaj vid kriminala danas uglavnom podrazumeva mlađe i tehnologijama obučene osobe. Danas se zloupotrebe izvode na daljinu u odnosu na mesto i predmet napada, koriste se sofisticirani uređaji poput bežičnih mikrokamera, lažnih tastatura za unos PINa i drugi. Veći obim bezgotovinskog plaćanja ostvario bi se, ako bi potrošači bili sigurniji u elektronička plaćanja. Opravdan razlog uvođenja čip tehnologije je prvenstveno povećan rast zloupotrebe kartica sa magnetnom trakom širom sveta. Napadači imaju jasan uvid u detalje ovih kartica što im omogućava da identifikuje slabe tačke obezbeđenja od prevare. ATM su bankarski automati koji obavljaju rutinske operacije za korisnike. To su aparati koji sadrže mikroprocesore i koji imaju online vezu sa bazom podataka u banci. U suštini se radi o novom tipu elektronskih filijala savremenih banaka koje mogu da obave niz rutinskih operacija sa individualnim klijentima. Biometrija igra značajnu ulogu u obezbeđivanju sigurnosti, i predstavlja budućnost sigurne provere identiteta. Sve više uređaja koje svakodnevno koristimo su povezani sa biometrijom (prepoznavanje glasa na mobilnim uređajima, prepoznavanje lica na pametnim telefonima i sl.). Kao posledica toga, klijenti će biti sve zahtevniji kada je u pitanju bezbednost njihovih bankovnih računa.

**2. BEZGOTOVINSKO PLAĆANJE – PLATNE
KARTICE**

Plaćanje bez upotrebe gotovog novca nazivamo bezgotovinskim plaćanjem. Ono se vrši preknjižavanjem potrebnog iznosa plaćanja sa računa dužnika na račun poverioca kod iste kreditne ustanove i njene mreže filijala, ili između različitih kreditnih ustanova i to uz pomoć različitih instrumenata. Ovim plaćanjem postižu se velike uštede u poslovima, prostorijama, uređajima, kadrovima itd. jer se smanjuje potreba izrađivanja, čuvanja, prenošenja i čestog prebrojavanja velike količina gotovog novca.

2.1. Platne kartice

Platne kartice su instrument bezgotovinskog plaćanja koji vam omogućava plaćanje robe i usluga i podizanje gotovog novca. Brojni su razlozi za njihovo korišćenje [1]:

- Ušteda vremena i za korisnike i za trgovce,
- Jednostavnost i komfor,
- Sigurnost u slučaju eventualnog gubitka,
- Raspoloživost svih sredstava na računu.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragana Šarac, vanr. prof.

Prilikom plaćanja mogu se koristiti sva novčana sredstva kojima raspolažemo na računu (uključujući i dozvoljen minus u nekim bankama, tj. kreditni limit), a ne samo gotovinu koju imamo u novčaniku. Na teritoriji Srbije postoje sledeće vrste platnih kartica:

Domaće:

- DinaCard
- YUBA

Strane:

- VISA,
- MasterCard/Maestro,
- American Express,
- Diners Club Card.

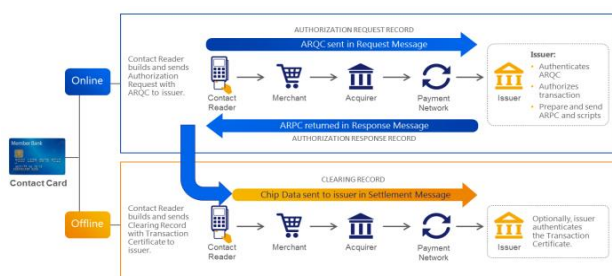
Najčešće korištena sredstva za obavljanje finansijskih transakcija u bezgotovinskom poslovanju su:

- Kreditne/debitne kartice,
- Pametne kartice (Smart Card) i elektronski novčanici (Electronic Wallet),
- Elektronski ili digitalni novac (Electronic Money, Digital Cash),
- Sistemi mikropaćanja (Micropayment).

Sigurnost kod korišćenja plastičnih kartica obezbeđuje se putem personalnog identifikacionog broja (personal identification number – PIN) koji se ugrađuje na magnetnu traku ili čip i sastoji se od nekoliko digitalnih znakova. PIN broj služi da se vlasnik identifikuje prilikom korišćenja kartice na ATM uređajima (bankomatima). Za pojedine vrste platnih kartica kao što je to na primer debitna platna kartica Maestro koja pripada MasterCard platnom sistemu PIN broj se koristi za identifikaciju ne samo prilikom korišćenja kartice na ATM uređajima već i prilikom svakog plaćanja roba/usluga na prodajnim mestima koja su opremljena POS terminalima. ISO Standard 9564 je međunarodni standard za upravljanje ličnim identifikacionim brojem (PIN-om) i za obezbeđenje sigurnosti u finansijskim uslugama. Najznačajnije kartične organizacije zajedničkim naporima su razvile EMV standard koji predstavlja skup specifikacija čijim se sleđenjem omogućava kreiranje okruženja u kojem je moguća nesmetana interakcija između različitih kartica, terminala i drugih sistema. Standard definiše sigurnosni okvir za debitne i kreditne platne aplikacije, a koji počiva na četiri elementa [2]:

- Autentifikacija kartice (Card Authentication Method),
- Kontrola rizika (Risk Management),
- Offline PIN,
- Uzajamna autentifikacija.

Na slici 1., prikazana je autentifikacija procesa plaćanja:



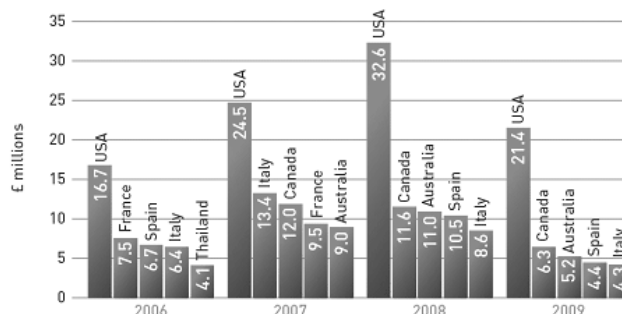
Slika 1. EMV autentifikacija procesa plaćanja [2]

3. PREDNOSTI PRIMENE ČIP KARTICA

Pametne ili čip kartice rade na principu pohrane i obrade određenih informacija koje se koriste tokom izvršavanja transakcije od strane mikropcesora ugrađenog u samu karticu. Memorija na čipu omogućava smeštanje i izvršavanje većeg broja platnih (debit, credit) ili neplatnih aplikacija (loyalty) na samo jednoj kartici. Postoji niz aspekata koji idu u prilog prelasku sa platnog sistema zasnovanog na magnetnoj traci na čip tehnologiju od kojih su najznačajniji [3]:

- Povećan kapacitet skladištenja i sigurnost podataka koji pruža čip u odnosu na magnetnu traku.
- Mogućnost smart kartice da sadrži i razne druge aplikacije kao i da pruži dodatne usluge koje se ne odnose samo na usluge plaćanja
- Podrška offline PIN verifikaciji na mestu servisa umesto online PIN verifikacije ili ručnog potpisa.
- Veoma lako se prilagođava novim tehnološkim rešenjima koja mogu biti jednostavno implementirana na postojećim karticama.

Prednosti EMV su očigledne i uglavnom se odnose na funkcionalnosti koje pruža čip tehnologija. Za razliku od statičke strukture magnetne trake koja omogućava samo zapis i čitanje podataka, čip sadrži mikropcesor sa svojim operativnim sistemom i celim skupom komandi koje može izvršiti. Na grafiku 1., prikazane su zemlje sa najvećim brojem kartičnih zloupotreba koje su napravljene ukradenim, izgubljenim ili krivotvorenim britanskim karticama, do 2009. godine kada dolazi do EMV migracije [3].



Grafik 1. Zemlje sa najvećim brojem zloupotrebljenih britanskih kartica (u milionima eura) [3]

4. SIGURNOSNA REŠENJA U ELEKTRONSKOM BANKARSTVU

Ulaganja u cilju dostizanja bezbednosti e-banking sistema su jednako bitna kao i ulaganja u sam njihov razvoj, jer opstanak elektronskog bankarstva zavisi od ugleda banaka i njihovih sposobnosti da ubede klijente da su njihova sredstva i transakcije sigurni. Kao i kod svih online transakcija, postoje rizici u elektronskom bankarstvu koji uključuju bezbednost transakcija, finansijski gubitak zbog lične ili bankovne greške i gubitak vremena u korišćenju sistema i ispravljanju grešaka. U pogledu sistema elektronskog bankarstva, neki od bezbednosnih zahteva su:

- Kriptovana komunikacija (kako bi komunikacija bila zaštićena od „man-in-the-middle“ napada, neophodno je da ista bude kriptovana),
- Limiti transakcija (limiti mogu biti po transakciji, dnevni, mesečni, itd.),
- Dvofaktorska identifikacija.

Biometrijska identifikacije se često koristi kao oblik zaštite od rizika i zloupotreba u elektronskom bankarstvu. Korišćenje PIN-ova i lozinki u e-banking proverama identiteta je često mnogo jače sredstvo u teorijskom nego u praktičnom smislu, jer korisnici u velikom broju slučajeva imaju problem da zapamte više PIN-ova i lozinki. Shodno tome, dolazi do rasta bezbednosnih rizika jer korisnici biraju jednostavne lozinke koje lako pamte i pri tome ignorišu bezbednosne savete banaka. U ovom slučaju, kao nauka o identifikovanju i prepoznavanju korisnika na osnovu njegovih fizičkih osobina i ponašanja, biometrija je efikasno rešenje. Ove osobine korisnika je veoma teško falsifikovati i kopirati. Njih korisnik ne može da izgubi ili zaboravi. Da bi neka biometrijska karakteristika bila pogodno sredstvo za identifikaciju, mora ispunjavati sledeće uslove:

- Univerzalnost – mora biti prisutna kod svake osobe;
- Jedinstvenost ili individualnost – kod svake osobe karakteristika mora biti različita;
- Trajnost i nepromenljivost – ne sme i ne može se promeniti protekom vremena;
- Jednostavnost prikupljanja i poređenja – najveća prednostu odnosu na druge metode.

Biometrija igra značajnu ulogu u obezbeđivanju sigurnosti i predstavlja budućnost sigurne provere identiteta [4].

5. BUDUĆNOST AUTOMATSKIH SAMOUSLUŽNIH ŠALTERA

Automatski samouslužni šalteri (ATM) su od svoje pojave prošli kroz nekoliko etapa u razvoju, a u svakoj etapi bili su obogaćivani nizom dodatnih funkcija. Napretkom tehnologije ATM uređaji polako postaju nešto više od aparata za isporuku gotovine, povećavajući na taj način svoj profitni potencijal. Oko 11% finansijskih institucija u SAD je putem ATM uređaja dostavljalo poštanske uputnice, poštanske marke, kupone, putničke čekove i sl.

Danas postoje ATM uređaji koji mogu da primaju papirne novčanice i čekove u stokovima do 50 komada, a u stanju su da odštampaju priznanicu sa kopijom deponovanih čekova. Noviji ATM uređaji omogućavaju korisnicima personalizaciju putem mogućnosti programiranja izvesnih karakteristika. ATM može da prati i memoriše navike svakog klijenta i da mu prilikom naredne posete uputi odgovarajuću marketinšku poruku.

Kada bi ATM bio u mogućnosti da vrši interakciju sa različitim mobilnim uređajima, to bi moglo doneti značajne uštede u vremenu.

Klijenti bi mogli putem svog mobilnog uređaja da unesu neophodne podatke i pripreme sve svoje transakcije (deponovanje ili podizanje novca, prenos na druge račune...), nakon čega bi mobilni uređaj signalizirao ATM uređaju da izvrši sve transakcije, čime bi se uštedelo vreme i smanjili redovi ispred ATM uređaja [5].

6. BIOMETRIJSKI SISTEMI ZAŠTITE

Reč biometrija nastala je od dve grčka reči: bios (život) i metria (meriti). Sama ideja biometrije je veoma stara, ali razvoj biometrije kao moderne nauke omogućen je tek pojavom računara i savremene tehnologije. Sama biometrija predstavlja automatizovan metod utvrđivanja identiteta osobe na osnovu fizioloških karakteristika kao što su otisak prsta, geometrija ruke, lice, raspored vena, glas, dužica ili mrežnjača oka. Reč biometrija nastala je od dve grčka reči: bios (život) i metria (meriti). Sama ideja biometrije je veoma stara, ali razvoj biometrije kao moderne nauke omogućen je tek pojavom računara i savremene tehnologije. Sama biometrija predstavlja automatizovan metod utvrđivanja identiteta osobe na osnovu fizioloških karakteristika kao što su otisak prsta, geometrija ruke, lice, raspored vena, glas, dužica ili mrežnjača oka.

Postoje dve vrste biometrijskih sistema:

1. unimodalni i
2. multimodalni.

Biometrijski sistemi koji vrše proveru posmatrajući samo jednu ulaznu osobinu, predstavljaju unimodalne sisteme i zato često dolazi do greške. Zbog provere samo jednog podatka, moguća je velika sličnost proveravane karakteristike kod više osoba, pa zbog toga ovi sistemi nisu pouzdani. Multimodalni sistemi predstavljaju biometrijske sisteme koji se zasnivaju na proveri više ulaznih podataka koji se u određenom trenutku sjedinjavaju. Tako da je kod ovih sistema, minimalna mogućnost greške.

Za proveru identiteta moguća su tri načina:

- nešto što jesi (izgled-biometrija) – ne postoji mogućnost gubljenja ili zaboravljanja identifikatora, a i mogućnost falsifikovanja je znatno smanjena.
- nešto što imaš (dokument, identifikaciona kartica) – dobra strana ovog pristupa je što korisnik ne mora ništa da pamti, ali postoji mogućnost gubljenja dokumenta, kao i falsifikovanja ili krađe.
- nešto što znaš (PIN, šifra, neka lična informacija) – mana ovakvog načina identifikacije je u tome što korisnik često, u strahu da ne zaboravi lozinku, istu zapiše na papir i na taj način ona postaje fizički objekat podložan zloupotrebi [6].

Biometrijski sistemi mogu da koriste:

- Oko za identifikaciju,
- Otisak prsta,
- Raspored crta lica,
- Geometrija dlana i prstiju,
- Termogram lica i tela,
- 3D fotogrametrijska antropologija i 3D facijalna rekonstrukcija,
- Telesni mirisi,
- Analiza glasa,
- Rukopis i potpis,
- Dinamiku tipkanja.

Primenom savremenih tehnoloških rešenja u sistemima za identifikaciju korisnika, povećava se nivo bezbednosti i zaštita od zloupotrebe ličnih podataka. Prepoznavanje skeniranjem mrežnjače je tehnologija koja za prepoznavanje koristi karakteristike krvnih sudova na dnu

oka. To je najpouzdanija biometrijska metoda identifikacije. Do danas nije pronađen način da se mrežnjača falsifikuje. Otisci prsta invarijantni su na starosne promene, te stoga za sada predstavljaju jednu od najiskorištavanijih biometrijskih karakteristika pri dokazivanju identiteta. Tehnologija otisaka prsta toliko je česta u identifikaciji da je postala sinonim za biometriju. Glavni problem s tehnologijom otiska prsta je prihvatljivost običnom korisniku, jer se otisci prsta tradicionalno koriste u kriminalističkim i forenzičkim istragama.

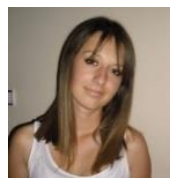
7. ZAKLJUČAK

Na osnovu izvršene analize sistema zaštite platnih kartica od zloupotrebe, zaključuje se da se u budućnosti više pažnje treba posvetiti biometrijskim sistemima identifikacije korisnika, jer se primenom biometrije poboljšava bezbednost korisnika i pruža znatno veća preciznost pri utvrđivanju identiteta korišćenjem ATM uređaja. Potrebno je pripremiti sadašnje, a i buduće generacije za nove informacione tehnologije. Nije dovoljno samo stihijski krenuti u projekte pripreme za korišćenje biometrije, već je potrebno daleko veće angažovanje nego što je to danas.

8. LITERATURA:

- [1] Narodna banka Srbije, *"Vodič za platne kartice"*, 2007.
- [2] MOORE CSI, *"EMV standard"*, 2011.
- [3] B.Topić, Pavković, *"Efekti primjene čip kartica kao savremene tehnologije plaćanja u bankarstvu"*, Operacioni menadžment u funkciji održivog ekonomskog rasta i razvoja Srbije, NLB Razvojna banka a.d. Banja Luka, Bosna i Hercegovina, (2011-2020).
- [4] J.Obradović, M.Dragosavac, S.Arsić, *"Primena savremenih tehnologija u bankarstvu, zloupotreba i moguća rešenja"*, SYNTHESIS Beograd, 2015.
- [5] V.Grbić, S.Nićin, *"Budućnost automatskih samouslužnih šaltera"*, Evropska revija, Evropski univerzitet Brčko distrikta, Brčko, 2015.
- [6] Milena Vladić, *"Identifikacija korisnika pomoću biometrije oka u elektronskom poslovanju"*, Studentski časopis za teoriju i praksu menadžmenta, Univerzitet u Beogradu, Tehnički fakultet u Boru, 2015.

Kratka biografija:



Nataša Čačić rođena je u Šapcu 1993. godine. Osnovne studije na Fakultetu tehničkih nauka iz oblasti Saobraćaja - Poštanski saobraćaj telekomunikacije završila je 2016. godine.

kontakt: natasa_cacic@yahoo.com



Dragana Šarac rođena je u Novom Sadu 1969. godine. Doktorirala je na Fakultetu tehničkih nauka 2009. godine. Oblas interesovanja poštanski saobraćaj komunikacije.

DISTRIBUIRANO REŠAVANJE SISTEMA LINEARNIH JEDNAČINA KORIŠĆENJEM RASPBERRY PI UREĐAJA PREKO AD-HOC MREŽE**DISTRIBUTED SOLUTION OF LINEAR EQUATIONS USING RASPBERRY PI DEVICES VIA AD-HOC NETWORK**

Silard Ištván, Dejan Nemec, *Fakultet tehničkih nauka, Novi Sad*

Oblast – SAOBRAĆAJ

Kratak sadržaj – Zadatak ovog master rada jeste da objasni šta je Raspberry Pi, šta znači pojam Ad-Hoc, da opiše postupak povezivanja tri Raspberry Pi-a u Ad-Hoc mrežu i da pokaže kako se rešavaju proste linearne jednačine na tri Raspberry Pi-a istovremeno razmenom rezultata tih jednačina međusobno. U ovom master radu biće još objašnjeno kako je ovaj uređaj nastao, koji modeli uređaja postoje, koja dodatna oprema postoji, kao i operativni sistem koji je napravljen baš za ovaj uređaj.

Ključne reči: Raspberry Pi, linearne jednačine, Ad-Hoc

Abstract – The subject of this project is to explain what is Raspberry Pi, what does the term Ad-Hoc mean, to describe the process how to connect three Raspberry Pi's via Ad-Hoc network and to show how to solve linear equations using the three Raspberry Pi's simultaneously by exchanging the results of these equations among themselves. In this project, it will be further explained how this device developed, the models of this device, accessories and the operating system that was created specifically for this device.

Keywords: Raspberry Pi, linear equations, Ad-Hoc

1. UVOD

Zadatak ovog master rada je da objasni šta je Raspberry Pi, kako je nastao, koji modeli uređaja postoje, koja dodatna oprema za Raspberry Pi postoji i kako funkcioniše operativni sistem Raspbian.

Pored toga, ovaj rad obuhvata i programske jezike, stavljajući aspekt na programski jezik Python, pošto je kod koji rešava linearne jednačine napisan u navedenom programskom jeziku.

U radu je dato detaljno objašnjenje kako se vrši povezivanje više Raspberry Pi-eva u Ad-Hoc mrežu bežičnim putem i dat je detaljan prikaz napisanih kodova za svaki Raspberry Pi sa grafičkim prikazima rezultata u realnom vremenu. Zaključni deo sadrži oblasti u kojima je Raspberry Pi najčešće korišćen.

2. RASPBERRY PI

Raspberry Pi je najmanji računar na svetu. Ima veličinu kreditne kartice koji se povezuje sa monitorom ili televizorom preko HDMI priključka i sa tastaturom i mišem pomoću USB priključka. Ovaj mali računar može da se koristi u elektronskim projektima, za surfovanje po Internetu, u izradi dokumenata, za programiranje i čak za igranje manjih igrica. Raspberry Pi može da pokrene i video visokog kvaliteta (HD video) [1].

Raspberry Pi je nastao u Velikoj Britaniji 2012. godine kao projekat koji je imao zadatak da olakša učenje osnovnih računarskih nauka u školama. Međutim, vremenom, Raspberry Pi je postao veoma popularan. Entuzijasti ga koriste, a noviji modeli imaju razne upotrebe, čak i u robotici [2]. Raspberry Pi 1 Model B predstavlja prvu generaciju malog i moćnog računara. Predstavljen je u februaru 2012. godine u Velikoj Britaniji. Model B+ je finalna verzija originalnog modela. Cena uređaja se kreće od 25\$ do 35\$.

Specifikacije za Raspberry Pi 1 Model B+ su sledeće:

- Čip: Broadcom BCM2835 SoC
- Arhitektura jezgra: ARM11
- Procesor: 700MHz Low Power ARM1176JZF5 Applications Processor
- Grafička kartica: Dual Core VideoCore IV Multimedia Co-Processor, koji sadrži Open GL ES 2.0, hardverski ubrzan OpenVG i 1080p30 H.264 visoko-profilno dekodiranje
- Memorija: 512MB SDRAM
- Operativni sistem: sistem se podiže preko Micro SD kartice, koristi se Raspbian, ali moguće je i instalirati još neke operativne sisteme
- Dimenzije: 85 x 56 x 17mm
- Napajanje: Micro USB Socket 5V, 2A [3]

Na slici 1 dat je prikaz za Raspberry Pi 1 Model B.



Slika 1. Raspberry Pi 1 Model B+

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dejan Vukobratović, vanr. prof.

3. PROGRAMSKI JEZICI

Programski jezik je bilo koji od veštačkih jezika kojim je moguće dati detaljne instrukcije računaru, kao što su specifikacije za izračunavanje ili algoritam. Te instrukcije se mogu izvršavati direktno kada su ugrađene u računar u posebnom obliku koji je odredio proizvođač, tzv. mašinski jezik, posle jednostavnog procesa zamene izražene u odgovarajućem asemblerskom jeziku, ili posle prevođenja iz nekog jezika višeg nivoa.

Mašinski i asemblerski jezici su jezici niskog nivoa, koji zahtevaju od programera da se posveti upravljanju svim aspektima vezanim za čuvanje podataka i operacije nad njima. Na drugom kraju nalaze se jezici visokog nivoa, koji su bliži prirodnom jeziku i oslobađaju programera brige o tim stvarima, takođe su čitljiviji i daleko lakši za pisanje programa [4].

Generacije programskih jezika jesu:

- Prva generacija – To su jezici niskog nivoa kao što je mašinski jezik.
- Druga generacija – To su takođe jezici niskog nivoa kao što su assembly jezici koji se obično koriste za video igre i video uređivanje (eng. video editing) i ponekad u jezgrima (eng. kernel).
- Treća generacija – To su jezici visokog nivoa, kao što su C, C++, Java, JavaScript, Visual Basic i drugi.
- Četvrta generacija – To su jezici koji se sastoje od izjava sličnih izjavama na ljudskom jeziku. Često se koriste u bazama podataka, i tu spadaju Perl, PHP, Python, Ruby, SQL i drugi.
- Peta generacija – To su programski jezici koji sadrže vizuelne alate za pisanje programa. U ovu generaciju spadaju jezici kao što su Mercury, OPS5, Prolog, i drugi [5].

4. POSTUPAK INSTALIRANJA OPERATIVNOG SISTEMA RASPBIAN

Instaliranje Raspbian-a je vrlo jednostavno i za osobe koje nemaju velika znanja u IT oblastima. Za instaliranje ovog operativnog sistema (OS) potrebni su: računar sa čitačem Micro SD kartica (danas skoro ne postoji laptop koji nema ovakav čitač), Internet konekcija i memorijska kartica od min. 8 GB.

Prvo je neophodno preuzeti Raspbian OS, a to se može uraditi na dva načina direktno preko sajta ili korišćenjem torrent servisa. U slučaju da je Internet konekcija brza i bezbedna (bez prekida) onda može da se izabere direktno preuzimanje, a u suprotnom je poželjno izvršiti preuzimanje pomoću torrent-a.

Preuzimanje Raspbian OS preko zvaničnog sajta je vrlo jednostavno. U web pretraživač unese se termin “Raspbian download” i klikne se na prvi link koji se dobije kao rezultat pretrage. Klikom na taj link, pretraživač će prikazati stranicu gde postoje dva izbora: Noobs i Raspbian. U ovom slučaju, izabere se Noobs, pošto je to lakši način za instaliranje. Nakon toga biće data dva izbora: Noobs i Noobs Lite. Izabere se Noobs.

Posle preuzimanja Noobs.zip fajla, potrebno je još instalirati program “SDFormatter” na računar. Postupak

pronalaženja i preuzimanja ovog programa je isti kao i za Raspbian OS, tj. ukuca se “SDFormatter download” u web pretraživač, klikne se na prvi link i posle prihvatanja ugovora vezano za licencu, klikne se na “OK”. Nakon toga potrebno je instalirati program “SDFormatter”. Formatiranje kartice se vrši u nekoliko kratkih koraka:

- Ubaci se Micro SD kartica u čitač kartica i pogleda se koje slovo pripada kartici (npr. F:).
- Pokrene se program “SDFormatter”.
- Kod Format Option-a klikne se na Option i kod Format Size Adjustment-a izabere se ON i zatim klikne se na OK.
- Kod Drive-a izabere se slovo koja pripada kartici (npr. F:).
- Klikne se na Format.

Posle ovih koraka, memorijska kartica je formatirana, i potrebno je još da se ekstrahuje sadržaj Noobs.zip fajla koji je preuzet sa zvaničnog sajta i da se prekopiraju ti fajlovi na karticu. Potrebno je obratiti pažnju da sve što je sadržano u Noobs.zip fajlu bude prekopirano u “Root” deo kartice. Posle završetka svega navedenog, može se uključiti uređaj.

Kada se uređaj startuje prvi put, postaviće se pitanje koji OS treba da se instalira. Tada treba da se izabere “Raspbian” i zatim se klikne na “Install”. Posle toga treba sačekati da se instalacija završi i uređaj je spreman za rad.

5. AD-HOC

Bežična Ad-Hoc mreža (eng. Wireless Ad-Hoc Network – WANET) ili mobilna Ad-Hoc Mreža (eng. Mobile Ad-Hoc Network – MANET) je decentralizovana vrsta bežične mreže. Mreža je Ad-Hoc jer se ne oslanja na već postojeću infrastrukturu, kao što su ruteri (eng. Router) u žičnim mrežama ili pristupne tačke u upravljanim (infrastrukturnim) bežičnim mrežama. Umesto toga, svaki čvor učestvuje u rutiranju posredstvom podataka za druge čvorove, tako da određuje koji čvorovi prosleđuju podatke dinamičkim putem na osnovu mrežne povezanosti i algoritma rutiranja koji se koristi [6].

Bežične mobilne Ad-Hoc mreže su samo-konfigurisajuće, dinamične mreže u kojima se čvorovi mogu slobodno kretati. Bežične mreže nemaju kompleksno podešavanje i administraciju infrastrukture, omogućavajući uređajima da kreiraju i pristupaju mrežama „na putu” – bilo gde, bilo kada [6].

5.1. Postupak povezivanja tri Raspberry Pi uređaja u Ad-Hoc mrežu

Za kreiranje Ad-Hoc mreže poželjno je koristiti isti model Raspberry Pi-a. Pored toga, treba obratiti pažnju da ista verzija Raspbian OS bude na svakom Raspberry Pi-u, jer će se u suprotnom javiti greška kod uspostavljanja veze. Za ovaj konkretan primer, korišćena su tri uređaja Raspberry Pi 3 Model B sa operativnim sistemima Raspbian, instaliranim pomoću “imidž” fajla “NOOBS” verzije 2.4.0 sa datumom pojavljivanja 10.04.2017. Kreiranje Ad-Hoc mreže između Raspberry Pi-eva se radi na identičan način za svaki Raspberry Pi, jedina razlika će

biti IP adresa. Za kreiranje novog fajla za podešavanja Ad-Hoc mreže koristi se komanda: „sudo nano interfaces-adhoc“. Nakon kucanja prethodne komande, dobija se prazan prozor u koji se unosi sledeće:

```
auto lo
iface lo inet loopback
iface eth0 inet dhcp
auto wlan0
iface wlan0 inet static
address 192.168.1.1
netmask 255.255.255.0
wireless-channel 1
wireless-essid RPiwireless
wireless-mode ad-hoc
```

Posle kreiranja novog fajla za Ad-Hoc podešavanja, potrebno je instalirati softver koji će omogućiti dodeljivanje IP adresa za svaki uređaj koji želi da se konektuje na mrežu. Komanda je sledeća: „sudo apt-get install isc-dhcp-server“. Ako se slučajno desi greška na kraju kada softver želi da se pokrene, ne treba obratiti pažnju na to, potrebno je samo instalirati navedeni softver.

Posle instaliranja navedenog softvera, kreira se novi fajl koji se naziva „dhcpd.conf“, koji je potrebno izmeniti na sledeći način:

```
“sudo nano /etc/dhcp/dhcpd.conf”
```

Nakon kucanja prethodne komande, otvoriće se „dhcpd.conf“ fajl. Potrebno je ostaviti samo sledeće redove:

```
ddns-update-style interim;
default-lease-time 600;
max-lease-time 7200;
authoritative;
log-facility local7;
subnet 192.168.1.0 netmask
255.255.255.0 {
    range 192.168.1.5 192.168.1.150;
}
```

Nakon unosa gore navedenog, potrebno je pritisnuti tastere „Ctrl+X“, izabere se „Yes“ i pritisne se taster „enter“. U poslednjem koraku, kuca se sledeća komanda:

```
sudo cp /etc/network/interfaces-adhoc
interfaces.
```

Posle ovih komandi, potrebno je restartovati Raspberry Pi [7].

6. REŠAVANJE SISTEMA LINEARNIH JEDNAČINA KORISTEĆI TRI RASPBERRY PI-A PREKO AD-HOC MREŽE

Kodovi koji se koriste za rešavanje sistema linearnih jednačina napisani su u programskom jeziku Python. Raspberry Pi treće generacije ima u sebi ugrađen program koji se naziva „Idle Python 3.x“. Pored navedenog programa, korišćen je i poseban modul za grafički prikaz rezultata svih jednačina u realnom vremenu, koji se

naziva „Matplotlib“. Ovaj modul je besplatan i koristi grafički korisnički interfejs (GUI) od MATLAB-a [8].

Pošto modul „Matplotlib“ nije instaliran po default-u, potrebno ga je instalirati na svakom Raspberry Pi-u. To se radi pomoću sledeće komande, koja se kuca u „Terminal“ Raspberry Pi-a: sudo apt-get install python3-matplotlib.

Prvi Raspberry Pi ima IP adresu 192.168.1.1 i računa jednačinu (1):

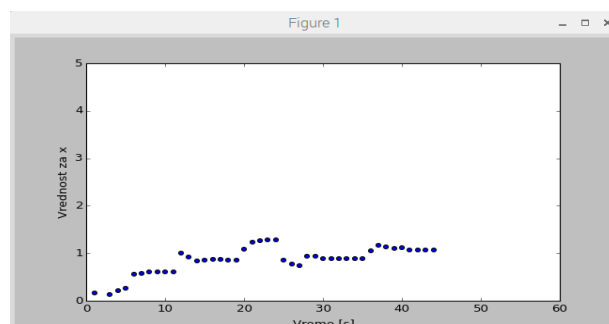
$$x = \frac{16 - 5y + 3z}{7} \quad (1)$$

Pošto ovaj Raspberry Pi šalje prvi, potrebno je da izračuna x, koji će se proslediti na ostala dva Raspberry Pi-a. Zbog ovog koraka, manuelno su dodeljene vrednosti 0 i za y i za z. Vrednost koji se dobija, šalje se dalje na oba Raspberry Pi-a. Svaki Raspberry Pi ima svoju IP adresu, pa se tačno zna kome se šalje i od koga se prima. Broj port-a koji se koristi kod slanja i prijema nije toliko važan, jedina stvar na koju treba da se obrati pažnja jeste da se koristi isti broj porta na svakom Raspberry Pi-u. Za tu svrhu poželjno je koristiti broj veći od 1024, pošto su manje vrednosti rezervisane za druge potrebe.

Kod je tako napisan da ne čeka obe vrednosti da se prime, već računa sa prvim primljenim, a za drugu vrednost ako prethodno nije primljeno ništa, uzima se 0, a u slučaju da je već prethodno primljena neka vrednost, onda se koristi poslednja primljena vrednost. Posle svakog izračunavanja vrednosti x, nacrtat se tačka na grafikonu u realnom vremenu, tj. može se pratiti kako radi kod.

Prikaz tih rezultata dat je na slici 2. Pošto while petlja izvršava kod do beskonačnosti, korišćen je tajmer koji će zaustaviti kod nakon željenog vremena.

Na osnovu slike 2, vidi se da je rezultat u okolini vrednosti 1, što odgovara stvarnom rešenju ove jednačine.



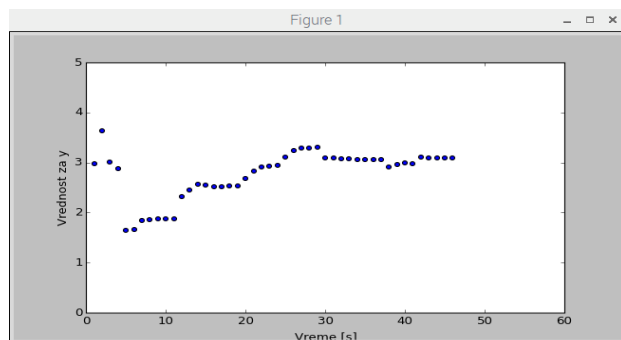
Slika 2. Prikaz rezultata nakon rešavanja linearnih jednačina za prvi Raspberry Pi

Drugi Raspberry Pi ima IP adresu 192.168.1.2 i računa jednačinu (2):

$$y = \frac{-8 - 3x - 2z}{-5} \quad (2)$$

Prikaz rezultata je dat na slici 3. U ovom kodu nije korišćen tajmer koji će zaustaviti računicu, pošto prvi i treći Raspberry Pi već ima taj deo u svom kodu koji će zaustaviti slanje vrednosti x i z.

Na osnovu slike 3, vidi se da je rezultat u okolini vrednosti 3, što odgovara stvarnom rešenju ove jednačine.

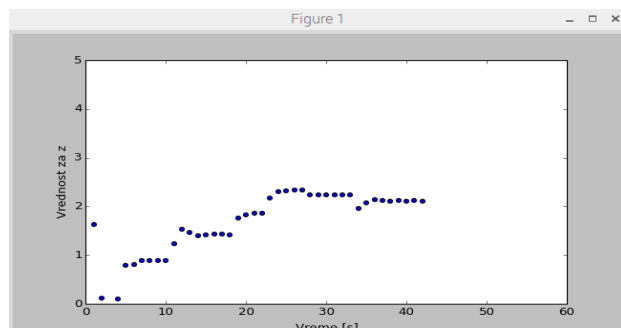


Slika 3. Prikaz rezultata nakon rešavanja linearnih jednačina za drugi Raspberry Pi

Treći Raspberry Pi ima IP adresu 192.168.1.3 i računa jednačinu (3):

$$z = \frac{-5x - 3y}{-7} \quad (3)$$

Ovaj kod radi na istom principu kao i prethodna dva, a rezultat je dat na slici 4.



Slika 4. Prikaz rezultata nakon rešavanja linearnih jednačina za treći Raspberry Pi

7. ZAKLJUČAK

Zahvaljujući raznoj dodatnoj opremi i programskim jezicima koji postoje za Raspberry Pi, mogu se napraviti razni projekti u svim oblastima koji postoje. Raspberry Pi može da se koristi i u oblasti poštanskog saobraćaja, npr. za praćenje poslovanja šalterskih radnika ili kurira koji rade na terenu, tako što se podaci sa računara ili drugog elektronskog uređaja proslede u određenu bazu podataka i na osnovu tih podataka napravi se kratak kod za iscrtavanje rezultata u realnom vremenu.

8. LITERATURA

- [1] <https://www.raspberrypi.org/help/faqs/#intro> (pristupljeno u julu 2018.)
- [2] https://en.wikipedia.org/wiki/Raspberry_Pi (pristupljeno u julu 2018.)
- [3] <https://cdn-shop.adafruit.com/datasheets/pi-specs.pdf> (pristupljeno u julu 2018.)
- [4] https://sr.wikipedia.org/sr-el/programski_jezik (pristupljeno u julu 2018.)
- [5] <https://www.computerhope.com/jargon/num/1gl.htm> (pristupljeno u julu 2018.)
- [6] https://sr.wikipedia.org/wiki/Bežična_ad-hok_mreža (pristupljeno u julu 2018.)
- [7] <http://duinokit.com/ShowAndTell/index.php?topic=50.0> (pristupljeno u julu 2018.)
- [8] <https://en.wikipedia.org/wiki/Matplotlib> (pristupljeno u julu 2018.)

Kratka biografija:



Silard Ištvan rođen je 25.06.1994. godine u Bačkoj Topoli. Osnovnu školu završio je u Srbobranu. Godine 2009. upisuje Tehničku školu u Bečeju i završava je 2013 godine. Odmah posle srednje škole upisuje osnovne akademske studije na Fakultetu tehničkih nauka, Departmana za saobraćaj, smer poštanski saobraćaj i telekomunikacije, koje uspešno završava 2017. godine. Iste godine upisuje master akademske studije na istom fakultetu.



Dejan Nemec rođen je 1972. god. Diplomirao, specijalizirao i magistrirao je na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva. Oblast interesovanja su telekomunikacije i obrada signala.

U realizaciji Zbornika radova Fakulteta tehničkih nauka u toku 2017. godine učestvovali su sledeći recenzenti:

Aco Antić	Dušan Gvozdenc	Milan Trifković	Slavko Đurić
Aleksandar	Dušan Kovačević	Milan Trivunić	Slobodan Dudić
Erdeljan	Dušan Uzelac	Milan Vidaković	Slobodan Krnjetin
Aleksandar Ristić	Duško Bekut	Milena Krklješ	Slobodan Morača
Bato Kamberović	Đorđe Ćosić	Milica Kostreš	Sonja Ristić
Biljana Njegovan	Đorđe Lađinović	Milica Miličić	Srđan Kolaković
Bogdan	Đorđe Obradović	Milinko Vasić	Srđan Popov
Kuzmanović	Đorđe Vukelić	Miloš Slankamenac	Srđan Vukmirović
Bojan Batinić	Đura Oros	Milovan Lazarević	Staniša Dautović
Bojan Lalić	Đurđica Stojanović	Miodrag	Stevan Milisavljević
Bojan Tepavčević	Filip Kulić	Hadžistević	Stevan Stankovski
Bojana Beronja	Goran Sladić	Miodrag Zuković	Strahil Gušavac
Branislav Atlagić	Goran Švenda	Mirjana	Svetlana Nikoličić
Branislav Nerandžić	Gordana	Damnjanović	Tanja Kočetov
Branislav Veselinov	Milosavljević	Mirjana Malešev	Tatjana Lončar
Branislava Kostić	Gordana Ostojić	Mirjana Radeka	Turukalo
Branislava	Igor Budak	Mirko Borisov	Todor Bačkalić
Novaković	Igor Dejanović	Miro Govedarica	Toša Ninkov
Branka Nakomčić	Igor Karlović	Miroslav	Uroš Nedeljković
Branko	Ivan Beker	Hajduković	Valentina Basarić
Milosavljević	Ivana Katić	Miroslav Plančak	Velimir Čongradec
Branko Škorić	Ivana Kovačić	Miroslav Popović	Velimir Todić
Cvijan Krsmanović	Jasmina Dražić	Mitar Jcanović	Veran Vasić
Damir Đaković	Jelena Atanacković	Mladen Kovačević	Veselin Avdalović
Danijela Lalić	Jeličić	Mladen Radišić	Veselin Perović
Darko Čapko	Jelena Borocki	Momčilo Kujačić	Vladimir Katić
Darko Marčetić	Jelena Kiurski	Nađa Kurtović	Vladimir
Darko Reba	Jelena Radonić	Nebojša Pjevalica	Radenković
Dejan Ubavin	Jovan Petrović	Neda Pekarić Nađ	Vladimir Strezoski
Dragan Ivanović	Jovan Vladić	Nemanja	Vladimir Škiljajica
Dragan Ivetić	Jovanka Pantović	Stanisavljević	Vlado Delić
Dragan Jovanović	Karl Mičkei	Nenad Katić	Vlastimir
Dragan Kukolj	Katarina Gerić	Nikola Brkljač	Radonjanin
Dragan Mrkšić	Ksenija Hiel	Nikola Đurić	Vuk Bogdanović
Dragan Pejić	Laslo Nađ	Nikola	Zdravko Tešić
Dragan Šešlija	Leposava Grubić	Jorgovanović	Zoran Anišić
Dragana Bajić	Nešić	Nikola Radaković	Zoran Brujić
Dragana	Livija Cvetičanin	Ninoslav Zuber	Zoran Jeličić
Konstantinović	Ljiljana Vukajlov	Ognjen Lužanin	Zoran Mijatović
Dragana Šarac	Ljiljana Cvetković	Pavel Kovač	Zoran Milojević
Dragana Štrbac	Ljubica Duđak	Peđa Atanasković	Zoran Mitrović
Dragiša Vilotić	Maja Turk Sekulić	Petar Malešev	Zoran Papić
Dragoljub	Maša Bukurov	Predrag Šiđanin	Željen Trpovski
Novaković	Matija Stipić	Radivoje Dinulović	Željko Jakšić
Dragoljub Šević	Milan Rackov	Radovan Štulić	
Dubravka Bojanić	Milan Rapajić	Rastislav Šostakov	
Dušan Dobromirov	Milan Simeunović	Slavica Mitrović	